

LM 95 Project Report
FAST TRAINING AND PORTABILITY

Project Leader

Mitch Weintraub (SRI)

Members

Yaman Aksu (JHU)

Satya Dharanipragada (JHU)

Sanjeev Khudanpur (U. Maryland)

Hermann Ney (RWTH Aachen)

John Prange (DoD)

Andreas Stolcke (SRI)

Additional contributions by

Fred Jelinek (JHU)

Liz Shriberg (SRI)

April 18, 1996

Abstract

This report summarizes the work of the project group “Fast Training and Portability” at the 1995 Language Modeling Workshop held at the Center for Speech and Language Processing at Johns Hopkins University.

1 Introduction and Summary

Mitchel Weintraub

1.1 Overview

The goal of fast training is to make better use of small data sets. A common class of language models that have been extensively used for speech recognition are N-gram language models. These types of language models contain very little structure and can use millions of parameters to model the probability of word sequences. However, to train such models well can require hundreds of millions of words of training data. Since this amount of data is very difficult and costly to acquire in the LVCSR domain, our first goal is to explore alternative algorithms that can make better use of small data sets.

There were three approaches to make better use of small data sets that our group explored. These approaches were to use:

1. Class Models: These models grouped words into classes. The approach to detecting classes were to use automatic clustering or to use a linguistically-motivated tag set (Section 2.4).
2. Tied-Mixture Multinomial Language Models: Tied-mixture models have been used extensively in acoustic models and are a compact way of sharing model parameters. This approach was extended to language modeling (Section 3).
3. Cache Language Models: These language models have been successfully used in adapting dictation systems and we explored their usefulness for modeling conversational speech (Section 2.5).

A second goal of our group is to explore different algorithms to combine information from different domains. While the first approach tried to make efficient use of a small amount of data from the actual domain, this approach focused on how to make use of language model data from other domains. There are tremendous amounts of textual data that are available from other sources. However, these sources are different from the current application. Our second goal is to explore algorithms that can use information from other domains.

There were three approaches to combine information from multiple domains that our group explored: These approaches were to use:

1. Interpolation of Multiple Language Models: This approach computes a weighted sum of the different language model probabilities. We explored both static and dynamic weighting of the different language models (Section 2.3).
2. Language Model Adaptation using the Minimum Divergence Principle: This approach focused specifically on how to adapt a language model from one domain to another (Section 4).
3. Removing “disfluencies” and using a “Disfluency Cleanup” Model: If disfluencies are a major factor in the difference between written and spoken speech, then removing disfluencies and using a traditional text-based language model might be a useful way of making use of written textual data (Section 5).

1.2 Summary of Results

The algorithms developed during the summer workshop were able to reduce the perplexity of the test set. There were four techniques that were successful in reducing the perplexity: (a) the interpolation of Switchboard and North-American Business News (NABN) models, (b) cache LMs, (c) class LMs, and (d) adapted NABN LMs using maximum-entropy language models that are adapted using minimum divergence.

Some of these perplexity reductions translated into reductions in the word-error rate. Rescoring N-Best lists with different language models was able to significantly reduce the word-error rate. This reduction in word-error rate was significant at the 0.05 level. Techniques (a) interpolating LMs and (d) adapting maximum entropy LMs were both successful in reducing the word-error rate, although the improvement for the maximum entropy model was from a worse initial condition.

The other techniques were not successful in improving performance. Although removing disfluencies did not give any improvement in performance, the work in this area led us to develop a useful diagnostic method for language models, based on local perplexity analysis. It also yielded an unexpected result that forced us to partially revise our prior understanding of the effects of disfluencies on language modeling. Tied-Mixture LMs, and class LMs were unable to improve performance over the baseline models. These techniques suggest that it is very difficult to develop other models that make more efficient use of the given training data.

A by-product of our research efforts was that the language modeling software currently being developed by Andreas Stolcke at SRI was significantly improved with regards to robustness and coverage of new types of models. We also developed an interface that allowed the SRI LM toolkit to be linked to the HTK lattice tools used at the workshop. Finally, we shared our LM tools with other groups in several cases where the existing software proved too inflexible or slow.

1.3 Post-Workshop Update

There have been a number of developments that have occurred after this summer's workshop. Satya Dhara-nipragada and Sanjeev Khudanpur have reimplemented the maximum entropy language model and completed a set of language model adaptation experiments. These experiments were finished after the workshop because the scope of this task both in terms of new software and new algorithms was very large.

The perplexity experiments using the cache language model were conducted at the end of the workshop. After the workshop, Andreas Stolcke and Mitch Weintraub implemented the cache LM for speech recognition. These experiments showed that while both models (the model using knowledge of what the person really said and the model using recognition hypotheses) improved perplexity significantly, they did not reduce the word-error rate.

SRI continued development of its language modeling tools, which are currently being prepared to be released for general use by the speech community.

SRI collected "Switchboard-like" data for the workshop that consisted of both high-quality and telephone speech. The primary purpose of this data was to be able to use the high-quality Sennheiser recordings at the workshop. It became apparent that it would also be useful to collect read versions of these conversations by the same people under the same conditions. SRI has completed the collection of this data and expects to release this data to the community later this year.

The techniques, tools and software that were developed at the workshop have been used by SRI both in its research as well as in the Spanish Call-Home evaluations. SRI's results demonstrated that these same techniques for improved language models could be transferred from our English Switchboard system to be used in other languages.

2 Variations on N-gram based Language Models

H. Ney, A. Stolcke and M. Weintraub

2.1 Introduction

In this section we present a number of incremental improvements on the standard back-off N-gram models that served as a baseline for the language modeling at the workshop. These methods are mostly well-known from the LM literature, and have been tested in domains where vast amounts of training data had been available (such as the WSJ corpus). The primary purpose of these experiments was to assess the effectiveness of these methods in the Switchboard context. We also developed a few improvements, such as the dynamic Bayesian interpolation scheme and a cache language model based on N-best lists.

The various models were evaluated first by their perplexity, as computed on the standard development test set (including 1192 Switchboard and 669 SRI-Switchboard utterances). If a model looked promising its effect on word error rate was assessed by rescoring 2500-best lists for the Switchboard portion of the development test set, which had previously been generated using the baseline trigram model. The language model weighting and word transition penalty used in rescoring were unchanged from the baseline experiments.

2.2 Higher-order N-grams

Before any other variants on N-gram models we tried simply increasing the length of the context used. As can be seen below, a 4-gram model gives a very slight (non-significant) improvement over trigrams, and a 5-gram model gives no further improvement.

Model	PP SWB	WER SWB
3gram (baseline)	92.0	49.73%
4gram	91.2	49.17%
5gram	91.3	-

(Note the baseline result differs slightly from the value shown below since this particular experiment used lattice rescoring as opposed to N-best list rescoring).

As expected, the limited amount of training data in Switchboard seems to limit the effectiveness of N-gram models beyond trigrams.

2.3 Interpolation with an NABN language model

The Switchboard language model suffers from little training data, and hence bad probability estimates. One way to improve these estimates is to interpolate them with those from a larger training corpus. The combined estimates will thus have a smaller variance, at the expense of a bias towards the statistics of the larger corpus. The interpolation weight needs to be chosen carefully to optimize the trade-off between these two factors.

We chose the North American Business News (NABN) training corpus as used by the 1994 ARPA CSR evaluations as the second LM training source. It consists of about 250 million words of text. A standard trigram model was built from the corpus, containing only N-grams covered by the 23K Switchboard vocabulary. The model contained 8.4 million bigrams and 13.4 million trigrams (singleton trigrams were discarded).

In the first experiment, the SWBD and NABN models were mixed in a fixed proportion, according to the formula

$$p(w|h) = (1 - \mu) \cdot p(w|h; \text{SWB}) + \mu \cdot p(w|h; \text{NAB})$$

where h is the trigram history of the current word w . μ is the fixed interpolation weight; values around 0.2 were found to give best results.

The following table compares the fixed interpolated model with both of the component models in isolation.

Model	PP SWB	PP SWB+SRI	WER SWB
SWB only, $\mu = 0$	92.0	101.9	49.69%
Fixed, $\mu = 0.2$	83.0	94.4	48.73%
NABN only, $\mu = 1$	326.4	394.7	-

The interpolation reduces perplexity by about 10%, and reduces the word error by almost 1% absolute.

This model can be slightly improved by making the interpolation weight μ a function of the running history of the N-gram predictor. One rationale for doing this is that the estimate should rely more on one or the other model if the context “looks” more like the corresponding part of the training corpus. The model thus becomes

$$p(w|h) = (1 - \mu(h)) \cdot p(w|h; \text{SWB}) + \mu(h) \cdot p(w|h; \text{NAB})$$

where $\mu(h)$ is the posterior probability of “being in” the NABN domain.

$$\begin{aligned} \mu(h) &= p(\text{NAB}|h) \\ &= \frac{\mu_0 \cdot p(h|\text{NAB})^\alpha}{(1 - \mu_0) \cdot p(h|\text{SWB})^\alpha + \mu_0 \cdot p(h|\text{NAB})^\alpha} \end{aligned}$$

Here μ_0 is the “prior” probability for a NABN context. This value can be set to the best fixed μ value ($\mu_0 = 0.2$). The exponential weight α can be used to smooth the deviations from the prior value. For $\alpha = 1$ we obtain the straight Bayesian estimate, for $\alpha = 0$ the model degenerates to the fixed interpolation model.

The table below shows the performance of the Bayesian interpolation model for various lengths of context to adapt to.

history	best α	PP SWB	PP SWB+SRI	WER SWB
	Baseline	92.0	101.9	49.69%
none	0.0	83.0	94.4	48.73%
unigram	1.0	83.7	91.2	48.43%
bigram	0.1	82.5	89.9	48.55%

The Bayesian interpolation using unigram histories thus gives a small additional 0.3% word error improvement. (The combined difference from the baseline model is significant at the 0.05 level.)

2.4 Class-based models

2.4.1 Part-of-speech based N-grams models

In an attempt to overcome sparse data problems, one can group words into classes. One such approach assigns words to part-of-speech (POS) categories, and models the POS sequence underlying a word sequence using standard N-gram models.

For the following experiment, each word w was mapped to a single, the most likely POS tag $g(w)$. The tag set contained 36 elements and was similar to the one used by the Penn Treebank annotations, but included tags for unknown word and sentence end. A POS-based trigram model of the form

$$P_{POS}(w|g(u), g(v))$$

was interpolated with a word-based trigram model. The interpolation weight μ as shown below was optimized on the test data.

Model	PPL SWB
$\mu = 0.00$: no POS	93.5
$\mu = 0.15$:	90.5
$\mu = 1.00$: pure POS	161.2

The perplexity reduction is very modest, and no rescoring experiments were done with this model.

2.4.2 N-grams extended with POS tags

A related idea is to extend the history of standard N-grams so as to include tags for the more distant words, thus counteracting the sparse data problem. To test this approach we trained N-gram models of various orders from a tagged corpus, and likewise tested on a tagged test set (i.e., assuming knowledge of the tags). In each of these models the first word in the history is replaced by its POS tag.

The perplexities below are not comparable to others reported here due to the different training and test sets.

Model	PP
ww (2gram)	92.0
tww	86.3
www (3gram)	78.2
twww	77.8
www (4gram)	77.4

We can conclude that while adding an extra tag to a word N-gram reduces the predictive accuracy of the LM, the same or even better performance can be achieved by simply adding another word to the history.

2.4.3 Word classes by clustering

An alternative approach to class-based LM seeks to find appropriate word classes in an unsupervised way without relying on prior grouping. This can be achieved by a clustering algorithm that optimizes the word-to-class mapping so as to reduce the entropy of the model on a cross-validation set. This is particularly easy to do in the case of a bi-class model

$$p(w|v) = p_o(w|g(w)) \cdot p_{bi}(g(w)|g(v))$$

The number of classes to work with is fixed. The ”+1” notation below refers to a special class reserved for unseen words. The following table shows perplexity reduction of such a bigram model as a function of the number of classes.

number of classes	training perplexity	test(SWB) perplexity
35+1	153.8	148.9
50+1	143.8	140.0
100+1	121.7	124.3
200+1	106.7	116.9
400+1	94.2	112.6

Since such a model alone is weaker than a standard word trigram model, it needs to be interpolated with a trigram LM to give any improvement:

$$p(w|u, v) = (1 - \mu) \cdot p_{word}(w|u, v) + \mu \cdot p_{class}(w|g(v))$$

The following perplexity results refer to optimized μ values:

number of classes	μ	perplexity
no classes	0.0	93.5
35+1	0.2	90.3
	1.0	148.9
100+1	0.25	89.4
	1.0	124.3
400+1	0.25	90.3
	1.0	112.6

The perplexity improvement over the baseline is around 3%, although slightly better than for fixed POS-based class grammars. No rescoring experiments were done using class-based models.

2.5 Cache Language Models

A cache language model is one that remembers recently seen words (beyond the memory of a standard N-gram model) and raises their probability on the theory that words tend to repeat within a unit of text, conversation, etc.

If the reference word sequence were available a cache LM can be built from an unsmoothed unigram model over the last M words, where M is the size of the cache. In the present simple version the cache is not flushed at conversion or other segment boundaries. The unigram model is then interpolated at a small weight with a standard model.

This “supervised” cache gives the following perplexity, using an interpolation weight of 0.7.

Model	PP SWB	PP SWB+SRI
Baseline	92.0	101.9
+cache (M=100)	88.0	95.4
+cache (M=400)	86.8	94.2
+cache (M=1000)	87.5	94.8

As can be seen, the perplexity reduction is around 5%, and seems to be rather robust to the choice of cache size.

To use a cache model in rescoring we face the problem that the correct hypotheses (and hence to cache contents) is not known. We tried to solve this problem by building the cache from the N-best lists returned by the recognizer. The rationale is that words that are recognized with high confidence will appear in many of the N-best hypotheses and thus receive a higher probability. Furthermore, we have access to not just the past, but also the future of an utterance in the cache. We experimented with two variants:

- The cache unigram for an utterance is built from all N-best lists for utterances in the same conversation, *including* the current utterance.
- The cache unigram for an utterance is built from all N-best lists for utterances in the same conversation, *excluding* the current utterance.

Since the cache model gave us the second-largest perplexity reduction after the NABN interpolated model, we also tried both techniques together. The parameter M below refers to the number of hypotheses used to train the cache model:

Model	PP	WER
Baseline	92.0	49.69%
top-500 cache (+ current utt.)	79.1	49.66%
top-100 cache (- current utt.)	87.5	49.57%
Baseline + NAB	83.7	48.43%
+ top-500 cache (+ current utt.)	74.3	49.29%
+ top-100 cache (- current utt.)	80.4	48.41%

Looking at the perplexities we first notice a very low value for the case where hypotheses for the current utterance are included in the cache. This can be explained by the fact that the language model is in fact conditioned on the acoustic evidence in this case, and is thus no longer an independent knowledge source. These values are therefore not meaningful in comparison with standard perplexities.

However, even excluding current utterances, the perplexity of the cache model is virtually the same as in supervised mode. Surprisingly, though, this improvement does not lead to a significant word-error reduction.

It remains to be seen if a cheating cache model (using the references for past and/or future utterances) would give a significant improvement in recognition performance.

2.6 Conclusion

Of all the various methods tried here to improve N-gram models in the Switchboard domain, the only one that gave a small, yet significant reduction in word error was the interpolation with an external language model trained on out-of-domain text. This suggest that the current N-gram LM technology has indeed reached its limits in this domain, and can only be pushed by adding more data (and possibly being more clever about using out-of-domain data). Other approaches along these lines are described in Section 4.

3 Tied-Mixture Multinomial Language Models

Mitchel Weintraub and Andreas Stolcke

3.1 Introduction

Tied-mixture models have been used extensively for acoustic modeling of speech. For acoustic models, a moderate-sized set of Gaussian probability distributions are used as the “basis set.” This is because every state’s output probability distribution is composed of some weighted combination of these basic distributions. The Gaussian distributions are shared by all models, while each model determines it’s own mixture weights.

For language models, the probability distribution to be estimated is the probability of the next word. Since words are discrete events, we use a discrete probability distribution, which we refer to as a multinomial distribution. Each multinomial distribution is a vector that contains the probabilities of the words:

$P(\text{word}_i|\text{multinomial}_j)$. There is a set of common multinomial models that are shared among all distributions. If you want to estimate some probability distribution for the next word, the distribution can be computed as a weighted combination of the common models. The basic model has the form shown below in Equation 1:

$$P(w_i|Hist) = \sum_j P(w_i|mult_j)P(mult_j|Hist) \quad (1)$$

$P(w_i|mult_j)$ is the size of the vocabulary V and $P(mult_j|Hist)$ is the size of the number of multinomial mixtures, typically on the order of 50 to 1000.

The motivation of this model is similar to that used for acoustic modeling: can we estimate a set of basis vectors that can be used to approximate the true language model distributions? In the next section we will describe how this model is trained.

3.2 Estimation of Model Parameters

The multinomial models are initialized using a bigram back-off model. A number of previous words are selected as the seed histories. For each word used as a seed history, the back-off bigram probability distribution over the Switchboard vocabulary conditioned on this seed word is used.

To select the seed words, information about the tagged corpus computed by Yaman Aksu and John Prange were used. For the Switchboard corpus, the frequency distribution for the tagged words was computed, and for each tag, the most likely words having that tag was computed. Each tag was given the same number of seed words, and the most likely words for each tag was used as the seed words.

The mixture weights $P(mult_j|Hist)$ were initialized as uniform distributions. A bigram tied-mixture language model was developed. Therefore, the number of histories used was the size of the vocabulary V .

The equations for re-estimating the mixture weights and multinomials is listed below:

$$P(mult_j|Hist) = \sum_{w_i} P(w_i|Hist) \frac{P(w_i|m_j)P(m_j|Hist)}{\sum_{m_k} P(w_i|m_k)P(m_k|Hist)} \quad (2)$$

$$P(w_i|mult_j) = \sum_{Hist} \frac{P(m_j|w_i, Hist)P(w_i|Hist)}{\sum_{w_k} P(m_j|w_k, Hist)P(w_k|Hist)} \quad (3)$$

In Equation 2, the second term is the probability of selecting a particular multinomial model given the history and the current word. This term is weighted by the different words that are observed for that particular history.

In Equation 3, the term represents the probability of observing the current word given the multinomial and the history. However, during the workshop, instead of computing $P(m_j|w_i, Hist)$ as indicated in the above equation, we instead used the average of this term over all word histories: $P(m_j|Hist)$. This error was only discovered after the workshop during the preparation of this paper, and the effect of this error on the estimates of $P(w_i|mult_j)$ has not yet been measured.

3.3 Perplexity Results for Switchboard

After seeding the tied-mixture models and training them, we ran a series of perplexity experiments. The perplexities were computed on the development sets for both the Switchboard and the SRI-Switchboard development sets. The numbers are reported separately for each partition as well as for the overall development test set.

Comparing the results from Tables 1 and 2 indicates that the overall perplexities decrease and seem to level out at approximately the bigram level. Since these are bigram tied-mixture models, the perplexities are similar to the perplexities of the baseline bigram model.

If we look in more detail at how the tied-mixture multinomial model compares with a bigram model (Table 3), we can see that the perplexity of unseen events are much lower for the tied-mixture language model than a conventional bigram language model. Conversely, the perplexities were slightly higher for the conditions when the bigram exists. From this we can see that the tied-mixture model is a smoother model that generalizes better to unseen data at the expense of modeling the frequently occurring data.

	SWBD	SRI-SWBD	OVERALL
32 Multinomials	136	164	147
177 Multinomials	114	140	124
396 Multinomials	112	138	122

Table 1: Perplexity of bigram tied-mixture multinomial model as a function of the number of mixtures when trained and tested on Switchboard.

	SWBD	SRI-SWBD	OVERALL
Bigram	109	135	119
Trigram	93	120	103

Table 2: Baseline N-gram perplexity when trained and tested on Switchboard.

3.4 Maximum Entropy Adaptation of Tied-Mixture Language Models

The structure of the tied-mixture language model makes it relatively straightforward to impose constraints on the model. To experiment with this approach, we decided to train a tied-mixture language model on the NABN corpus and impose the unigram constraints of Switchboard.

The unigram distribution for the Switchboard model can be computed as:

$$P_{\text{SWBD}}(w_i) = \sum_{m_j} P_{\text{SWBD}}(w_i|m_j)P_{\text{SWBD}}(m_j) \quad (4)$$

$$P_{\text{NABN}}(w_i) = \sum_{m_j} P_{\text{NABN}}(w_i|m_j)P_{\text{NABN}}(m_j) \quad (5)$$

$$P_{\text{NABN}}(w_i|m_j) = \frac{P_{\text{NABN}}(w_i|m_j) \frac{P_{\text{SWBD}}(w_i)}{P_{\text{NABN}}(w_i)}}{\sum_{w_k} P_{\text{NABN}}(w_k|m_j) \frac{P_{\text{SWBD}}(w_k)}{P_{\text{NABN}}(w_k)}} \quad (6)$$

Looking at Equation 6, we can see that the multinomial probability vector is scaled by the ratio of the Switchboard and the NABN unigram frequencies, and then normalized to sum to 1.0 (the denominator in Equation 6). After scaling the multinomials in this fashion, the unigram frequencies for the NABN corpus was recomputed using Equation 5. These new estimates for the adapted NABN unigram probabilities were used in Equation 6, and this process was iterated until the unigram probabilities of the maximum entropy adapted NABN corpus matched the Switchboard unigram probabilities. The perplexities that this model gives is shown in Table 4

Notice that the bigram tied-mixture multinomial language model has a lower perplexity (449) than the traditional back-off bigram N-gram language model (497). As we saw from the results in Table 3, we know that the tied-mixture language model is smoother when dealing with words in new contexts that have not been observed in the training data.

	bigram doesn't exist	bigram exists
Bigram LM	17725	61
177 Mult TM LM	10692	69

Table 3: Perplexity as a function of whether the testing bigram occurred in the training data (The testing bigram existed 87% of the time, and 13% of the time a Bigram LM had to back off to unigram probabilities).

Model	Perplexity
Baseline NABN Bigram	497
Baseline NABN Trigram	394
396 Bigram Tied-Mixture Multinomial	449
+ Imposing Switchboard Unigram Constraints	188
+ Smoothing NABN and Switchboard Multinomials	156
+ Smoothing NABN and Switchboard Mixture Weights	132

Table 4: Effect of maximum entropy adaptation of a tied-mixture multinomial model trained on the NABN corpus with the imposition of SWBD unigram constraints. (Perplexities on combined SWBD and SRI-SWBD devtest data.)

Model	Perplexity
396 Bigram Tied-Mixture Multinomial	
+ Imposing Switchboard Unigram Constraints	
+ Smoothing NABN and Switchboard Multinomials	
+ Smoothing NABN and Switchboard Mixture Weights	132
396 Switchboard Bigram Tied-Mixture Multinomial	
interpolated with 396 NABN Bigram Tied-Mixture Multinomial	118
396 Switchboard Bigram Tied-Mixture Multinomial	
interpolated with 396 NABN Bigram Tied-Mixture Multinomial	
+ Imposing Switchboard Unigram Constraints	
+ Smoothing NABN and Switchboard Multinomials	
+ Smoothing NABN and Switchboard Mixture Weights	117

Table 5: Effect of interpolating tied-mixture multinomial model trained and standard N-gram language models.

Imposing the unigram constraints on the language model results in a significant reduction in perplexity (449 to 188). This dramatic reduction in perplexity is due primarily to the correction of low-probability words (e.g. words like UH-HUH that have a very low probability in NABN text but has a high Switchboard probability).

This improvement can be further enhanced by smoothing the adapted language model with the Switchboard language model. In addition to imposing the Switchboard unigram constraints, we can smooth the adapted multinomials with the Switchboard multinomials. We can do this because the NABN multinomials were initialized with the Switchboard Tied-Mixture Multinomials, and therefore the multinomials retain some correspondence between the two domains. This smoothing can be done before interpolating the language models. Smoothing the language models together results in a reduction in perplexity to 132.

This resulting language model can then be interpolated with the Switchboard language model. These results are shown in Table 5.

These results indicate that when interpolating the language models, smoothing the adapted model before interpolation does not improve perplexity much (118 to 117). The resulting perplexity of this model is only slightly lower than the standard bigram language model that has a perplexity of 119. An additional improvement might have been obtained by also imposing Switchboard bigram language model constraints but these algorithms have not yet been developed.

3.5 Recognition Results

The previous experimental results measured perplexity using the different language models. The real test is their effect on recognition performance. The experimental results listed in Table 6 are for rescoring 2500-best

Model	WER
Baseline Bigram	51.42%
Baseline Trigram	49.55%
396 Switchboard Bigram Tied-Mixture Multinomial	51.42%
396 NABN Bigram Tied-Mixture Multinomial	
+ Imposing Switchboard Unigram Constraints	
+ Smoothing NABN and Switchboard Multinomial	
+ Smoothing NABN and Switchboard Mixture Weights	52.33%
396 NABN Bigram Tied-Mixture Multinomial	56.75%
396 Switchboard Bigram Tied-Mixture Multinomial	
Interpolated with 396 NABN Bigram Tied-Mixture Multinomial	
+ Imposing Switchboard Unigram Constraints	
+ Smoothing NABN and Switchboard Multinomials	
+ Smoothing NABN and Switchboard Mixture Weights	51.46%

Table 6: Word-error rate for N-best rescoring experiments on Switchboard development test set.

lists that were generated with a Switchboard N-gram language model.

The results from Table 6 indicate that the bigram tied-mixture multinomial gave the same performance as the bigram N-gram model (51.42). The language model that had the lowest perplexity (interpolated language model perplexity of 117) had a word-error rate of 51.46; this is insignificantly higher than the baseline bigram condition.

It is interesting to notice that rescoring the N-best list with the NABN language model only increased the error rate to 56.75 even though the perplexity for this language model was 449. This is an absolute increase of 5.3% over the case when a task-specific language model has been used. SRI has observed approximately similar increases in performance when using the NABN language model directly. This relatively small difference indicates that (a) there is not much benefit from training a language model using a small data set, and (b) there is not much degradation by using a language model trained on a very large amount of text. We hypothesize that poor acoustic models are limiting the difference between these two conditions, and that at much lower error rates the difference between these conditions should be much greater.

3.6 Analysis of Results

To understand some of the characteristics of the tied-mixture language model, we computed the perplexity of the bigram mixture weights (Table 7). For each word history, there is a probability distribution over the mixture weights. The sharper the distribution of these mixture weights (more probability mass concentrated into a small number of multinomials), the lower the perplexity of this distribution.

The reason that the perplexity of the distribution is extremely low for very high number of successors is that these contexts that have a very large number of counts are effectively dominating the training data for a single multinomial vector. The reason that the perplexity of the distribution is extremely low for very small number of successors is that when you have only one example of a successor word, the maximum-likelihood estimate for the lambdas is to give all your weight to the multinomial vector with the highest probability for that word.

3.7 Summary

The performance of the tied-mixture language model was identical in terms of recognition performance and very close in terms of perplexity. While it did not result in any performance improvement as hoped, it did have lower perplexity when tested on bigrams that did not occur in the training data.

The structure of the language model made it convenient to adapt the model using maximum entropy constraints. In particular, we used Switchboard unigram constraints to adapt a NABN tied-mixture language

# Unique Successors	Perplexity
1	1.0
2	1.79
3	2.45
4	2.89
5	3.34
6	3.61
7	3.78
8	4.26
9	4.32
10-20	4.44
20-50	4.06
50-100	3.10
100-200	2.23
200-500	1.45
500-1000	1.07
1000-10000	1.0

Table 7: Entropy of the mixture weights as a function of the amount of training data for a particular history.

model to the Switchboard domain. Unfortunately, this did not help language model performance. This can be understood since the baseline Switchboard bigram language model has a coverage of 87%.

4 Language Model Adaptation using the Minimum Divergence Principle

S. Dharanipragada, F. Jelinek, and S. Khudanpur

4.1 Introduction

It is well known that reliable estimation of language model probabilities requires a large amount of data. Therefore, for domains such as Switchboard, which has approximately 2.1 million words, language models built using only the domain-specific data will be unreliable and hence ineffective. However, it is often the case that these “sparse domains” have much information common with other “rich domains.” For instance, there is likely to be considerable amount information common between the Switchboard domain and the richer Wall Street Journal domain. Hence, the question that naturally arises is: how can we take advantage of the rich domain to improve the language model on the sparse domain? In this project, we address the above general question with the specific intention of improving language models for the Switchboard task using the Wall Street Journal corpus.

It is clear that in order to utilize the rich domain for developing language models on the sparse domain, we need a way to combine information in the two domains in a suitable manner. To this end, we adopt the following approach:

- First, we select a suitable rich domain and construct a language model for it – we refer to this model as the prior language model.
- Next, we select constraints that describe information that is characteristic to the sparse domain, and that can be reliably estimated.
- Finally, we transform the above language model in such a way that the transformed language model meets the constraints with *minimum deviation* from the prior language model.

Constraints are often on N-gram and/or word-class probabilities and the deviation between two language

models (or probability distributions) is usually measured using the standard discrimination information distance measure also known as the Information Divergence measure or the Kullback-Liebler distance.

In the next section, we give a mathematical formulation for the above approach.

4.2 Mathematical Preliminaries

We consider a trigram language model and adopt the following notation:

V	: a discrete, finite set	(vocabulary + { < UNK > })
w	: element of V	(words)
$P(w_1, w_2, w_3)$: probability mass function on $V \times V \times V$	(trigrams)
$\tilde{P}(w_1, w_2)$: $\sum_{w_3} P(w_1, w_2, w_3)$	(probability of history (w_1, w_2))
$P(w_3 w_2, w_1)$: conditional probability	(language model)
\mathcal{P}	: a set of probability mass functions	(admissible models)

The information divergence (I-Divergence) between two distributions P and Q is given by

$$D(P||Q) = \sum_{w_1} \sum_{w_2} \sum_{w_3} P(w_1, w_2, w_3) \log \frac{P(w_1, w_2, w_3)}{Q(w_1, w_2, w_3)}.$$

We consider linear equality constraints, which can be written using “feature functions” $f_i : V \times V \times V \rightarrow R$ as

$$E_P[f_i] = \alpha_i, \quad i = 1, 2, \dots, K.$$

The constants α_i are chosen to be either the sample means of the feature functions or their Good-Turing estimates.

The language modeling problem can now be stated as:

Given a prior distribution Q , find a distribution P^* that

- satisfies the constraints: $E_P[f_i] = \alpha_i, \quad i = 1, 2, \dots, K.$, and
- minimizes $D(P||Q)$

i.e.,

$$P^* = \arg \min_P D(P||Q) \quad \text{subject to} \quad E_P[f_i] = \alpha_i, \quad i = 1, 2, \dots, K.$$

P^* is also referred to as the I-Projection of Q onto the linear family of distributions

$$\mathcal{P} \triangleq \{P : E_P[f_i(w_1, w_2, w_3)] = \alpha_i, \quad i = 1, \dots, K\}$$

The following theorem characterizes the I-projection of Q onto \mathcal{P} .

Theorem: [3, 2] If \mathcal{P} is a linear family of distributions such that $P(w_1, w_2, w_3) > 0 \forall w_1 w_2 w_3$ for at least one $P \in \mathcal{P}$, then the I-Projection of Q on \mathcal{P} has the form

$$P^*(w_1, w_2, w_3) = \frac{Q(w_1, w_2, w_3) e^{\sum_{i=1}^K \lambda_i f_i(w_1, w_2, w_3)}}{Z(\lambda_1, \dots, \lambda_K)}$$

where $Z(\cdot)$ is a normalizing constant.

When dealing with trigram language models, we are interested in conditional distributions instead of marginal distributions. Therefore, instead of first obtaining joint distributions, and then deriving conditional distributions from the joint distributions, we can formulate the problem directly in terms of conditional distributions. To this end, we define the conditional I-Divergence between two conditional probability distributions P and Q , with a given distribution on the history, \tilde{F} , as

$$\begin{aligned} D(P||Q|\tilde{F}) &= \sum_{w_1} \sum_{w_2} \tilde{F}(w_1, w_2) \sum_{w_3} P(w_3|w_2, w_1) \log \frac{P(w_3|w_2, w_1)}{Q(w_3|w_2, w_1)} \\ &= D(P\tilde{F}||Q\tilde{F}) \end{aligned}$$

\tilde{F} is usually taken to be the empirical joint distribution on the history. The following corollary to the the above theorem is applicable for the conditional I-divergence problem.

Corollary: For the conditional I-Divergence problem

$$P^*(w_3|w_2, w_1) = \frac{Q(w_3|w_2, w_1)e^{\sum_{i=1}^K \lambda_i f_i(w_1, w_2, w_3)}}{Z(w_1, w_2, \lambda_1, \dots, \lambda_K)}$$

The K parameters specifying P^* can be determined by the following iterative algorithm due to Darroch and Ratcliff, widely known as the ‘‘Generalized Iterative Scaling Algorithm.’’

Generalized Iterative Scaling Algorithm [3]

- Set $\lambda_1^{(0)} = \lambda_2^{(0)} = \dots = \lambda_K^{(0)} = 0$
 - $n = 0$
 - Set $P_n(w_1, w_2, w_3) = \tilde{F}(w_1, w_2) \frac{Q(w_1, w_2, w_3)e^{\sum_{i=1}^K \lambda_i^{(n)} f_i(w_1, w_2, w_3)}}{Z(w_1, w_2, \lambda_1^{(n)}, \dots, \lambda_K^{(n)})}$
 - **while (stopping criterion not met)**
 - {
 - for each** $j = 1, 2, \dots, K$
 - {
 - **compute** $\hat{\alpha}_j = E_{P_n}[f_j(w_1, w_2, w_3)]$
 - **compute** $\theta_j = \log \frac{\alpha_j}{\hat{\alpha}_j}$
 - }
 - for each** $j = 1, 2, \dots, K$
 - **update parameters:**
 - $\lambda_j^{(n+1)} = \lambda_j^{(n)} + \theta_j$
 - for each** (w_1, w_2) **in training data**
 - **update normalizing constants:**
 - $Z(w_1, w_2, \lambda_1^{(n+1)}, \dots, \lambda_K^{(n+1)}) = \frac{Z(w_1, w_2, \lambda_1^{(n)}, \dots, \lambda_K^{(n)})}{F(w_1, w_2)} E_{P_n}[e^{\sum_{i=1}^K \theta_i^{(n)} f_i(w_1, w_2, w_3)}]$
 - $n = n + 1$
 - }

4.3 Experimental Results on Switchboard

Approximately 2.1 million words of training data was used for developing trigram language models on Switchboard. The language models were tested on a test set consisting of 1192 sentences ($\sim 10,000$ words) for both Perplexity and Word Error Rates. Word error rates were computed by rescoreing the N-best hypotheses that were provided to us.

We first developed a Maximum Entropy (ME) Model, i.e. with a uniform prior distribution. We selected the North American Business News Corpus (NABN) as the ‘‘rich’’ domain and used a trigram language model developed for it, as the prior distribution for developing a Minimum Divergence (MD) Model for Switchboard. For both the ME and MD models we imposed only N-gram constraints. All trigrams with counts ≥ 3 , all bigrams with *adjusted*¹ counts ≥ 2 and all unigrams with adjusted counts ≥ 1 in the training set were constrained. As target values, we used Good-Turing estimates of the probabilities with thresholds² of 1, 7, 7 for unigram³, bigram and trigram counts respectively. The perplexity and word-error rates of these models are shown in the tables below in comparison with a SWBD Baseline Trigram model.

¹ Only those sample points, (w_1, w_2, w_3) , that do not appear in any trigram constraints contribute to the adjusted count of the bigram (w_2, w_3) .

² A GT threshold of 7 for trigrams, for example, implies that target values for trigrams with counts > 7 , are set to their relative frequencies, while for trigrams with counts ≤ 7 , Good-Turing estimates are used.

³ All unigrams seen with adjusted counts ≥ 1 were considered reliable, and hence a threshold of 1 was used

Perplexity of the LM95 dev-test

Model	N ₀ Bigrams	N ₀ Trigrams	Perplexity
Baseline Trigram	300 K	180 K	92.94
ME Model (with uniform prior)	100 K	95 K	96.90
MD Model (with NABN as prior)	100 K	95 K	86.73

WER Results (2500-best Rescoring)

Model	% Correct	% Sub	% Del	% Ins	% Err	% Sent Err
Baseline Trigram	56.09	32.07	11.85	5.64	49.55	80.45
ME Model (with uniform prior)	55.86	31.66	12.48	5.30	49.44	81.19
MD Model (with NABN as prior)	57.58		11.41		48.18	79.01

Two sets of comparisons may be made based on these results.

1. Measured against the ME language model with a uniform prior, the MD technique (with the NABN trigram language model as a prior) results in a small reduction of perplexity as well as word error rate. This points to the potential gains that may be realized by using language models from data rich domains in new domains where training data is scarce. Since we use the comparison between the ME and the NABN based MD language models to make this claim, it is clear that the gains are indeed due to the information in the prior language model and not due to any inherent advantages of maximum entropy techniques. This is not obvious if the performance of the NABN based MD model is compared with the baseline trigram model alone.
2. The baseline trigram language model and the ME language model result in comparable word error rates even though the ME language model has a higher perplexity. This is consistent with earlier experiences with maximum entropy models and indicates that discarding very low count N-grams in a maximum entropy setting is competitive with using Good-Turing estimates for their counts in the Katz backoff scheme. However, if the discarded N-grams were instead used in some reasonable way, *e.g.* as parts of word class N-gram constraints, it is conceivable that further gains could be made.

4.4 Conclusions and Future Work

The method of adapting a language model developed on a rich (usually text) corpus to a sparse domain, such as Switchboard, using the minimum divergence approach has shown reasonable promise. The experiments we have conducted so far are not exhaustive. We plan to investigate this approach further in the future. In particular, the use of more sophisticated constraints, *e.g.* on classes of words rather than specific words, and other sources of prior models, which appear to be closer to the target domain than text corpora, will be explored. In all our experiments thus far, we have imposed strict equality constraints on only those N-grams that have a significant count. In a sparse domain there will be many n-grams with very low counts. These ngrams nevertheless, provide significant information on the sparse domain and hence, discarding them completely is clearly inappropriate. We thus need a way to incorporate the information provided by these low-count events. We are currently investigating the use of interval constraints (or relaxed constraints) instead of strict equality constraints, to accommodate these low count events in the maximum entropy framework.

5 Language Modeling for Speech Disfluencies

Andreas Stolcke and Elizabeth Shriberg

5.1 Motivation and Overview

Speech disfluencies (DFs) are prevalent in spontaneous speech, and are among the characteristics distinguishing spontaneous speech from planned or read speech. DFs are one of many potential factors contributing to

the relatively poor performance of state-of-the-art recognizers on this type of speech, e.g., as found in the Switchboard [4] corpus.

Past work on disfluent speech has focused on disfluency detection, using either acoustic features [9, 8] or recognized word sequences [1, 5]. Our goal in this work is to develop a statistical language model (LM) that can be used for speech decoding or rescoring, and that improves upon standard LMs by explicitly modeling the most frequent DF types. The main reason to expect that DF modeling can improve the LM is that standard N-gram models are based on word predictions from local contexts, which are rendered less uniform by intervening DFs. Other researchers have recently started exploring approaches to DF modeling based on similar assumptions [6, 10].

Section 5.2 describes a simple N-gram-style DF model, based on the intuition that DF events need to be predicted and edited from the context to improve the prediction of following words. Section 5.3 compares the DF model with a baseline LM, in terms of both perplexities and word error rates on Switchboard data. The emphasis is on a detailed analysis of the model at DF and following word positions. Section 5.4 provides a general discussion of the results.

5.2 The Model

5.2.1 Disfluency types

Following [11], DFs can be classified based on how the actual utterance must be modified to obtain the intended fluent utterance, i.e., the utterance a speaker would produce if asked to repeat his or her utterance. The types can be characterized by the type of editing required.

Filled pauses (FP) The pause filler (typically “uh” or “um”) must be excised.

```
SHE UH GOT REAL LUCKY THOUGH
--> SHE GOT REAL LUCKY THOUGH
```

Repetitions (REP) Contiguous repeated words must be removed.

```
IT'S A IT'S A FAIRLY LARGE COMMUNITY
--> IT'S A FAIRLY LARGE COMMUNITY
```

Deletions (DEL) Words without correspondence in the repaired word sequence must be deleted.

```
I DID YOU HAPPEN TO SEE ...
--> DID YOU HAPPEN TO SEE ...
```

We know from prior work [11] that these three types of DF are the most frequent across a variety of spontaneous speech corpora, accounting for over 85% of DF tokens in the Switchboard corpus.⁴ See [11] for a description of other, less frequent, types of DF that are not modeled explicitly in our LM. For example, we are not modeling word substitutions or speech errors.

5.2.2 The Cleanup Model

The central assumption incorporated in our DF language model is that probability estimates for words after a DF are more accurate if conditioned on the intended fluent word sequence. A secondary assumption is that DFs themselves can be modeled as word-like events, each having a probability conditioned on the context. A standard language model, by contrast, would look only at the surface string of words and assign word probabilities in a strictly sequential manner.

Because of the central assumption, we call our DF model the ‘Cleanup Model.’ It is implemented as a standard backoff trigram model with the following three modifications to account for DFs.

⁴DF frequencies in Switchboard were estimated from a hand-labeled subset of 60 conversation sides, containing 40,500 words. The coverage figure takes into account the further limits on modeled repetitions and utterance-medial deletions described below.

1. Words following a DF event are conditioned on the cleaned-up, fluent version of the context. Filled pauses are removed from contexts, as is the sequence of extraneous words in repetitions and deletions. For example, the probability estimate for “WANT” following “BECAUSE I I” would be

$$P(\text{WANT}|\text{BECAUSE I REP1}) = P(\text{WANT}|\text{BECAUSE I}) ,$$

where **REP1** denotes a repetition event. The repeated “I” is deleted from the context.

2. Disfluencies are represented by probabilistic events occurring within the word stream, some of which are hidden from direct observation. For simplicity, we model only the most prevalent subtypes for each DF class, namely filled pauses **UH** and **UM**, repetitions of one or two words (**REP1**, **REP2**), deletions at the beginning of a sentence (**SDEL**), and other one- or two-word deletions (**DEL1**, **DEL2**).
3. Just as words, DFs are treated as events that are assigned probabilities conditioned on their context. The contexts themselves are subject to DF cleanup as described above. For example, $P(\text{REP1}|\text{BECAUSE I})$ is the probability of repeating “I” after “BECAUSE.”

By representing DFs simply as another type of N-gram event, we allow DFs to be conditioned on specific lexical contexts, so that simple word-based regularities in their distribution can be captured. Furthermore, because of its simple N-gram character, the model does not embody specific assumptions or constraints about the distribution of DF events.

5.2.3 Probability computation

To account for the hidden DF events potentially occurring between any two words, a forward computation is carried out to find the probability of a sentence prefix $P(w_1 w_2 \dots w_k)$. Conditional word probabilities are then computed as

$$P(w_{k+1}|w_1 \dots w_k) = \frac{P(w_1 \dots w_{k+1})}{P(w_1 \dots w_k)} .$$

If the underlying N-gram model is a trigram, it is sufficient to keep eight states for each word position, according to whether the DF prior to w_k was **NODF** (none), **FP** (filled pause), **SDEL**, **DEL1**, **DEL2**, **REP1**, **REP2**, or the second position after a **REP2** event. To illustrate, the partial computation involving just the **NODF** and **REP1** states is shown here.

$$\begin{aligned} P(w_1 \dots w_k \mathbf{NODF} w_{k+1}) &= P(w_1 \dots w_{k-1} \mathbf{NODF} w_k) \\ &\quad p(w_{k+1}|w_{k-1} w_k) \\ &+ P(w_1 \dots w_{k-1} \mathbf{REP1} w_k) \\ &\quad p(w_{k+1}|w_{k-2} w_{k-1}) \\ P(w_1 \dots w_k \mathbf{REP1} w_{k+1}) &= \delta(w_k, w_{k+1}) \\ &\quad [p(w_1 \dots w_{k-1} \mathbf{NODF} w_k) \\ &\quad p(\mathbf{REP1}|w_{k-1} w_k) \\ &+ P(w_1 \dots w_{k-1} \mathbf{REP1} w_k) \\ &\quad p(\mathbf{REP1}|w_{k-2} w_{k-1})] \end{aligned}$$

where $\delta(w_i, w_j) = 1$ if $w_i = w_j$, and 0 otherwise. Trigram probabilities are denoted by $p(\cdot|\cdot)$; these are obtained through the usual backoff procedure [7]. The total prefix probability is then computed as

$$P(w_1 \dots w_k) = \sum_X P(w_1 \dots X w_k) ,$$

where X ranges over the hidden states representing the disfluency types (including **NODF**).

5.2.4 Estimation

The backoff N-gram probabilities in the model are estimated from N-gram counts, including counts of the DF events. We used standard Good-Turing discounting in the backoff for both baseline and DF trigram models. For experiments reported here involving hidden DF events, we used a subset of the Switchboard corpus that was hand-annotated for disfluencies as well as for linguistic segments.⁵ In the absence of hand-annotated training data, an iterative reestimation (EM) algorithm could be used to estimate the N-gram probabilities for hidden DF events.

When counting N-grams for the DF model, the same context modifications used in the DF cleanup operations must be performed on the training data. For example, the word sequence

<s> SHE UH GOT REAL LUCKY

is counted as having the following trigrams:

<s> SHE UH <s> SHE GOT
SHE GOT REAL GOT REAL LUCKY

Note that the trigrams

SHE UH GOT UH GOT REAL

which would be generated for a standard trigram LM are *not* generated for the DF model.

Because DF and word events are represented uniformly as N-grams in the model, the standard estimation procedure will normalize DF and non-DF event probabilities. This is a convenient simplification over alternative approaches in which DFs are modeled separately from the fluent word sequences.

5.3 Results and Analysis

5.3.1 Overall results

We trained a trigram model for FP, REP, and DEL disfluencies as described above, using 1.4 million words of Switchboard data labeled for DF events (see note 5). The model was then evaluated on a test set of 17,500 words. Table 8 compares baseline trigram and DF models.⁶

Table 8: Overall results

Model	Perplexity	Word error
Baseline trigram	119.1	50.21%
DF trigram	120.9	50.23%

As can be seen, there is no significant difference in recognition word error rates. While this may be due to a number of factors (some of which we discuss in Section 5.4), we would have expected at least a reduction in perplexity for the DF model; this was not the case. We wanted to know whether this was because our underlying assumptions were wrong, or whether it was due to other factors, so we decided to analyze the DF model performance in detail.

We note with regard to these and later results that some types of disfluencies may contain *word fragments* (from speakers cutting themselves off in mid-word). According to [11], 20 to 25% of repetitions and deletions in Switchboard contain word fragments; however, filled pauses, as classified here, never involve words fragments. Fragments are usually not part of the vocabulary of current recognizers, and are not modeled in our system. They were therefore omitted from the transcripts used for our perplexity computations. We can expect an additional benefit from successful fragment recognition, since they would serve as extra evidence for repetitions and deletions, as well as for other DF events.

⁵A preliminary version of annotated Switchboard data was made available to the 1995 Johns Hopkins Language Modeling Workshop; the LDC will release a final version.

⁶Both baseline and DF models were trained on the same data, which corresponds to only a portion of the full training corpus. Therefore, the perplexity figures are higher here than in some of the comparisons below.

Table 9: Local perplexities at filled pause positions.

Position	UH	UH+1	UH+2	UM	UM+1	UM+2	non-FP	overall
Baseline	39.0	223.5	89.8	174.9	36.7	71.9	103.4	101.9
FP model	39.9	291.5	91.4	175.8	73.4	69.2	103.4	103.3
#events	502	502	373	188	188	94		19426

Table 10: Local perplexities at repetition DF positions.

Position	REP1	REP1+1	REP1+2	REP2	REP2+1	REP2+2	non-REP	overall
Baseline	47.7	183.4	86.4	111.0	12.6	216.0	102.9	101.9
REP model	38.2	191.5	84.7	94.2	2.1	222.3	103.1	101.3
#events	386	386	320	44	44	44		19426

5.3.2 Analysis by DF type

To assess the potential of the DF model specifically at DF locations, we computed perplexities for models covering each DF type in turn, and separately for a number of word positions relative to the DF event. In each case, we also computed perplexities for the non-DF positions, to make sure the model did not penalize fluent text (reported below in the “non-DF” columns).

Filled pauses A trigram model with special DF modeling for filled pauses only was trained on 1.8 million words of acoustically segmented Switchboard transcripts. The test set consisted of 1861 acoustic segments containing 17,500 words. Table 9 shows the perplexities of the baseline and FP models for the FPs themselves (UH, UM), the words after (UH+1, UM+1), and two words after (UH+2, UM+2). The surprising result is that deleting FPs from N-gram contexts does not help the LM; it actually significantly *increases* the perplexity of the word following the FP. That is, on average, the FP itself is the best predictor of the following word, not the context preceding the FP. This conclusion is also supported by the corresponding bigram perplexities, which exhibit the same pattern. Apparently, FPs correlate strongly with certain lexical choices or syntactic structures, and thus give useful information regarding their neighbors to the right. We investigate this question further in Section 5.3.3.

Repetitions A trigram model with DF modeling for repetitions was trained and tested as described above. Table 10 shows word perplexities for positions relative to repetition events. REP1 refers to the second instance of a repeated word in a one-word repetition; REP1+1 and REP1+2 denote the first and second word, respectively, after such a repetition. REP2 and REP2+1 refer to the repeated words in a two-word repetition; REP2+2 denotes the word following a two-word repetition. Unlike the case of FPs, the Cleanup Model is generally beneficial in REP contexts, reducing the joint perplexity (all the above positions relative to REP) from 85.9 to 76.6.

We also tested whether the words following the repetition might be better predicted by the REP event itself, rather than the actual words being repeated, analogous to what we found for filled pauses; this turned out not to be the case.

Deletions A deletion-only DF model was trained on 1.4 million words of DF-annotated transcripts. In order to perform an analysis by DF type and position, the models were tested on 17,800 words of similarly annotated data.⁷ Only sentence and one-word deletions are reported in Table 11, since the test data contained

⁷Due to differences in amount of training data and type of segmentation, the perplexities are not directly comparable to the previous two studies.

Table 11: Local perplexities at deletion DF positions.

Position	SDEL	SDEL+1	DEL1	DEL1+1	non-DEL	overall
Baseline	415.5	49.5	523.3	35.0	75.5	76.2
DEL model	402.0	47.9	544.2	36.0	75.4	76.1
word event only	99.1		289.5			
#events	130	130	15			20454

only a single two-word deletion. The second row for “DEL model” gives the perplexity based only on the word following a deletion, without including the probability for the deletion event itself. This shows that the context modification has the intended effect of making the next word more likely on average.

5.3.3 Filled pauses and utterance segmentation

As shown above, the Cleanup Model as applied to filled pauses yields a higher perplexity overall than the baseline trigram model. This is largely attributable to poorer word probability estimates at locations immediately following a filled pause. In prior work Shriberg [11] observed that filled pauses tend to occur at linguistic segment (e.g., clause) boundaries. Since the standard LM test utterances are segmented according to acoustic criteria, filled pauses around linguistic boundaries can actually occur in the middle of acoustic utterance segments. At such locations, the assumptions of the Cleanup Model would be grossly violated, since the preceding words actually belong to a different linguistic segment. The standard model, on the other hand, can produce reasonable predictions, as the filled pause can serve as an indicator of the boundary.

To test this hypothesis we compared the perplexities of both models on a subset of the test data that was hand-annotated for linguistic segmentations, and that had been re-segmented accordingly (10250 words in 1325 segments). Specifically, we compared the perplexities of words following *medial* filled pauses, i.e., filled pauses not occurring as the first or last word in a linguistic segment. Results are shown in Table 12.

Table 12: Local perplexities after medial filled pauses

Position	UH+1	UM+1
Baseline	849.0	437.4
FP model	606.2	361.7

We see that the Cleanup Model is the better predictor for words following medial FPs, the reverse of the result for acoustically segmented utterances. That is, the cleanup assumption holds for medial FPs if one models utterances based on linguistic, rather than acoustic, segments.

5.3.4 Results from related work

We are aware of two other groups of researchers currently investigating similar approaches to DF language modeling. In [6] a cleanup-style model for filled pauses is described. Ries and Qui at CMU [10] have experimented with models for repetitions and certain types of sentence deletion that incorporate the cleanup assumption. Overall, their results are consistent with ours (higher perplexity for filled pauses, lower perplexity for repetitions and deletions), but the overall effects are small, as in our case.

5.4 Discussion and Conclusions

The preceding analysis shows that a disfluency model based on the intuition underlying the Cleanup Model can yield only very small improvements in model perplexity, although the cleanup assumption seems to be valid on the Switchboard data we used in our experiments. The local perplexity analysis we performed shows

that the word positions at and immediately following DF events can be predicted with sometimes significantly lower perplexity, although the effect on overall perplexity is very small, due to the low frequency of DF events.

An interesting (and *prima facie* unexpected) result was that the Cleanup Model does not lower perplexity for filled pauses in acoustically segmented utterances. We attribute this to the particular way that the cleanup assumption is violated by filled pauses at linguistic segment boundaries internal to an acoustic segment. There are correlations between segmentation and other types of DFs, too, but the effects on the LM should be smaller in those cases as the bigram contexts for following words are not as radically changed by different segmentations. Our findings highlight the need for a more careful modeling (possibly with automatic recovery) of linguistic structures in conversational speech, a topic we plan to address in future work.

However, even for repetitions and deletions, it does not follow that recognition accuracy would necessarily improve with better local perplexities. In fact, we tested a trigram DF model (modeling only REP and DEL events) against a standard trigram on a Switchboard test set of 1192 segments, and found virtually no difference in overall word error rate (49.5% in both cases). This can be attributed to a number of factors. First, the REP/DEL model affects only a small portion of the total corpus (less than two cases per 100 words). Second, its advantage in modeling REP/DEL contexts should rarely come into effect due to the high error rate on adjacent words.

There are other reasons why lower perplexity may not lead to reduced word error rate. For instance, it could be that DFs tend to involve words of high frequency for which good acoustic models exist, so that a slightly improved LM would not affect recognition accuracy.

The overall conclusion is that by DF modeling at the LM level, contrary to high hopes in parts of the LM community, one should not expect a significant improvement in terms of word recognition performance. The main reason is that DFs are inherently local phenomena that are modeled surprisingly well by standard N-grams, even without context “cleanup.”

On the positive side, our results confirm that DFs have a systematic, nonrandom distribution that can be partly captured even with simple N-gram-like models; it is therefore conceivable that more sophisticated approaches could reap benefit for recognition accuracy.

One potential source of improved DF modeling are correlations with speaker identity. For example, [1] found that speakers can be grouped into those preferring deletions over repetitions (‘deleters’), and those with the opposite tendency (‘repeaters’). Such cross-utterance effects could be modeled in the LM using standard techniques, e.g., using adaptive interpolation of specialized models.

Finally, we note that the language modeling techniques described could also be used for automatic disfluency tagging and removal. Given a sequence of words and a probabilistic DF model of the type used here, one can use a Viterbi-style backtrace to recover the most likely sequence of DF events underlying the words sequence. This is another application we plan to study in the future.

Even though the results from disfluency modeling are largely negative as far as LM performance is concerned, we feel that an important methodological lesson has been learned. One should try to analyze the phenomenon at hand to the point where one can understand if and how it can improve on the baseline model. The method of local perplexity comparisons employed here is therefore likely to be of more use in the future.

References

- [1] J. Bear, J. Dowding, and E. Shriberg. Integrating multiple knowledge sources for detection and correction of repairs in human-computer dialog. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 56–63, University of Delaware, Newark, Delaware, June/July 1992.
- [2] I. Csiszar. A geometric interpretation of Darroch and Ratcliff’s generalized iterative scaling. *Ann. Stat.*, 17(3):1409–1413, 1989.
- [3] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Ann. Math and Stat.*, 43(5):1470–1480, 1972.

- [4] J. J. Godfrey, E. C. Holliman, and J. McDaniel. SWITCHBOARD: Telephone speech corpus for research and development. In *Proceedings IEEE Conference on Acoustics, Speech and Signal Processing*, volume I, pages 517–520, San Francisco, March 1992.
- [5] Peter Heeman and James Allen. Detecting and correcting speech repairs. In *Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics*, pages 295–302, New Mexico State University, Las Cruces, NM, June 1994.
- [6] R. Isotani and S. Matsunaga. A study of handling filled pauses in statistical language modeling. In *Proc. Acoustical Society of Japan*, pages 81–82, 1994. [In Japanese].
- [7] 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*, 35(3):400–401, March 1987.
- [8] C. H. Nakatani and J. Hirschberg. A corpus-based study of repair cues in spontaneous speech. *Journal of the Acoustical Society of America*, 95(3):1603–1616, 1994.
- [9] D. O’Shaughnessy. Correcting complex false starts in spontaneous speech. In *Proceedings IEEE Conference on Acoustics, Speech and Signal Processing*, volume I, pages 349–352, Adelaide, Australia, 1994.
- [10] K. Ries, 1995. Personal communication.
- [11] Elizabeth E. Shriberg. *Preliminaries to a Theory of Speech Disfluencies*. PhD thesis, Department of Psychology, University of California, Berkeley, CA, 1994.