

# Capturing Long Distance Dependency in Language Modeling: An Empirical Study

**Jianfeng Gao**

Microsoft Research, Asia  
49 Zhichun Road, Haidian District  
Beijing 100080  
jfgao@microsoft.com

**Hisami Suzuki**

Microsoft Research  
One Microsoft Way  
Redmond WA 98052  
hisamis@microsoft.com

## Abstract

This paper presents an extensive empirical study on two language modeling techniques, linguistically-motivated word skipping and predictive clustering, both of which are used in capturing long distance word dependencies that are beyond the scope of a word trigram model. We compare the techniques to others that were proposed previously for the same purpose. We evaluate the resulting models on the task of Japanese Kana-Kanji conversion. We show that the two techniques, while simple, outperform existing methods studied in this paper, and lead to language models that perform significantly better than a word trigram model. We also investigate how factors such as training corpus size and genre affect the performance of the models.

## 1 Introduction

Long distance word dependency is a critical problem in language modeling. Classical language models are based on the trigram assumption, i.e. the next word is predicted based on the two immediately preceding words. However, the words relevant to predicting the next word may lay in any position beyond the scope of a word trigram. There are two critical questions in incorporating long distance word dependencies in language modeling: (1) How do we find the relevant words, i.e., how do we define long distance dependencies? Are they defined syntactically, semantically, or in any other way? (2) How can these dependencies be represented in a sound probabilistic model?

In this paper, we first describe two techniques which were originally introduced in Gao et al. (2002b): the first is the method of skipping words in an  $n$ -gram language model in a linguistically

meaningful way; the second is the use of clustering techniques in language modeling. We then present an extensive empirical comparison of these techniques with others that have been proposed previously for the same purpose, including those described in Goodman (2001), Isotani and Matsunaga (1994), and Chelba et al. (1997). We re-implemented these methods (with some modifications as necessary) and generated language models using a large amount of Japanese newspaper corpus. We evaluated the resulting models on the task of Japanese Kana-Kanji conversion, which is a realistic application of a language model. While being relatively simple to implement, we show that the two techniques outperform existing methods studied in this paper, and lead to language models that are significantly better than a word trigram model. We also investigate how such factors as training corpus size and genre affect the performance of the language models.

## 2 Two Techniques

A trigram model predicts the next word  $w_i$  by estimating the conditional probability  $P(w_i|w_{i-2}w_{i-1})$ , where  $w_i$  is the word to be predicted, called the *predicted word*, and  $w_{i-2}$  and  $w_{i-1}$  are the context words used to predict  $w_i$ , called the *conditional words*.

A *skipping model* is an extension of a trigram model in that it predicts words based on two conditioning words that may not be adjacent to the predicted word (e.g., Rosenfeld, 1994; Ney et al., 1994; Siu and Ostendorf, 2000). *Function word skipping* is a linguistically-motivated skipping model created by differentiating function words  $F$  from content words or *headwords*  $H$ . For example, in *Mary has bought a book in the store*, *Mary*, *bought*, *book* and *store* count as headwords, while *has*, *a*, *in*, *the* count as function words. Function word skipping incorporates two as-

sumptions about language. First, we observe that headwords across phrase boundaries have dependency relations with each other. Therefore, we hypothesize (in the trigram context) that headwords may be conditioned not only by the two immediately preceding words, but also by two previous headwords. Second, we find that headword trigrams are also *permutable*, in the sense that they tend to capture order-neutral semantic dependency. For example, *Mary bought a book* and *the book Mary bought* can be expressed by the same headword trigram (*Mary~bought~book*) if we allow such permutations.

*Clustering techniques* attempt to make use of similarities between words to produce a better estimate of the probability of word strings (Goodman, 2001). In the context of computing the trigram probability  $P(w_3/w_1w_2)$ , either the predicted word  $w_3$  or the conditional words,  $w_1$  and  $w_2$ , can be clustered when building a cluster-based trigram model. Gao et al. (2002a) presents a thorough comparative study on various clustering models using the task of Japanese Kana-Kanji conversion, concluding that a model that uses clusters for predicted words, called the *predictive clustering model*, performed the best in most cases. In the current study, we used the predictive clustering technique in two different ways: first, we considered the distinction between two classes of words,  $H$  and  $F$ , as two pre-defined clusters. Secondly, we used the clustering technique to cluster similar words and headwords. We will now look at these in turn.

## 2.1 Permuted headword trigram model

Assuming that each word token can uniquely be classified as a headword or a function word, the permuted headword trigram model (PHTM) can be considered as a cluster-based language model with two clusters, headword  $H$  and function word  $F$ . We then define the conditional probability of  $w_i$  based on its history as the product of two factors: the probability of the category ( $H$  or  $F$ ), and the probability of  $w_i$  given its category. Let  $h_i$  or  $f_i$  be the actual headword or function word in a sentence, and let  $H_i$  or  $F_i$  be the category of the word  $w_i$ . The PHTM can then be formulated as follows:

$$P(w_i | \Phi(w_1 \dots w_{i-1})) = \quad (1)$$

$$P(H_i | \Phi(w_1 \dots w_{i-1})) \times P(w_i | \Phi(w_1 \dots w_{i-1})H_i)$$

$$+ P(F_i | \Phi(w_1 \dots w_{i-1})) \times P(w_i | \Phi(w_1 \dots w_{i-1})F_i)$$

where  $\Phi$  is a function that maps the word history ( $w_1 \dots w_{i-1}$ ) onto equivalence classes.  $P(H_i | \Phi(w_1 \dots w_{i-1}))$  and  $P(F_i | \Phi(w_1 \dots w_{i-1}))$  are the category probabilities, and  $P(w_i | \Phi(w_1 \dots w_{i-1})F_i)$  is the word probability given that the category of  $w_i$  is  $F$ . We used the unigram estimate for word category probabilities, i.e.,  $P(H_j | \Phi(w_1 \dots w_{i-1})) \approx P(H_j)$  and  $P(F_j | \Phi(w_1 \dots w_{i-1})) \approx P(F_j)$ .<sup>1</sup> We also used the standard trigram estimate for function word probability, i.e.,  $P(w_j | \Phi(w_1 \dots w_{i-1}), F_j) \approx P(w_j / w_{j-2}, w_{j-1}, F_j)$ , and approximated  $P(F_j) \times P(w_j / w_{j-2}, w_{j-1}, F_j)$  by  $P(w_j | w_{j-2}, w_{j-1})$ . The estimation of headword probability is slightly more elaborate, which incorporates the components for function word skipping and headword permutation:

$$P(w_i | \Phi(w_1 \dots w_{i-1})H_i) = \lambda_1(\lambda_2 P(w_i | h_{i-2}h_{i-1}H_i) \quad (2)$$

$$+ (1 - \lambda_2)P(w_i | h_{i-1}h_{i-2}H_i))$$

$$+ (1 - \lambda_1)P(w_i | w_{i-2}w_{i-1}H_i).$$

This estimate is an interpolated probability of three probabilities:  $P(w_i | h_{i-2}h_{i-1}H_i)$  and  $P(w_i | h_{i-1}h_{i-2}H_i)$ , which are the headword trigram probability with or without permutation, and  $P(w_i | w_{i-2}w_{i-1}H_i)$ , which is the probability of  $w_i$  given that it is a headword, where  $h_{i-1}$  and  $h_{i-2}$  denote the two preceding headwords, and  $\lambda_1, \lambda_2 \in [0, 1]$  are the interpolation weights optimized on held-out data. By separating the estimates for the probabilities of headwords and function words, we arrive at the final estimate below, where all probabilities are estimated using maximum likelihood estimation with Katz's (1987) backoff smoothing method to deal with the sparse data problem:

$$P(w_i | \Phi(w_1 \dots w_{i-1})) = \quad (3)$$

$$\begin{cases} \lambda_1(P(H_i | w_{i-2}w_{i-1}))(\lambda_2 P(w_i | h_{i-2}h_{i-1}) \\ + (1 - \lambda_2)P(w_i | h_{i-1}h_{i-2})) \\ + (1 - \lambda_1)P(w_i | w_{i-2}w_{i-1}) \quad w_i: \text{headword} \\ P(w_i | w_{i-2}w_{i-1}) \quad w_i: \text{function word} \end{cases}$$

## 2.2 Predictive clustering model

Let  $\bar{w}_i$  be the cluster which word  $w_i$  belongs to. In this study, we performed word clustering for words and headwords separately. As a result, we have the following two predictive clustering models, (4) for words and (5) for headwords:

<sup>1</sup> See Gao et al. (2002b) for a more detailed description of the model and its parameter estimation method.

$$P(w_i | w_{i-2}w_{i-1}) = P(\overline{w_i} | w_{i-2}w_{i-1}) \times P(w_i | w_{i-2}w_{i-1}\overline{w_i}) \quad (4)$$

$$P(w_i | h_{i-2}h_{i-1}) = P(\overline{w_i} | h_{i-2}h_{i-1}) \times P(w_i | h_{i-2}h_{i-1}\overline{w_i}) \quad (5)$$

$w_i$ : headword

We computed these models following the method described in Gao et al. (2002b). Substituting Equations (4) and (5) into Equation (3), we obtain the cluster-based PHTM, referred to as C-PHTM.

### 3 Main Results

We evaluated the language models on the task of Japanese Kana-Kanji conversion, which consists of converting a text string of syllabary-based Kana into an appropriate combination of Kanji and Kana. This is similar to the task of speech recognition, except that it does not include acoustic ambiguity. Performance on this task is measured in terms of the character error rate (CER), given by the number of characters wrongly converted from the phonetic string divided by the number of characters in the correct transcript.

For our experiments, we used two newspaper corpora, Nikkei and Yomiuri Newspapers, both of which have been pre-word-segmented. We built language models from a 36-million-word subset of the Nikkei Newspaper corpus, performed parameter optimization on a 100,000-word subset of the Yomiuri Newspaper (held-out data), and tested our models on another 100,000-word subset of the Yomiuri Newspaper corpus. The lexicon we used contains 167,107 entries.

Our evaluation was done using the N-best rescoring method (N=100), in which a list of hypotheses is generated by the baseline language model (a word trigram model in this study), which is then rescored using a more sophisticated language model. The main results are shown in Table 1, where the "oracle" result indicates the upper bound on performance. We see that both PHTM and C-PHTM outperform the baseline word trigram model, and the improvements are statistically significant according to the  $t$  test ( $p < 0.01$ ).

### 4 Comparative Study on Capturing Long Distance Word Dependency

Approaches to incorporating long distance word dependency in language modeling can be classified along the scale of how much linguistic structure they use. On one edge of the scale, there are higher order  $n$ -gram models and skipping

models, which use no or very little linguistic information; on the other end of the spectrum, we have models that use sophisticated syntactic structure, such as dependency-based models (e.g., Collins, 1996; Chelba et al., 1997; Gao and Suzuki, 2003) and constituency-based models (e.g., Chelba and Jelinek, 2000; Charniak, 2001; Roark, 2001). PHTM described in the previous sections fall between the two in the complexity of the linguistic structure it uses; in particular, it is similar to the models proposed by Isotani and Masunaga (1996), but differs from their models in crucial details. In this section, we provide an extensive empirical comparison of PHTM with some of the previous approaches which we re-implemented and compared on the task of Japanese Kana-Kanji conversion.

#### 4.1 Comparison with higher-order $n$ -gram models

Goodman (2001) showed that with a training corpus consisting of 280 million words and using Kneser-Ney smoothing (Kneser and Ney, 1995), small improvements were observed even into 6-grams, suggesting the usefulness of larger context.

The PHTM can be thought of as a variation of a higher order  $n$ -gram model, in that the headword trigrams capture longer distance dependencies than trigram models. In order to see how far the dependency goes within our headword trigram models, we plotted the distribution of headword trigrams (y-axis) against the  $n$  of the word  $n$ -gram if it were to be captured by the word  $n$ -gram (x-axis) in Figure 1. For example, given a word sequence  $w_1w_2w_3w_4w_5w_6$ , and if  $w_1$ ,  $w_3$  and  $w_6$  are headwords, then the headword trigram  $P(w_6|w_3w_1)$  spans the same distance as the word 6-gram model.

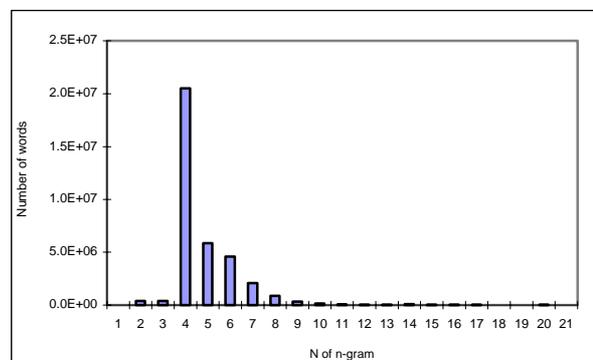


Figure 1. Distribution of headword trigrams against the  $n$  of word  $n$ -gram

From Figure 1, we can observe that approximately 95% of the headword trigrams can be captured by the higher-order  $n$ -gram model with the value of  $n$  smaller than 7. Based on this observation, we built word  $n$ -gram models with the values of  $n=4, 5$  and 6, using the training data described in Section 3. For all  $n$ -gram models, we used the interpolated modified absolute discount smoothing method (Gao et al., 2001), which is a modified version of Kneser-Ney smoothing and achieved the best CER results in our experiments. The CER results using higher-order  $n$ -gram models are presented in the third row of Table 1. They show that the performance of the higher order word  $n$ -gram models becomes quickly saturated as  $n$  grows: the best performance was achieved by the word 5-gram model, with the CER reduction of only 0.5% against the baseline trigram model. We increased the training data size to 180 million words, but obtained similar results. Following Goodman (2001), we suspect that the poor performance of these models is attributed to the data sparseness problem. From these results, we conclude that simply extending the context window by brute-force can achieve little improvement; in contrast, the use of even the most modest form of structural information such as the identification of headwords and automatic clustering can improve the performance.

## 4.2 Comparison with ATR models

Isotani and Matsunaga (1994) proposed two models that are based on the asymmetry between content and function words: the first model, referred to as ATR-I below, is an extension of word  $n$ -gram model: it is based on the assumption that the probability of each word in a sentence is determined by the preceding content and function word pair. Let  $f_i$  and  $c_i$  denote the last function word and the last content word in the substring  $w_1, \dots, w_{i-1}$ , relative to the word being predicted  $w_i$ . ATR-I can then be written as follows:

$$P(w_i) = \begin{cases} P(w_i | f_{i-1} w_{i-1}) & w_{i-1} : \text{content word} \\ P(w_i | c_{i-1} w_{i-1}) & w_{i-1} : \text{function word} \end{cases} \quad (6)$$

Isotani and Matsunaga used ATR-I alone without interpolating with another model, while Geutner (1996) interpolated ATR-I with a conventional word trigram model.

The second model, referred to here as ATR-II, is an approximation of ATR-I in that it has fewer parameters than ATR-I. It assumes that the

probability of each word is determined by the word immediately before it, and also by the preceding word of the same category (function word or content word) if the category of the word immediately before it is different from that of the word being predicted. In other words, ATR-II is of the form:

$$P(w_i) = \begin{cases} P(w_i | w_{i-1}) \times \frac{P(f_i | f_{i-1})}{P(f_i)} & w_{i-1} : \text{content word, } w_i : \text{function word (= } f_i) \\ P(w_i | w_{i-1}) \times \frac{P(c_i | c_{i-1})}{P(c_i)} & w_{i-1} : \text{function word, } w_i : \text{content word (= } c_i) \\ P(w_i | w_{i-1}) & \text{otherwise} \end{cases} \quad (7)$$

We built these models using the training data described in Section 3, and performed experiments using (a) ATR-I, (b) a combined model where ATR-I was interpolated with a word trigram model, and (c) ATR-II. The results are presented in the fourth row of Table 1. Our results are consistent with those presented in Isotani and Matsunaga (1994) and Geutner (1996). The CER using the ATR-I model alone is much higher than that of the baseline model. When interpolated with a word trigram model, the CER improves by 1.6%. Isotani and Matsunaga (1994) did not report error rate results of ATR-II; they only presented its perplexity, which was slightly higher than that of ATR-I. In contrast, our experiments show that ATR-II achieves lower CER than those either using ATR-I alone or using the combined model. We have two explanations for this: first, perplexity might not be a good criterion for evaluating combined language models. Second, ATR-II is a good approximation of ATR-I for practical purposes: both ATR models used the same set of information (i.e., the preceding content and function word pair) for word prediction, but ATR-II has the advantage of having many fewer parameters. Therefore, when the training data is not large enough, ATR-II might achieve better performance.

One significant difference between the ATR models and our own is that our models use separate probability estimates for headwords and function words as shown in Equation (3), which allows us to apply predictive clustering separately for these two submodels. In contrast, ATR models do not cluster words: the word categories are used only for the sake of finding the content and function word pair. In that sense, ATR models are

conceptually more similar to skipping models (e.g., Rosenfeld, 1994; Ney et al., 1994; Siu and Ostendorf, 2000), where only one probability estimate is applied for both content and function words. From the superior performance of PHTM to the ATR models, we believe that predictive clustering is a simple and effective technique in language modeling, and that the probability estimates for headword and function words should be done separately, as they play different semantic and syntactic roles in a sentence.

### 4.3 Comparison with dependency language models

Chelba et al. (1997) presents dependency language models (DLM) that capture linguistic constraints via a dependency structure, i.e., a set of probabilistic dependencies that express the relations between words in a sentence by an acyclic, planar, directed graph. Let  $W$  be a sentence, and  $D$  be its dependency structure. The DLM, in principle, recovers the probability of a sentence  $P(W)$  over all possible  $D$  given  $W$  by estimating the joint probability  $P(W, D)$ :  $P(W) = \sum_D P(W, D)$ . In practice, they introduce two approximations to make the model feasible. First, they take  $P(W) = \sum_D P(W, D) \approx P(W, D^*)$ , where  $D^*$  is the most probable dependency structure of the sentence, which is discovered by maximizing  $P(W, D)$ :  $D^* = \operatorname{argmax} P(W, D)$ . Below we simply use  $D$  to represent  $D^*$ . Second, they take  $P(W, D) = P(W|D)P(D) \approx P(W|D)P(D|W)$ , because it is very difficult to compute  $P(D)$  directly, while  $P(D|W)$  can be estimated through the parsing processing of generating  $D$ . Since this decomposition is statistically unwarranted, they only report word error rate results on the application of speech recognition: the best DLM achieves very small (less than 1%) yet statistically significant improvement ( $p < 0.02$ ) over a bigram model.

The DLMs we implemented for our experiments are different from Chelba et al. (1997) in the following three aspects. First, we simply take  $P(W) \approx P(W|D)$ . We then decompose the value of  $P(W|D)$  as the product of individual word probabilities:  $P(W|D) = \prod_{i=1..n} P(w_i|\Phi(w_1..w_{i-1}, D))$ . Here,  $D$  is determined using a modified version of Yuret's (1998) dependency parser.<sup>2</sup> The resulting

$D$  expresses the relations between headwords of each phrase in a sentence by an acyclic, planar, undirected graph where each headword (except the leftmost one) has exactly one related headword to its left. This is similar to link grammar described in Della Pietra et al. (1994).

Second, similar to PHTM, we differentiate the estimates for headword and function word probabilities (Equation (1)). We also used the unigram estimate for word category probabilities, and the standard trigram estimate for function word probability. For headword estimation, we chose a mapping function  $\Phi$  that retains (a) two preceding words  $w_{j-1}$  and  $w_{j-2}$ , (b) two preceding headwords  $h_{j-1}$  and  $h_{j-2}$ , and (c) one linguistically related word  $w_d$  according to  $D$ . The final estimate is given below, where  $\lambda_1, \lambda_2 \in [0,1]$  are the interpolation weights optimized on held-out data.

$$P(w_i|\Phi(w_1..w_{i-1}, D)) = \begin{cases} \lambda_1(P(H_i)(\lambda_2 P(w_i | h_{i-2}h_{i-1}) \\ + (1 - \lambda_2)P(w_i | w_d, (i, d) \in D)) \\ + (1 - \lambda_1)P(w_i | w_{i-2}, w_{i-1}) \\ \quad w_i: \text{headword} \\ \\ P(w_i | w_{i-2}, w_{i-1}) \\ \quad w_i: \text{function word} \end{cases} \quad (8)$$

Third, since a large Japanese training corpus annotated with  $D$  is not available, we used an unsupervised learning method that discovers  $D$  of a given sentence using a Viterbi iterative training procedure described in Gao and Suzuki (2003). It consists of three steps: (a) we assumed that each headword pair within a headword trigram constitutes an initial dependency, e.g., given a headword trigram  $(h_1, h_2, h_3)$ , there are 3 initial dependencies:  $d_{12}$ ,  $d_{13}$ , and  $d_{23}$ . From the initial dependencies, we computed an initial dependency parsing model similar to Collins (1996). (b) Given the parsing model, we used the Yuret's parser described above to select the most probable dependency structure for each sentence in the training data. This provides an updated set of dependencies. (c) We then re-estimated the parsing model parameters based on the updated dependency set. Steps (b) and (c) are iterated until the improvement in the probability of training data is less than a threshold.

The results are shown in the last row of Table 1. We can see that although DLM-1 (=DLM without

<sup>2</sup> The parser reads a sentence from left to right; after reading each new word  $w_j$ , it tries to link  $w_j$  to each of its previous words  $w_i$ , and push the generated dependency  $d_{ij}$  into a stack. When a dependency crossing or a cycle is detected in the stack, the dependency with the lowest dependency probability in conflict is

eliminated. We adopted this parser for detecting  $D$  in both testing and unsupervised training for its operating speed ( $O(n^2)$ ) and reasonably good accuracy (see Yuret (1998) for detail).

using the headword trigram submodel) outperformed the baseline model by 6.4% in CER reduction, the result is slightly (but significantly according to *t*-test) worse than that of HTM. This can be explained by the fact that the headword-based component of DLM-1 is bigram-based, while it is trigram-based in HTM. In DLM-2, we linearly interpolated all three probabilities in Equation (8), and obtained a statistically significant ( $p < 0.01$ ) improvement over HTM. The performance of DLM-2 (10.7% CER reduction) is quite similar to that of PHTM (10.5% CER reduction), which indicates an overlap in the data used in capturing long distance dependency: in fact, when we plotted the distribution of linguistically related word ( $w_d$  in Equation (8)) against the  $n$  of the headword  $n$ -gram, we discovered that about 83% of dependencies captured in DLM was also captured by the headword trigram model. The limited gain by incorporating the syntax-based dependency model might also be explained by the unreliable quality of the dependency annotation, which is generated in an unsupervised manner.

## 5 Varying Training and Test Corpora

Though it is very simple, PHTM achieves impressive performance in CER reduction over the comparison systems described in the previous section. However, the above experiments are all performed using newspaper text for both model training and testing, which is not a realistic scenario if we are to deploy the model in an application. In this section, we report the results of additional experiments in which PHTM is trained and tested on a much larger quantity and variety of text, in order to see the effect of data size and corpus genre on the model performance.

The basic process of model building was the same as described in Section 3, except for two settings: (a) we built a PHTM using the same lexicon but a much larger data set (328 million words) consisting of a diverse range of text including newspapers, novels, business letters, chat logs and encyclopedia articles, and tested the model on various text genres; (b) we pruned (but did not compress) the resulting model size into 60MB for the word trigram model and 80MB for the headword trigram model.<sup>3</sup> The results are shown in Table 2.

<sup>3</sup> The model sizes of the experiments described in Section 3 are 107MB for the word trigram model and 139MB for the headword trigram model, respectively.

As is observed from Table 2, there is a significant CER reduction on the Yomiuri newspaper corpus (16.6%); this is attributed to the fact that nearly 80% of the training data consists of Nikkei newspaper text. We can therefore say that when the genre of the test and training corpora matches, PHTM is extremely effective in terms of CER reduction. Note also that the use of a much larger training data set favorably affected the CER reduction in the newspaper domain: the CER reduction of PHTM went up from 10.5% to 16.6% on the Yomiuri newspaper corpus by using a larger training data set. The CER reduction, however, is only moderate in other genres. These results point to the observation that there is a serious data sparseness problem in PHTM: for the majority of cases, the relevant data is not present in the model, particularly when the test corpus is in a different genre from that of the training data.

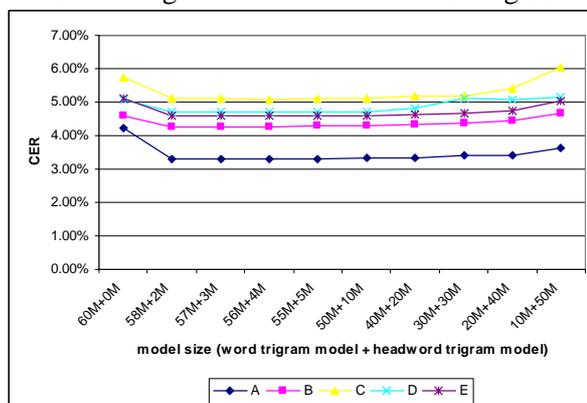


Figure 2. CER at various model size

However, there seems to be one interesting property of the data sparseness in PHTM that can be exploited for practical purposes. Figure 2 plots the CER reduction of PHTM models at various size combinations of word and headword trigram models, keeping the overall model size constant at 60MB. A-E in the figure corresponds to the test corpora described in Table 2. We can see that across all test corpora, the best performance is achieved by the model with 58MB of the word trigram model and 2MB of the headword trigram model. This indicates that the relevant and useful data for the purposes of language modeling is concentrated in the very small, high frequency portion of the data, and not distributed evenly over the model. This property of the headword trigram model allows it to be effective at an extremely small size, a desirable result for practical purposes.

Models were pruned using the techniques described in Stolcke (1998).

## 6 Conclusion

Linguistically-informed word skipping and predictive clustering are two simple techniques that work surprisingly well in language modeling. Using these techniques, our models achieved up to 15% CER reduction over a conventional word trigram model in a Japanese Kana-Kanji conversion task. Our models can effectively capture long distance dependencies among headwords up to the range of word 6-grams, while performing much better than a word 6-gram model. We attribute the success of our models to the use of linguistic structure in the form of headword identification and clustering, and to the fact that our models have many fewer parameters to train than higher-order  $n$ -gram models.

In the comparison with ATR models, we also showed that by clustering predictive words into head words and function words, we can optimize the probability estimates for these classes separately, leading to a superior performance of the proposed model.

We have also investigated the use of syntactic information in language modeling. One future challenge in this area is to find syntactically motivated dependencies that do not overlap with the dependencies that are already captured by headword-based  $n$ -gram models.

## Acknowledgments

We would like to thank Noriko Ishibashi and Yang Wen for their help with our experiments.

## References

- Charniak, Eugene. 2001. Immediate-head parsing for language models. In *ACL/EACL 2001*, pp.124-131.
- Chelba, Ciprian and Frederick Jelinek. 2000. Structured language modeling. *Computer Speech and Language*, Vol. 14, No. 4. pp 283-332.
- Chelba, C, D. Engle, F. Jelinek, V. Jimenez, S. Khudanpur, L. Mangu, H. Printz, E. S. Ristad, R. Rosenfeld, A. Stolcke and D. Wu. 1997. Structure and performance of a dependency language model. In *Processing of Eurospeech*, Vol. 5, pp 2775-2778.
- Collins, Michael John. 1996. A new statistical parser based on bigram lexical dependencies. In *ACL-34*: 184-191.
- Della Pietra, S., V. Della Pietra, J. Gillett, J. Lafferty, H. Printz and L. Ures. 1994. Inference and estimation of a long-range trigram model. Technical report CMU-CS- 94-188, Department of Computer Science, CMU.
- Gao, Jianfeng, Joshua Goodman and Jiangbo Miao. 2001. The use of clustering techniques for language model – application to Asian language. *Computational Linguistics and Chinese Language Processing*. Vol. 6, No. 1, pp.27-60.
- Gao, Jianfeng, Joshua Goodman, Guihong Cao and Hang Li. 2002a. Exploring asymmetric clustering for statistical language modeling. In *ACL 2002*, pp.183-190.
- Gao, Jianfeng, Hisami Suzuki and Yang Wen. 2002b. Exploiting headword dependency and predictive clustering for language modeling. In *EMNLP 2002*, pp.248-256.
- Gao, Jianfeng and Hisami Suzuki. 2003. Unsupervised learning of dependency structure for language modeling. In *ACL 2003*, pp.521-528.
- Goodman, Joshua. 2001. A bit of progress in language modeling. *Computer Speech and Language*. October, 2001, pp 403-434.
- Geutner, Petra. 1996. Introducing linguistic constraints into statistical language modeling. In *ICSLP96*, pp.402-405.
- Katz, S. M. 1987. Estimation of probabilities from sparse data for other language component of a speech recognizer. *IEEE transactions on Acoustics, Speech and Signal Processing*, 35(3): 400-401.
- Isotani, R. and Matsunaga, S. 1994. A stochastic language model for speech recognition integrating local and global constraints. In *ICASSP-94*, pp. 5-8.
- Kneser, R and H. Ney. 1995. Improved backing-off for  $m$ -gram language modeling. In *ICASSP'95*: 181-184.
- Ney, Hermann, Ute Essen and Reinhard Kneser. 1994. On structuring probabilistic dependences in stochastic language modeling. *Computer Speech and Language*, 8: 1-38.
- Roark, Brian. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 17-2: 1-28.
- Rosenfeld, Ronald. 1994. Adaptive statistical language modeling: a maximum entropy approach. Ph.D. thesis, Carnegie Mellon University.
- Siu, Manhung, and Ostendorf, Mari. 2000. Variable  $n$ -grams and extensions for conversational speech language modeling. *IEEE Transactions on Speech and Audio Processing*, 8: 63-75.
- Stolcke, A. 1998. Entropy-based pruning of backoff language models. In *Proceeding of DARPA News Transcription and Understanding Workshop*, Lansdowne, VA, pp.270-274.
- Yuret, Deniz. 1998. Discovery of linguistic relations using lexical attraction. Ph.D. thesis, MIT.

Model	Description	CER %	CER Reduction
Baseline	Word trigram model	3.73	---
Oracle	In the 100-best list with the minimum number of errors	1.51	59.5%
HTM	Equation (3) with $\lambda_1=0.2$ and $\lambda_2=1$	3.41	8.6%
PHTM	Equation (3) with $\lambda_1=0.2$ and $\lambda_2=0.7$	3.34	10.5%
C-PHTM	Equation (3) with $\lambda_1=0.3$ and $\lambda_2=0.7$	3.17	15.0%
4-gram	Higher-order $n$ -gram model with a modified version of Kneser-Ney interpolation smoothing	3.71	0.5%
5-gram		3.71	0.5%
6-gram		3.73	0.1%
ATR-I	Equation (6)	4.75	-27.3%
ATR-I +	ATR-I interpolated with Baseline	3.67	1.6%
ATR-II	Equation (7)	3.65	2.1%
DLM-1	Equation (8) with $\lambda_1=0.1$ and $\lambda_2=0$	3.49	6.4%
DLM-2	Equation (8) with $\lambda_1=0.3$ and $\lambda_2=0.7$	3.33	10.7%

**Table 1.** Comparison of CER results

corpus	CER of word trigram baseline (%)	CER of PHTM (Equation (3) with $\lambda_1=0.2$ and $\lambda_2=0.7$ ) (%)	CER reduction
A. Yomiuri newspaper	4.40	3.67	16.6%
B. Company newsletter	4.59	4.51	1.7%
C. Business letters	5.77	5.52	4.3%
D. Science essays	5.08	4.98	2.0%
E. Current topics	4.96	4.99	-0.6%

**Table 2.** CER results on various test corpora