Language and Pronunciation Modeling in the CMU 1996 Hub 4 Evaluation

Kristie Seymore, Stanley Chen, Maxine Eskenazi, and Ronald Rosenfeld

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

ABSTRACT

We describe several language and pronunciation modeling techniques that were applied to the 1996 Hub 4 Broadcast News transcription task. These include topic adaptation, the use of remote corpora, vocabulary size optimization, n-gram cutoff optimization, modeling of spontaneous speech, handling of unknown linguistic boundaries, higher order n-grams, weight optimization in rescoring, and lexical modeling of phrases and acronyms.

1. INTRODUCTION

The language modeling component of the CMU 1996 Hub 4 system was developed through a series of experiments in topic adaptation, the use of remote corpora, vocabulary size optimization, n-gram cutoff optimization, modeling of spontaneous speech, handling of unknown linguistic boundaries, higher order n-grams, weight optimization in rescoring, and lexical modeling of phrases and acronyms. These experiments were carried out using the Sphinx-III speech recognition system: one language model was used with the Sphinx-III decoder to generate N-best lists for each utterance, and these lists were then rescored with an additional language model to produce the final hypotheses.

Much of the work was focused on topic adaptation. Experiments in topic adaptation have shown promise in terms of perplexity and word error rate (WER) reduction. We present initial results in fine-tuned story adaptation, where the most similar topic-specific language models to a particular story are identified from over 5000 possible models. The chosen models are then interpolated at the word level with a general model, and the resulting model is used for N-best rescoring.

2. LANGUAGE MODELING

The language model vocabulary was chosen to be the 51,000 most frequent words in the Broadcast News training data also present in the CMU pronunciation dictionary. The baseline language model was a trigram model with Katz smoothing trained on the 130M words of Broadcast News training data with singleton bigrams and trigrams excluded. The perplexity of the Hub 4 development set using this model is 231.

2.1. Vocabulary Size Optimization

To investigate the effect of vocabulary size on recognition WER, we used the methodology developed in [8]. The change in WER produced by an increase in vocabulary size is composed of two main factors: an increase in WER due to the increased acoustic confusability between words in the vocabulary, and a decrease in WER due to a decreased out-of-vocabulary (OOV) frequency. To estimate the contribution to WER of acoustic confusability, we compared a trigram model built using a 51k word vocabulary with a trigram model built using a 20k word vocabulary supplemented with words from the development set such that both models have the same coverage on the development set. Any increase in WER for the 51k model can be attributed to increased acoustic confusability since both the 51k model and the supplemented 20k model have the same OOV rate on the development set. We ran experiments for the F0 and F1 conditions of the development set: the 51k model results in a 0.92% higher WER on F0, a 0.36% higher WER on F1, and a 0.61% higher WER on F0 and F1 combined. If we assume that acoustic confusability grows at most linearly and at least logarithmically with vocabulary size, we arrive at the slope values shown in Table 1.

Condition	Linear Slope (per 10 kW)	Logarithmic Slope (per doubling of vocab)
F0	+0.29	+0.68
F1	+0.11	+0.27
F0+F1	+0.19	+0.44

Table 1: Increases in WER due to acoustic confusability as vocabulary size increases.

Condition	OOV Rate difference	WER difference	%WER per %OOV
F0	1.29%	+1.53%	1.19
F1	0.84%	+1.03%	1.23
F0+F1	1.03%	+1.25%	1.21

Table 2: Increase in WER due to OOV's.

To estimate the contribution to WER of changes in OOV rate, we compared the trigram model built with the supplemented 20k vocabulary with a trigram model built with the 20k vocabulary unsupplemented with extra words from the development set. As the two vocabularies are nearly the same size, any difference in performance can be attributed to the change in OOV rate. Table 2 shows the increase in OOV rate and WER of the 20k unsupplemented vocabulary over the 20k supplemented vocabulary. The resulting WER increase per OOV-point is also shown for each condition.

Given an estimate of the OOV rate for a given vocabulary size, we can then estimate the associated WER using the above coefficients. The results of this calculation for F0 and F1 combined are plotted in Figure 1 and show how WER varies with vocabulary size. This figure suggests that a vocabulary in the range of 40k – 60k would be appropriate for the Broadcast News task, at least for F0 and F1. See [8] for analogous experiments on the NAB corpus.

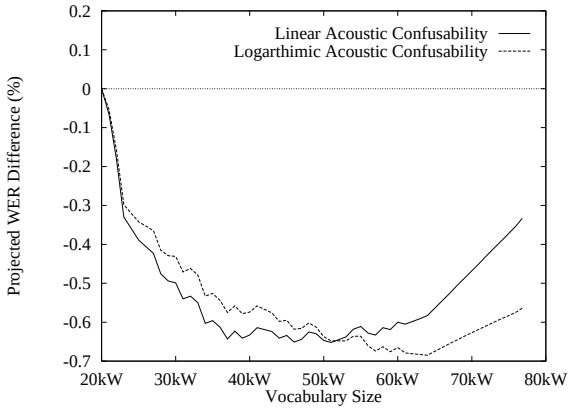

Figure 1: Projected WER based on estimated slopes for acoustic confusability and OOV rate, F0+F1 condition.

2.2. Linguistic Boundaries

In the Hub 4 task, the test data is divided into acoustic segments that do not generally correspond to linguistic segments such as sentences; acoustic segments often contain multiple sentences. While the test data does not contain sentence boundary information, this information is present in the training data. To model those trigrams in the test data that cross sentential boundaries, counts for cross-boundary n-grams were added into the trigram model. That is, for every sentence boundary $w_{-2}w_{-1}$</s><s>w_0w_1 in the training data, the trigrams $w_{-2}w_{-1}w_0$ and $w_{-1}w_0w_1$ were given counts, in addition to the standard trigrams. Adding the cross-boundary trigrams to the standard 51k language model lowered the perplexity on the development set from 231 to 224.

2.3. Spontaneous Speech

Filled pauses were not adequately represented in the language model training data, and their probabilities were severely underestimated in the baseline trigram model. In addition, silence (unfilled pause) events are never represented in transcripts. These problems were addressed by creating a special pause dictionary in the decoder. Each filled pause entry in the pause dictionary was assigned a unigram probability which was estimated from its frequency in the acoustic training data transcripts. The unigram probability of the silence event <sil> was estimated from the forced alignments of the acoustic data. The unigram probabilities were used as the language model score for these events in the decoder. All entries in the pause dictionary along with their unigram probability estimates are shown in Table 3. These events were also skipped by the trigram when it predicted words that follow them. Using this method with a Kneser-Ney smoothed trigram model, perplexity decreased from 211 to 180.

2.4. Weight Optimization

As in common practice, the total score $s(H)$ we assign to a hypothesis transcription H is

$$s(H) = \log p(A|H) + \alpha \log p(H) + p_i l(H)$$

where $\log p(A|H)$ is the *acoustic score*, $\log p(H)$ is the *language score*, α is the *language weight*, p_i is the *word insertion penalty*, and $l(H)$ is the number of words in the hypothesis H. We can re-write this equation as

Event	Probability
UH	0.00866
AH	0.00288
UM	0.00147
EH	0.00071
<sil>	0.10792

Table 3: The pause dictionary.

$$s(H) = \sum_{i=1}^{3} w_i s_i(H) \quad (1)$$

where $w_1 = 1$, $s_1(H) = \log p(A|H)$, $w_2 = \alpha$, $s_2(H) = \log p(H)$, $w_3 = p_i$, and $s_3(H) = l(H)$; *i.e.*, the total score of a hypothesis is a linear combination of several individual scores of the hypothesis. Clearly, we need not restrict ourselves to three scores: as touched on later, we attempted to improve performance by using multiple language scores instead of just one.

In order to combine multiple scores effectively, it is necessary to choose appropriate values for the weights w_i in equation (1). To do this, we use a similar methodology as developed in [6]. We implemented Powell's algorithm as described in *Numerical Recipes in C* [7, pp. 309-317] to automatically search for optimal weights given a set of N-best lists and the corresponding hypotheses' error rates. We search for the values of w_i that minimize the WER of the highest scoring utterance in each N-best list.

To evaluate the WER of a given set of acoustic and language scores on test data, we use two-way cross validation. We split the test data into two halves; in evaluating the number of errors in each half we use weights w_i optimized on the other half of the data. Unless otherwise specified, all WER's in this paper were produced using this methodology.

2.5. Smoothing

We compared two different smoothing techniques for trigram models: Katz smoothing [4] and Kneser-Ney smoothing [5]. Training on 130M words of Broadcast News data, we measured a perplexity of 237 for Katz smoothing and a perplexity of 219 for Kneser-Ney smoothing on the first two-thirds of the development set. By adding extra parameters to Kneser-Ney smoothing,[1] we lowered the perplexity to 211.

We then compared these smoothing techniques on speech recognition WER, by rescoring N-best lists produced from Broadcast News data. On F0 data, Katz smoothing and Kneser-Ney smoothing yielded nearly identical WER's: 19.4% v. 19.5%.[2]

[1] Instead of using a single absolute discount D, we use separate discounts D_1, D_2, and D_{3+} for 1-counts, 2-counts, and counts 3 and above, respectively. The values of these parameters are optimized on held-out data.

[2] There was a difference in performance when the word-insertion penalty was excluded: Kneser-Ney smoothing yielded 19.9% WER while Katz smoothing yielded 20.8% WER. This indicates that perhaps the word-insertion penalty compensates for the difference in performance seen in perplexity between the two smoothing techniques.

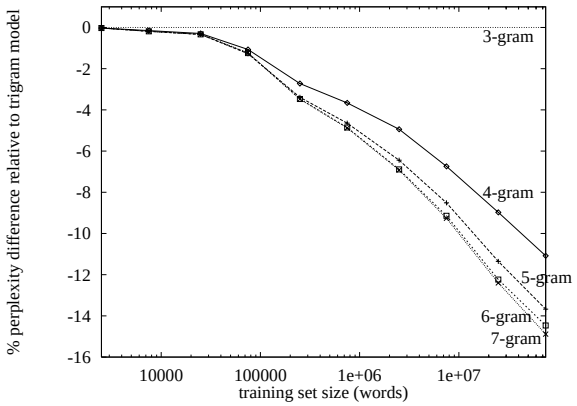

Figure 2: Perplexity of test data of n-gram models for various n relative to a trigram model, for a range of training set sizes, AP news data.

2.6. n-Gram Cutoff Optimization

Several different bigram and trigram cutoff combinations were tested for the language model with Katz smoothing. Perplexity results are shown in Table 4. Neither singleton trigrams nor singleton bigrams proved significant in N-best rescoring.

Bigram Cutoff	Trigram Cutoff	Cross-boundary Trigrams?	Perplexity
0	0	yes	223
0	1	yes	222
1	1	yes	224
1	1	no	231
0	2	no	230

Table 4: Perplexity results using different bigram and trigram cutoffs.

2.7. Remote Corpora

A language model was built from the 230MW North American Business News corpus, and was interpolated with the 51k baseline language model at the word level. Using a weight of 1/3 for the NAB model, development set perplexity was reduced by about 10%. Using the interpolated language score for N-best rescoring did not improve recognition results for F0 and F1, and only slightly improved results for F2.

2.8. Higher-Order n-Gram Models

We investigated the use of higher-order n-gram models, considering models as large as a 7-gram with no cutoffs. To make these models practical, we construct only those parts of the models required to evaluate the given test data.

In Figure 2, we display the reduction in perplexity on test data relative to a trigram model of n-gram models for various n on AP news data.[3] The x-axis describes the size of the training set used. For the right-most point in the graph (corresponding to a training set of about 75M words), a 7-gram model has 15% lower perplexity than a trigram model. In addition, from the graph it seems likely this difference will be greater for larger training sets. These results indicate that it may be worthwhile to use higher-order n-gram models when a large amount of training data is available. When trained on 130M words of Broadcast news data, a 7-gram model yielded about a 10% lower perplexity than the corresponding trigram model.

However, using higher-order n-gram models produced mixed results in terms of speech recognition WER on Broadcast News data. With one acoustic model, the use of a 7-gram model in N-best list rescoring resulted in a reduction in WER from 19.5% to 18.8% on F0 and 40.8% to 39.8% on F3 over a trigram model. However, with a different acoustic model the use of a 7-gram model resulted in an increase in WER of about 0.1% absolute.

[3] We use Kneser-Ney smoothing [5].

3. LEXICAL MODELING

In lexical modeling, we tried to better represent increased coarticulation for spontaneous speech as well as frequent acronyms. To detect pronunciation modeling weaknesses, the decoder was run on the development set and the training set (F0 and F1 only) of the 1996 data, producing a word lattice for these utterances. A reference word or word sequence that did not show up in the word lattice was flagged as a potential pronunciation modeling error. The sequence was not flagged if it showed up within 10 frames of its expected segmentation, as defined by the forced alignment of the reference transcript. The flagged sequences were examined manually by viewing the waveform, listening to the whole speech file, looking at the decode lattice, and referring to the 51K dictionary (the SPHINX III lexicon) for the existing pronunciations. When it was determined that the pronunciation model was lacking, corrective action was taken. For some words, alternate pronunciations were added. In the case of strings of short words, such as *I want to* being pronounced `/AY W AH N AH/`, the whole phrase was added as a separate entry in the dictionary. About 250 phrases, including multiple pronunciations, were added in this way.

In addition, the 147 most frequent acronyms in the BN language corpus, representing some 85% of the acronym tokens, were added as entries in the lexicon (besides being represented letter by letter — *i.e.* `C._N._N.` as well as C. and N.). The main motivation behind this was the hypothesis that sequences of short, acoustically confusable words such as individual letter names, are likely to lead to search errors.

Using both the phrases and the acronyms, preliminary results on the F0 portion of the development set showed a 0.4% absolute reduction in WER. Results on the F1 portion (where more coarticulation effects are expected) could not be compared in a controlled way due to changes in other components of the system, but we estimate that the WER improvement there was significantly higher.

4. TOPIC ADAPTATION

We are currently looking at methods of topic adaptation in unrestricted domains, using the BN domain as our testbed due to its semantic richness. Adapting statistical language models using topic information has been successful in the past (for example, [1, 3, 10]), but the majority of adaptation attempts have focused either on a one-of-N classification, where a new document is assumed to belong to only one of a (typically small) number of disjoint topic sets, or on coarse topic classification, where only a few topics are defined. But in real applications, every document, story or conversation is typically a unique and hitherto unseen combination of several elemental

topics. We are experimenting with a language model adaptation scheme that takes a new piece of text and finds the most similar topics from over 5000 clusters from the training data. Stories from the Broadcast News corpus that share similar topics are gathered into a set of clusters based on manually-assigned keywords that were present in the corpus. The ($tf \times idf$) measure, popular in information retrieval, is used to find the clusters that are most similar in topic to the text we are decoding. Language models built from the most similar topic-specific training clusters are interpolated with a general trigram language model, and N-best hypotheses are rescored with a topic-specific language model score. We report on a series of experiments designed to investigate the reductions in perplexity and word error rate made possible by such adaptation.

4.1. Clustering

In the Broadcast News corpus, story boundaries are marked and keywords have been manually assigned to each story. Topic clusters are created by defining each unique keyword as a label for a cluster. For each keyword, all stories that have that keyword are assigned to its particular cluster. Each keyword-cluster is then a candidate to be used in future adaptation.

An interesting feature of this type of clustering is the presence of data overlap between clusters. If one story contains five different keywords describing its content, then the text for the story will appear in five different clusters. Data overlap between clusters does not present a problem when calculating the similarity between each cluster and a new piece of data. However, if agglomerative clustering were to be used to merge similar clusters in order to reduce the number of distinct topics in the training data, the effects of data overlap on the measure of cluster similarity would need to be considered. Excluding the overlapping data from all similarity calculations may be sufficient; however, valuable topic information is lost in the clustering decision process by not considering stories where two nodes are obviously related. Other possible solutions include assigning half of each duplicated story to each leaf, or using supervised clustering to make reasonable decisions.

Agglomerative clustering has been used successfully for topic adaptation in a mixture modeling framework [1, 3]. However, one advantage of retaining a high number of individual topic clusters, instead of merging the clusters down to a small number, is the ability to make fine distinctions between different subjects and mix unusual topics together that may occur in a future story. As similar clusters are merged together, they lose their topic focus, but they acquire the advantage of having additional data to build more statistically sound language models.

One way to combine the advantages of having larger clusters due to agglomerative clustering and having the topic focus of a large number of individual clusters is to build a *topic tree*. The basic clusters defined by the keywords from the corpus constitute the leaves of the tree, and agglomerative clustering is used to merge similar clusters together up towards the root. When complete, each path from leaf to root specifies a set of nodes that start out in a very distinct topic, and then gradually become more general as the clusters become larger. At runtime, automatic topic identification is performed on a decoded document and results in a small number of active leaf topics. Language models built at various nodes along the active paths from leaf to root can be combined to best model the current document. The language models along the active paths benefit from additional data, whereas leaf models, which may be quite small, retain the advantage of being very specific. Since automatic topic clustering does not always result in optimal clustering decisions, we are currently investigating semi-automatic methods where the system asks for cues whenever its confidence in its clustering decision is weak.

4.2. Finding Similar Clusters

Once we have a set of topic clusters, the text in each cluster can be represented as a vector containing a weighted entry for each unique word. Formally, if a cluster contains t distinct words, the cluster text can be represented as a t-dimensional vector of weights $\mathbf{D}_i = (w_{i1}, w_{i2}, w_{i3}, \ldots, w_{it})$, where one weight is assigned to each unique word. The weight of each word in the vector is given by the ($tf \times idf$) measure frequently used in information retrieval [9]:

$$w_{ik} = tf_{ik} \log(N/n_k) \tag{2}$$

The term frequency, *tf*, is the number of times word k appears in cluster i. The inverse document frequency component, *idf*, computes the log of the ratio of N, the total number of clusters, to n_k, the number of clusters containing word k. This weighting function assigns high values to topic specific words, which are those words that appear with high frequency within one cluster but appear in relatively few other clusters. Words that occur in many clusters, or that occur with low frequency, are deemed more general and are assigned low weights.

Given a new text represented by weight vector $\mathbf{D}_j$, the topic similarity between cluster i and the new text can be computed with the following cosine measure [9]:

$$sim(\mathbf{D}_j, \mathbf{D}_i) = \frac{\sum_{k=1}^{t} w_{jk} w_{ik}}{\sqrt{\sum_{k=1}^{t} (w_{jk})^2 \sum_{k=1}^{t} (w_{ik})^2}} \tag{3}$$

Equation 3 gives the cosine of the angle between the two vectors representing the two sets of text. It is normalized for vector length, so that large clusters are not favored. This similarity measure produces a high value when the two texts being compared are similar, with a value of 1 when they are identical. A similarity value of zero means that the topics of the texts are unrelated.

4.3. Model Interpolation

The similarity between the hypothesized transcription produced by the first decoder pass and each cluster is calculated. Even if the error rate of the original hypothesis is significant, the errors should not be topic correlated, and the correct content words in the hypothesis should provide enough weight for the selection of appropriate clusters. For the experiments presented here, only leaf clusters (elemental topics) are used for topic adaptation. Individual language models are built from the most similar clusters, and the cluster models are interpolated together at the word level with a general language model (the root of the topic tree) using weights obtained by minimizing the perplexity of the hypothesis with the EM algorithm. The N-best lists for the hypothesis are then rescored according to the language score given by the interpolated language models.

4.4. Experiments

The training data used in these experiments for topic adaptation is the Broadcast News corpus obtained from Primary Source Media. The data covers the period from 1992–1995 and consists of 130 million words. Story boundaries are marked, and each story is accompanied by a set of keywords that describe the story's content. The corpus

was split into topic clusters by collecting the keywords from all stories and assigning each keyword to a cluster. The text for each story was assigned to the clusters of the story's keywords. Many of the keywords have sub-categories, in which case the sub-categories were separated from the main keyword and treated as keywords themselves. For the four years worth of data, 8806 topic clusters were created in this manner. The number of topic clusters was then reduced by excluding from the clusters all stories that contained more than six keywords. These stories tend to be summarization reports of many news events. All clusters that contained only one story were also eliminated due to a lack of sufficient data to accurately represent that topic. Additionally, clusters belonging to non-topic keywords, such as U. S. state names, were chosen to be excluded after manual inspection. A total of 5883 clusters remained for topic adaptation. No agglomerative clustering was used in this set of experiments.

The most frequent 63k words from the four years of Broadcast News text defined the vocabulary for calculating cluster similarity. The Hub 4 development set was used as the test set. The story boundaries present in the development set were used to divide the set into 57 stories, with each story containing from 6 to 2131 words.

Two of the largest stories from the test set were chosen for initial adaptation experiments. Story A (791 words) is about the Helms Burton Act and the United States' efforts to keep other countries from doing business with Cuba. Story B (2131 words) discusses the suspicions of drug use by Chinese swimmers during the 1996 Olympics. The similarities between each story and the most data rich 500 and 1000 clusters, as well as all 5883 clusters, were calculated. The 5, 10 and 20 most similar clusters were chosen for each case. The 10 most similar clusters for story B chosen by the $(tf \times idf)$ measure when all 5883 clusters were considered are shown in Table 5.

Story B - Correct Transcript	
Similarity	Cluster Keyword
0.306	China
0.296	Olympic Games
0.252	Olympic Games, Barcelona, 1992
0.244	Favored nation clause
0.212	Chinese Americans
0.212	Drug testing
0.211	Olympic Games, Atlanta, 1996
0.209	Intellectual property rights
0.195	Swimming
0.183	Athletes

Table 5: Ten most similar clusters out of 5883 for Story B.

Our baseline 51k general trigram backoff language model was used for the first-pass Sphinx III recognition hypothesis reported below. The 51k vocabulary was used to create trigram backoff language models from each of the most similar clusters. The cluster language models and the 51k general language model were interpolated at the word level, with weights obtained using half of the correct story transcript. The perplexity was computed using these weights on the other half of the story. The two perplexities computed for each story half were combined to give the overall perplexity of the story when using topic-specific language models. Using only the general 51k trigram language model, we obtain a perplexity of 243 for Story A and 262 for Story B. Perplexity results are shown in Tables 6 and 7.

The lowest perplexities for these two stories were obtained when interpolating the 20 most similar clusters chosen from among all 5883 clusters. Adding additional models may reduce the perplexity even more. The experiments above have an unrealistic component in that the correct story transcripts were used to select the most similar clusters to use for interpolation. Therefore, instead of using the correct transcripts, errorful transcripts for these two stories were next generated by taking the N-best lists (N = 500) output by Sphinx III for the development set, and choosing the highest scoring hypothesis for each segment of the story. The transcripts for both stories have a word error rate of 45%. The similarity between the errorful transcripts and all 5883 clusters was computed, and the top 10 most similar clusters for story B are shown in Table 8. It is interesting to note that even using very errorful transcripts, many of the same clusters are chosen as when using correct transcripts.

Top Matches	Number of clusters considered		
	500	1000	5883
5	227	226	227
10	222	200	226
20	211	203	200
Baseline Perplexity = 243			

Table 6: Perplexity for Story A interpolating different numbers of models.

Interpolating the 5, 10 and 20 most similar language models, optimizing interpolation weights on half of the correct story transcript at a time, yields the perplexity values shown in Table 9. The addition of errors into the hypothesis transcript hurts the perplexity performance of the topic models on the correct story text. However, the adaptation still improves perplexity over the baseline performance by 8% for Story A and 16% for Story B.

Most importantly, we'd like to know if using the interpolated language model weights will help improve the word error rate of Story A and Story B in an N-best rescoring paradigm. Rescoring the combined N-best lists (N = 500) from both stories (2922 words) with the original acoustic score, a language score and a word insertion penalty results in the WER's shown in Table 10. The *FP* column indicates whether or not filled pauses were predicted from their unigram probabilities, and the *Posterior* column indicates whether or not the model interpolation weights were weighted by the unigram probability of the last word in the history [2].

Rescoring Stories A and B with the topic language score results in a lower word error rate (41.7%) than using the Katz trigram score (42.6%). However, more improvement was obtained by rescoring with the Kneser-Ney trigram model (40.9%). The evaluation set was

Top Matches	Number of clusters considered		
	500	1000	5883
5	211	211	222
10	210	211	204
20	210	210	199
Baseline Perplexity = 262			

Table 7: Perplexity for Story B interpolating different numbers of models.

Story B - Errorful Transcript	
Similarity	Cluster Keyword
0.377	China
0.308	Favored nation clause
0.279	Chinese Americans
0.279	Olympic Games
0.261	Intellectual property rights
0.246	Chinese in the United States
0.243	Olympic Games, Barcelona, 1992
0.225	Wu, Harry
0.223	Civil rights
0.216	Zemin, Jiang

Table 8: Ten most similar clusters out of 5883 for Story B.

rescored using the topic language score (N = 200.) The topic score lowers the 2nd pass decoder output WER from 35.5% to 35.3%, but the Kneser-Ney score results in a WER of 34.9%. Although the topic score decreases the overall WER, better results are obtained by rescoring with a Kneser-Ney trigram model. Future work will focus on Kneser-Ney smoothing for topic models, agglomerative clustering, and model selection and interpolation in the context of a topic tree.

5. EVALUATION SYSTEM

Our final evaluation system employed two different language models. A Katz-smoothed trigram language model with cross-boundary trigrams, excluding singleton trigrams and bigrams, using the 51k vocabulary that was supplemented with 208 phrases and 147 acronyms, was used for two decoding passes. After the two passes, N-best lists (N = 200) were generated from the decoder lattices, and the N-best hypotheses were rescored using a Kneser-Ney-smoothed trigram model with no cutoffs. Both language models predicted pauses using the unigram probabilities shown in Table 3.

After two decoder passes, the overall word error rate on the evaluation test set was 35.5% (PE) and 36.5% (UE). N-best rescoring resulted in a 0.6% WER decrease to 34.9% (PE) and 35.9% (UE). In absolute terms, N-best rescoring had the most effect on conditions F2 (-1.1% PE, -0.7% UE), F5 (-1.6% PE, -3.3% UE) and F6 (-2.0% PE, -2.0% UE).

6. ACKNOWLEDGMENTS

This research was sponsored by the Department of the Navy, Naval Research Laboratory under Grant No. N00014-93-1-2005. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The first author is additionally supported under a National Science Foundation Graduate Research Fellowship.

References

1. P. Clarkson and A. Robinson. Language model adaptation using mixtures and an exponentially decaying cache. In *Proceedings of ICASSP-97*, 1997. To appear.
2. Mitch Weintraub et al. Fast training and portability. In *1995 Language Modeling Summer Research Workshop: Technical Reports*, Center for Language and Speech Processing, Johns Hopkins University, Baltimore, 1995.
3. R. Iyer and M. Ostendorf. Modeling long distance dependence in language: Topic mixtures vs. dynamic cache models. In *Proceedings of the ICSLP*, pages 236–239, 1996.
4. Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-35(3):400–401, March 1987.
5. Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 181–184, 1995.
6. M. Ostendorf, A. Kannan, S. Austin, O. Kimball, R. Schwartz, and J. R. Rohlicek. Integration of diverse recognition methodologies through reevaluation of n-best sentence hypotheses. In *Proceedings of the DARPA Workshop on Speech and Natural Language*, pages 83–87, February 1991.
7. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, Cambridge, 1988.
8. Ronald Rosenfeld. Optimizing lexical and n-gram coverage via judicious use of linguistic data. In *Proceedings of Eurospeech 95*, pages 1763–1766, 1995.
9. G. Salton. Developments in automatic text retrieval. *Science*, 253:974–980, 1991.
10. Satoshi Sekine and Ralph Grisham. NYU language modeling experiments for the 1995 CSR evaluation. In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*, 1995.

Top Matches	5883 Clusters Considered	
	Story A	Story B
5	227	233
10	225	223
20	224	221

Table 9: Perplexity for Stories A and B, choosing clusters with errorful transcripts.

Language Score	FP	Posterior	Story A+B WER
Oracle (best in list)			34.4%
Katz 3-gram	no	N/A	42.6%
Kneser-Ney 3-gram	no	N/A	41.8%
Kneser-Ney 3-gram	yes	N/A	40.9%
Topic	no	no	42.0%
Topic	yes	no	41.7%
Topic	no	yes	42.1%
Topic	yes	yes	41.7%

Table 10: WER's for Stories A and B combined, using different language scores.