Lattice based language models

Pierre Dupont[†] and Ronald Rosenfeld

September 1997

CMU-CS-97-173

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This paper introduces lattice based language models, a new language modeling paradigm. These models construct multi-dimensional hierarchies of partitions and select the most promising partitions to generate the estimated distributions. We discussed a specific two dimensional lattice and propose two primary features to measure the usefulness of each node: the training-set history count and the smoothed entropy of its prediction. Smoothing techniques are reviewed and a generalization of the conventional backoff strategy to multiple dimensions is proposed. Preliminary experimental results are obtained on the SWITCHBOARD corpus which lead to a 6.5 % perplexity reduction over a word trigram model.

Project sponsored by the National Security Agency under Grant No. MDA904-97-1-0006. The United States Government is authorized to reproduce and distribute reprints notwithstanding any copyright notation hereon.

[†]Current address: Dépt. Math., Université Jean Monnet, 23, rue P. Michelon, 42023 Saint-Etienne cedex, France, E-mail: `pdupont@univ-st-etienne.fr`

Keywords: speech recognition, statistical language modeling, lattice based models, smoothing techniques

1 Introduction

Statistical language modeling is concerned with estimating the probability of various linguistic events, using large samples of language data. As used in automatic speech recognition, statistical language models typically estimate $Pr(w|h)$ – the conditional distribution of the identity of the next word in a sentence or document, given the current history (namely the identity of the words that occurred up to this point).

The most common statistical language model is the N-gram, which makes the simplifying assumption:

$$\Pr(w|h) \triangleq \Pr(w_i|w_1, w_2, \ldots, w_{i-1}) \approx \Pr(w_i|w_{i-N+1}, \ldots, w_{i-1})$$

N-gram models have dominated statistical language modeling ever since their introduction in the 1970's [1]. In spite of their apparent limitations, N-gram models proved simple, robust, and surprisingly hard to improve on ([2]).

Within the N-gram paradigm, much work was done on smoothing, word clustering and adaptation. In smoothing, the dominant ideas are those of discouting ([3, 4]), and backing off to ([5]), or linear interpolation with ([1]), lower order models. In clustering words, most algorithms use iterative methods that greedily attempt to minimize local information theoretic measures ([6, 7]). N-gram based adaptation work consists primarily of variations on interpolating the static model with small N-grams built from more pertinent data (or from the document's history, i.e. a cache) ([8, 9, 10, 11]).

During the last decade, several attempts were made to break away from the N-gram paradigm. These include decision trees and maximum entropy models.

Decision tree language models ([12]) are the ultimate in partition based modeling, because they can implement arbitrary partitions. But this richness is also the source of their main weakness, which is the computational intractability of finding the optimal tree. This leads to greedy searches and other algorithmic and modeling compromises, which affect the quality of the resulting model. As a consequence, decision trees have not yet succeeded in significantly improving on the baseline N-gram model.

Another problem with decision trees is data fragmentation. Once a tree has been constructed, each history fits into exactly one leaf, and the resulting estimation is based only on the training data that belong on the path from the root to that leaf. No use is made of training data which may be intimately related to the current situation but which happens to diverge early on into other paths due to orthogonal questions higher up in the tree.

The ability to combine evidence based on diverse and partially overlapping features was the main motivation behind the introduction of the maximum entropy paradigm to language modeling ([13, 14, 15, 16, 17]). As was demonstrated in ([15]), maximum entropy models can successfully integrate diverse knowledge sources in a unified and consistent statistical framework, and can result in significant improvement over the existing state-of-the-art N-gram based techniques. However, as was also discussed in ([15]), training maximum entropy models is computationally very demanding, which renders them of little use when large amounts of data (e.g. a hundred millions words) are available.

This report discusses lattice based language models — an alternative language modeling paradigm which we have just started exploring. Like a decision tree, a lattice is based on a set of partitions of the history, and like an N-gram, the set of partitions is strongly constrained by word order and word ordinal position. But unlike a decision tree, estimates are constructed using multiple partitions which may or may not be refinements of each other. This allows multiple, partially overlapping knowledge sources to be incorporated, as in a maximum entropy model. But unlike the latter, training a lattice based model is not computationally demanding.

2 Outline

Classical N-gram models define a particular partitioning of the history space. For example a 3-gram model is defined as $P(w_i|h) \triangleq P(w_i|w_{i-2}w_{i-1})$ where all histories h sharing the same last two words are considered to be equivalent. Another way of partitioning the history space relies on word classes. For example a class 3-gram model[1], can be defined as $P(w_i|g_{i-2}g_{i-1})$ where the history is seen as a sequence of classes $(g_{i-2}g_{i-1})$. Similarly this class history can be made more coarse by clustering the original classes into superclasses. Thus, in addition to the *model order*, the definition of a *hierarchy of classes* allows for a natural extension to N-gram models in which the space of histories can be partitioned. This idea is described in section 3 where *lattices of N-gram models* are introduced. Analysis of this approach as used on the Switchboard data is given in section 4. The definition of lattices of

[1]Classes are introduced here only for defining various partitions of the history space. The resulting model, i.e. $p(w_i|g_{i-2}g_{i-1})$, contrasts with a more traditional class 3-gram where the prediction is made in two steps: $p(w_i|g_i)p(g_i|g_{i-2}g_{i-1})$.

N-gram models also suggests an extension to the backoff smoothing, called *two-dimensional backoff* which is detailed in section 5.

Linear combination of predictors is *static* in the sense that the interpolation weights remain fixed after their estimation. A backoff model can also be considered static since the most specific predictor is always used when available[2]. Here we investigate language models that *dynamically* choose among a *large* set of predictors. In other words the combination of various predictors depends on their estimated quality in a given context. These ideas are developed in section 6.

3 Lattice of N-gram models

3.1 Model definition

A lattice of N-gram models is shown in figure 1. In this particular example, 17 predictors are considered. They correspond to five model orders (from 5-gram on the left to unigram on the right) and a hierarchy of four class levels (the word level at the bottom and the coarsest class level at the top). The different class levels collapse to one predictor in the unigram case, in which all histories are mapped to one equivalence class.

The lattice structure represents a set of inclusion relations. In particular, the hierarchy of classes defines the following inclusions in the space of histories:

$$\{w_{i-1}\} \subseteq g_{i-1} \subseteq G_{i-1} \subseteq \ldots$$

or similarly

$$\{(w_{i-2}w_{i-1})\} \subseteq \{(g_{i-2}g_{i-1})\} \subseteq \{(G_{i-2}G_{i-1})\} \subseteq \ldots$$

The order of the model also defines the following inclusions in the space of histories

$$\{(w_{i-4}w_{i-3}w_{i-2}w_{i-1})\} \subseteq \{(w_{i-3}w_{i-2}w_{i-1})\} \subseteq \ldots \subseteq \{w_{i-1}\}$$

or similarly

$$\{(g_{i-4}g_{i-3}g_{i-2}g_{i-1})\} \subseteq \{(g_{i-3}g_{i-2}g_{i-1})\} \subseteq \ldots \subseteq \{g_{i-1}\}$$

[2] Even backoff models with cutoffs [18] are static as long as the cutoff values are fixed for all histories.

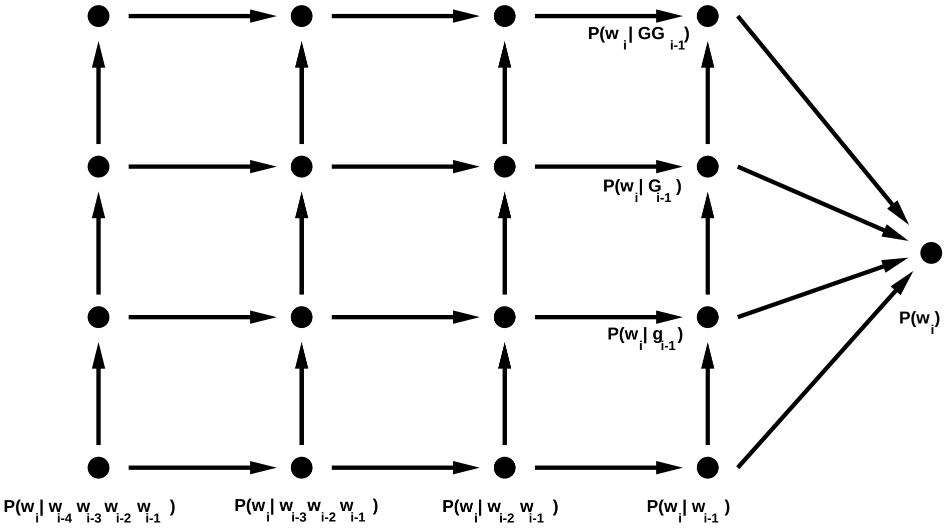

Figure 1: Lattice of language models

Each node of the lattice represents a particular predictor which, for a given history h and a given predicted word w, is associated with two counts $C(h, w)$ and $C(h)$. A maximum likelihood estimate[3] of the probability $P(w|h)$ is given by the ratio of these counts: $p(w|h) = \frac{C(h,w)}{C(h)}$. Thus apart from the important issue of smoothing the probability estimate is completely defined with two counts.

The lattice structure reflects the partial ordering between predictors together with the specificity of each. In our example, the 5-gram at the word level is the most specific predictor while the unigram is the least specific one. In other words, the lattice can also be seen as a DAG in which any path goes from a more specific to a less specific predictor. The traditional backoff model becomes a particular case in which the lattice is reduced to one dimension (associated with the model order) and backing off consists of moving towards a less specific predictor.

While the unigram is the least specific model, it is also the most reliable, as it is estimated with the largest amount of data. Our ultimate goal is to find the best combination of predictors that trades specificity off against reliability. More qualitative measures related to theses questions are defined in section 3.3.

Possible extensions to this model include more flexible (and numerous) history partitions, i.e. where every position in the history can come from

[3] In this paper an uppercase P denotes a true probability and a lowercase p denotes a probability estimate.

a different class level. Other dimensions can also be considered since each predictor can be estimated on a hierarchy of corpora or on a topic tree.

3.2 Independent predictors

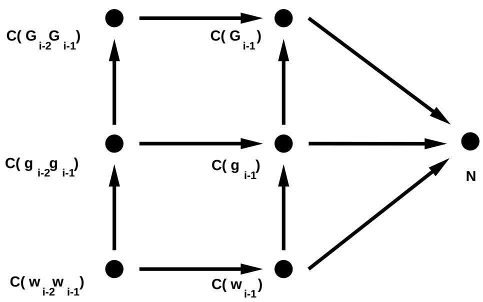

Figure 2: Lattice of history counts

The estimates represented in the lattice nodes as described so far are *dependent* predictors. For instance, the set of 3-gram histories represented in the lattice are a subset of the 2-gram histories. Under some circumstances, it is conceivable that we would want to make statistical measures on lattice nodes as if they were not dependent on each other. For instance, testing the hypothesis that adjacent nodes' distributions are identical would demand that comparison data would be drawn from independent distributions. For this reason we developed the following method for construction of *independent* estimators from the dependent estimators. While this construction is not used in any of the following experiments, we have included its derivation here for the sake of completeness.

Each predictor in the lattice is characterized, for a given history h and a given predicted word w, by two counts $C(h, w)$ and $C(h)$. Let us first consider the history counts at each node as represented in figure 2. N denotes the training set size, i.e. the unigram "history" count, and $C(w_{i-2}w_{i-1})$ denotes the history count for a particular word 3-gram. For simplicity, we assume here that the word trigram is the most specific predictor[4].

Along the horizontal dimension we can build independent predictors in the following way. The word bigram clearly depends on the word trigram since $\{(w_{i-2}w_{i-1})\} \subseteq \{w_{i-1}\}$ but one can compute an independent "bigram" by a difference of two sets. A corrected count $\hat{C}(w_{i-1})$ is obtained by subtracting the count $C(w_{i-2}w_{i-1})$ from the original bigram history count $C(w_{i-1})$:

[4]The proposed calculation can be extended trivially to lattices of any order.

$$\hat{C}(w_{i-1}) = C(w_{i-1}) - C(w_{i-2}w_{i-1}) \tag{1}$$

The corrected count can be written more accurately as $C(\overline{w_{i-2}}w_{i-1})$ where $\overline{w_{i-2}}$ denotes any word except w_{i-2}.

This subtraction operation can be applied to other counts in the same way. For instance, the corrected unigram count becomes $\hat{N}$:

$$\begin{aligned} \hat{N} &= N - C(w_{i-2}w_{i-1}) - C(w_{i-1}) + C\{(w_{i-2}w_{i-1}) \cap (w_{i-1})\} \\ &= N - C(w_{i-1}) \end{aligned} \tag{2}$$

In both cases, counts of independent events can be obtained by subtracting adjacent counts in the lattice. The same result applies to the vertical dimension since

$$\hat{C}(g_{i-2}g_{i-1}) = C(g_{i-2}g_{i-1}) - C(w_{i-2}w_{i-1}) \tag{3}$$

and

$$\begin{aligned} \hat{C}(G_{i-2}G_{i-1}) = \;& C(G_{i-2}G_{i-1}) - C(w_{i-2}w_{i-1}) \\ & -C(g_{i-2}g_{i-1}) + C\{(w_{i-2}w_{i-1}) \cap (g_{i-2}g_{i-1})\} \\ = \;& C(G_{i-2}G_{i-1}) - C(g_{i-2}g_{i-1}) \end{aligned} \tag{4}$$

A similar computation can be performed for predictors which have more than one predecessor in the lattice. Following the same reasoning as before, a corrected count $\hat{C}(g_{i-1})$ can be obtained from the original bigram history g_{i-1} by subtracting the two adjacent counts, $C(w_{i-1})$ and $C(g_{i-2}g_{i-1})$, and by adding the count of their intersection $C\{(w_{i-1}) \cap (g_{i-2}g_{i-1})\}$:

$$\begin{aligned} \hat{C}(g_{i-1}) = \;& C(g_{i-1}) - C(w_{i-1}) - C(g_{i-2}g_{i-1}) \\ & +C\{(w_{i-1}) \cap (g_{i-2}g_{i-1})\} \\ = \;& C(g_{i-1}) - C(w_{i-1}) - C(g_{i-2}g_{i-1}) \\ & +C(g_{i-2}w_{i-1}) \end{aligned} \tag{5}$$

Here an additional count, i.e. $C(g_{i-2}w_{i-1})$, must be collected for each estimate[5]

Finally, the same reasoning may be followed to produce corrected joint counts $\hat{C}(h, w)$ where the subtraction operations are parallel to the ones described above. The partitioning of the history space corresponds now to mutually exclusive instead of inclusive sets.

[5] The unigram predictor has also more than one predecessor in the lattice, but in this case the final corrected count is given by $\hat{N} = C(G_{i-1})$.

3.3 Entropy of smoothed distribution

As mentioned in section 3, we are interested in measuring both the reliability and specificity of each of the predictors in the lattice. We can assume that high count histories will be reliably estimated. Consequently we will use the history count $C(h)$ itself as our reliability measure. To capture the notion of specificity we would have to consider the distance between the distribution associated with a predictor and the true, but unknown, distribution $P(w|h)$. A related notion is the usefulness of the prediction which can be measured as the entropy of the history $H(h)$, that is the entropy of the estimated distribution $p(w|h)$ to predict any word w from a fixed history h:

$$H(h) = -\sum_w p(w|h) \log p(w|h) \tag{6}$$

In general, the estimate $p(w|h)$ must be smoothed since for many predictors there is so little data that the entropy estimate is highly unreliable. However, smoothing is usually performed by combining several predictors. In the proposed framework this would correspond to considering the same sequence of tokens as members of different history partition classes. Such a smoothing mechanism would no longer allow for measuring the entropy of a particular history, but rather of combined histories. A solution to this problem consists in smoothing by absolute discounting and backing off to the unigram distribution $p(w)$ as described in equation 7.

$$p(w|h) = \begin{cases} \frac{C(h,w)-d}{C(h)} & \text{if } C(h,w) > 0 \\ \alpha(h)p(w) & \text{otherwise} \end{cases} \tag{7}$$

where d denotes the discounting value (typically 0.5) and $\alpha(h)$ is a normalizing factor.

This simple smoothing technique combines only the original predictor with the unigram in such way that comparison with other predictors is meaningful. A high entropy value indicates a flat distribution for this particular history while a low entropy indicates a sharp distribution.

Another advantage of the proposed smoothing is the low cost of the entropy calculation[6]. Let H_1 denote the entropy of the unigram distribution, i.e. $H_1 = -\sum_w p(w) \log p(w)$, which needs to be computed only once. The entropy calculation can then be rewritten as in equation 8, in which all sum-

[6]The entropy calculation for the independent estimators described in section 3.2 would be much more costly than that described here. This is the main reason that the independent estimator formulation was not used in the experiments described in this paper.

mations are over the set of words w such that $C(h,w) > 0$. This set of words generally represents a very small fraction of the vocabulary.

$$
\begin{aligned}
H(h) \; &= \; -\sum_{w:C(h,w)>0} p(w|h)\log p(w|h) \\
&\quad\; -\sum_{w:C(h,w)=0} \alpha(h)p(w)\log(\alpha(h)p(w)) \\
&= \; -\sum_{w:C(h,w)>0} p(w|h)\log p(w|h) \\
&\quad +\alpha(h)[\;\; H_1 + \sum_{w:C(h,w)>0} p(w)\log p(w) \\
&\qquad \log\alpha(h)((\sum_{w:C(h,w)>0} p(w)) - 1)]
\end{aligned} \tag{8}
$$

where

$$
\alpha(h) = \frac{\sum_{w:C(h,w)>0} \frac{d}{C(h)}}{1 - \sum_{w:C(h,w)>0} p(w)}
$$

3.4 Hierarchical clustering

The definition of a lattice of N-gram models relies on the development of word classes which deterministically map a word w to a class $g(w)$. This mapping can be automatically constructed by a clustering algorithm such as the one proposed by Kneser and Ney [19]. Its objective is to find a mapping such that an associated class bigram model[7] has a locally minimal perplexity on the training data. This criterion can be shown to be equivalent to the local minimization of the loss of mutual information between words.

Ney's algorithm does not construct a hierarchical clustering since the number of classes is fixed a priori[8]. However, a hierarchy of classes can be obtained by relabeling the training data according to the estimated word-to-class mapping and by iterating the clustering with a smaller number of classes. Figure 3 shows a typical example of hierarchical clusters constructed from the Switchboard data, where the number of classes used was 1600, 300 and 50, respectively.

[7]For a class bigram model the probability of the next word is given by $p(w_i|g(w_i))p(g(w_i)|g(w_{i-1}))$

[8]Notice that the optimal number of classes for a single class bigram model can be estimated on a held-out set or by leaving-one-out.

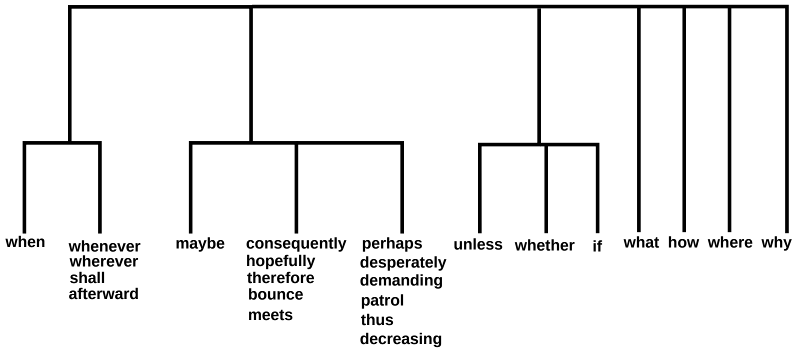

Figure 3: Hierarchical clustering example

4 Data Analysis

4.1 Switchboard data

The Switchboard data used in the present work consists of about 2.5 million words of transcribed conversational speech [20]. We chose a vocabulary of 9802 words corresponding to a coverage of 98.5 %. This vocabulary is closed as it contains the special token UNK to which any out of vocabulary word is mapped.

For the current experiments the data were randomly split into 3 sets. The first set forms the training data from which the counts are computed. The second set is a held-out set used for analysis and additional parameter estimation. Finally the test set is used for evaluating the perplexity of the proposed models. Table 1 summarizes the number of sentences and word tokens in these data sets. A hierarchy of classes was also built from the training data with respectively 1600, 300 and 50 classes as described in section 3.4.

Dataset	# sentences	# words
Training	140,807	2,365,741
Held-out	9,000	151,346
Test	2,568	39,956

Table 1: Switchboard data

4.2 History and prediction hit ratios

For each lattice node and for each word to be predicted in the held-out set, the two counts $C(h)$ and $C(h, w)$ can be computed on the training data. The

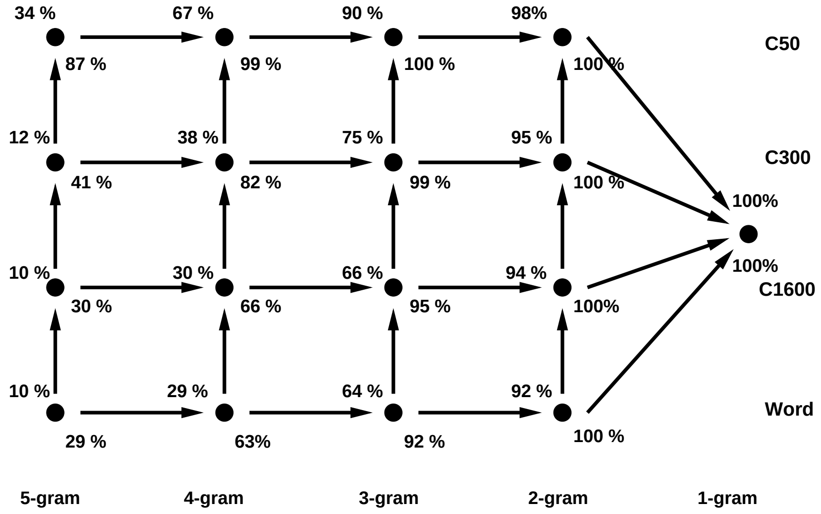

Figure 4: History and prediction hit ratios

history hit ratio is the fraction of the time a particular history, observed in the held-out set, was already seen in the training set (i.e. $C(h) > 0$). The *prediction hit ratio* is the fraction of the time the next word was already seen after that particular history in the training set (i.e. $C(h, w) > 0$).

Figure 4 gives the history and prediction hit ratios on the held-out set. For example, at the 1,600 class level (C1600) the 4-gram histories h were already observed 66 % of the time while the joint events (h, w) were already observed 30 % of the time[9].

In a typical backoff model a prediction miss occurs when a backoff to a lower order is required. Given that our reference model is a word 3-gram, it is interesting to see which of the other predictors might be used instead. Figure 5 gives the prediction hit ratios of all predictors when the word 3-gram model would back off to lower order predictors. On this data set, the word 2-grams could be used in 77 % of these cases and a 4-gram model with 50 classes could also be used 43 % of the time. This indicates that other predictors in the lattice could overcome a prediction miss of the reference model.

[9]Notice that the 4-gram history hit ratio is not necessarily equal to the 3-gram prediction hit ratio. This is due to sentence boundary effects.

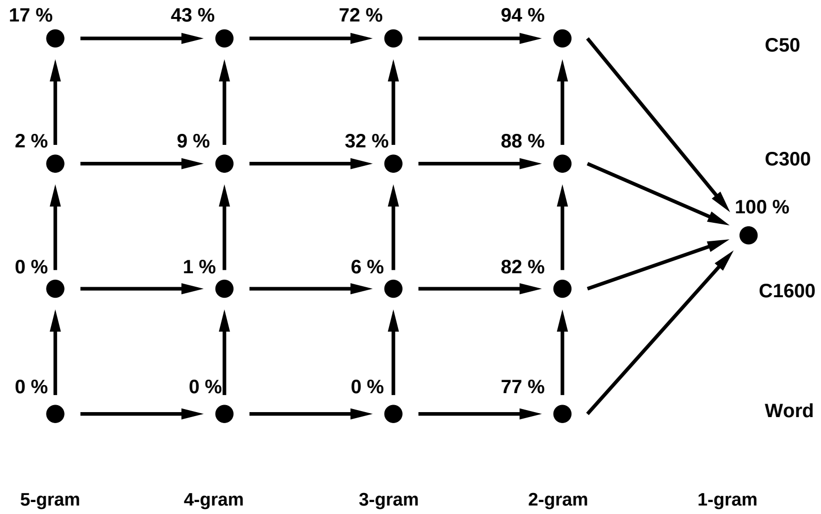

Figure 5: Prediction Hit Ratio when word trigram backs off

The analysis of the prediction hit ratio for each predictor is of little use in practice as the next word must be known in advance in order to determine whether a particular predictor would hit or miss it. However the count of the history is known before the prediction occurs. Figure 6 presents the relation between the count of the history $C(h)$ of any of the 16 predictors[10] in the lattice and the prediction hit. The data was gathered by pooling together datapoints from all 16 predictors applied to the same count. As it may be expected the prediction hit increases rapidly as the count of the history increases[11].

[10]The unigram model, absent from figure 6, has a 100 % prediction hit ratio since the vocabulary is closed.

[11]Notice the logarithmic scale along the horizontal axis of figure 6.

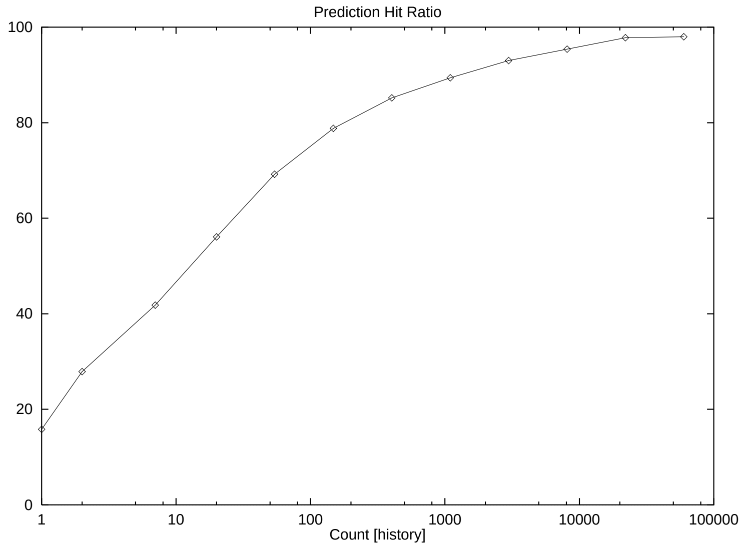

Figure 6: Prediction hit ratio as a function of history count

The previous analysis can be refined as follows. In figure 7, the prediction hit ratio is plotted as a function of the history count $C(h)$ and the entropy $H(h)$. In particular, over all predictors such that $C(h)$ falls between 100 and 10,000 the prediction hit will be relatively higher if the entropy is smaller. In other words, among all predictors falling in this count interval, one should prefer the most specific ones – the ones with the lowest entropy. It is interesting to note that 90% of the counts actually fall in this interval as can be concluded from the histogram of counts presented in figure 8.

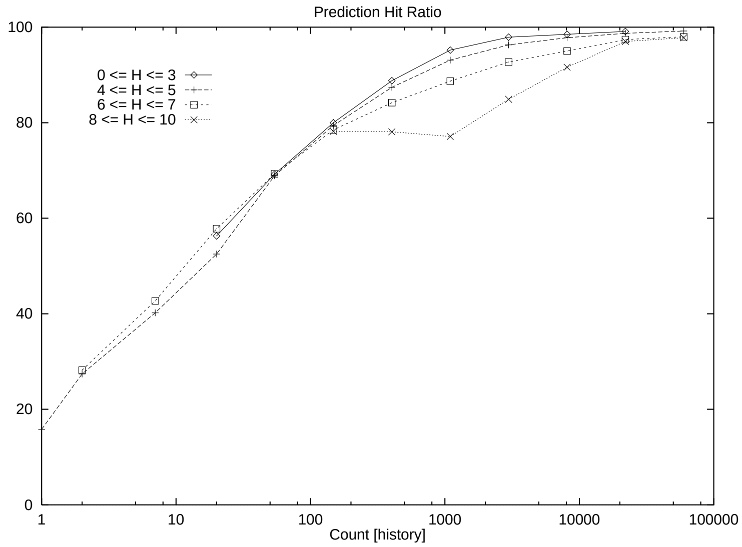

Figure 7: Prediction hit ratio as a function of history count and entropy

5 Smoothing techniques

Traditional backoff models combine several predictors to overcome the ever present data sparseness problem. We review in section 5.1 the details of backoff models and we show in section 5.2 how these techniques can be extended to lattices of N-gram models.

5.1 Backoff scheme

Equation 9 describes a particular backoff scheme where d_C denotes the discounted value subtracted from the counts of seen events. This discounted value may depend on the count $C(h,w)$ as in Turing-Good discounting [21] or may be constant as in the case of absolute discounting [22]. The normalized discounted probability mass is distributed to unseen events in proportion to their backoff estimates ($p_{back}(w|h)$). Here the backoff distribution $p_{back}(w|h)$ is only used if the higher order estimate cannot be used, that is when $C(h,w) = 0$. We will refer to this particular backoff scheme as *shadowing* since the higher order estimate shadows the backoff distribution.

$$p(w|h) = \begin{cases} \frac{C(h,w)-d_C}{C(h)} & \text{if } C(h,w) > 0 \\ \alpha(h) p_{back}(w|h) & \text{otherwise} \end{cases} \tag{9}$$

Equation 10 describes a different backoff scheme in which the backoff distribution is used in all cases and the normalization factor, here $\gamma(h)$, is

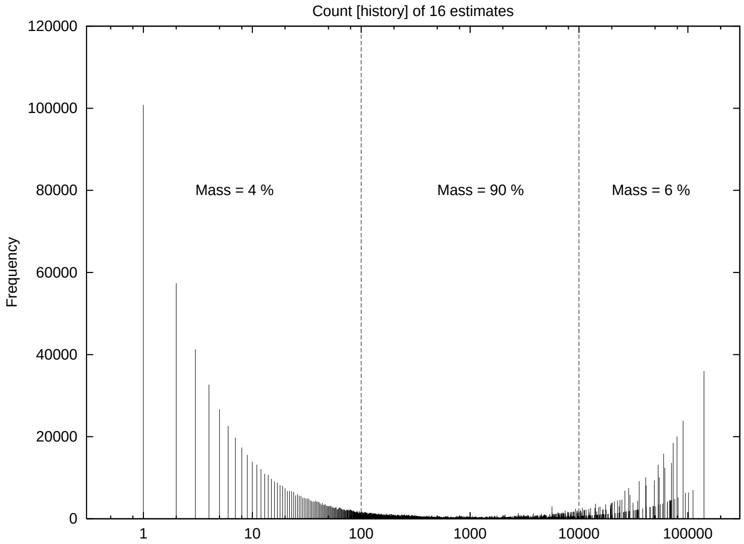

Figure 8: Histogram of history counts

defined accordingly. We will refer to this particular backoff scheme as *non-shadowing*[12].

$$p(w|h) = \begin{cases} \frac{C(h,w)-d_C}{C(h)} + \gamma(h)p_{back}(w|h) & \text{if } C(h,w) > 0 \\ \gamma(h)p_{back}(w|h) & \text{otherwise} \end{cases} \tag{10}$$

In both of these schemes the backoff distribution is given by the lower order estimate, such as a 2-gram serving as backoff for a 3-gram. Kneser and Ney proposed an alternative backoff distribution which performs better [23]:

$$p_{back}(w|h) = \frac{C(.,\hat{h},w)}{\sum_{w'} C(.,\hat{h},w')} \tag{11}$$

where

$$C(.,\hat{h},w) = \sum_{g:\hat{g}=\hat{h},C(g,w)>0} 1$$

Here $\hat{h}$ denotes a coarser history that is typically a 2-gram history if h denotes a 3-gram history. $C(.,\hat{h},w)$ corresponds to the number of different coarser histories $\hat{h}$ where the word w has been observed ignoring the frequency of these events.

In summary there are at least four possible methods of smoothing available. We can decide whether or not shadowing is used and whether or not

[12]Ney et al. uses the term *non linear interpolation* [22].

	Baseline	+KN backoff distributions	+non-shadowing	+both
C50 2g	126	125	127	123
3g	110	109	117	104
C300 2g	102	100	102	99
3g	89	87	91	82
C1600 2g	98	96	98	94
3g	85	84	85	78
Word 2g	97	95	97	94
3g	84	83	84	77

Table 2: Comparison of held-out perplexities for various backoff schemes

the Kneser-Ney (KN) backoff distributions are used. Table 2 summarizes the results obtained on the Switchboard data with various model orders and different class models. The baseline corresponds to the use of shadowing together with the use of lower order estimates as backoff distributions.

5.2 Two-dimensional backoff

The lattice structure (see figure 1) suggests an extension to the original backoff idea. If for example we consider a 3-gram predictor at the word level, there are two adjacent predictors which can be used as backoff distributions, namely a 2-gram at the word level and another 3-gram at the next class level.

Considering two backoff distributions was already proposed in a different context [24] where a speaker specific language model was combined with a non-specific language model. In that case however a hierarchy between the backoff distributions was defined *a priori*. To the contrary in the model proposed here both backoff distributions are combined by linear interpolation. Equations 12 and 13 formalize this idea in the case of shadowing and non-shadowing respectively.

$$p(w|h) = \begin{cases} \frac{C(h,w)-d_C}{C(h)} & \text{if } C(h,w) > 0 \\ \alpha(h,\lambda)[\lambda_1 p_{back_1}(w|h) + \lambda_2 p_{back_2}(w|h)] & \text{otherwise} \end{cases} \tag{12}$$

$$p(w|h) = \begin{cases} \frac{C(h,w)-d_C}{C(h)} + \gamma(h)[\lambda_1 p_{back_1}(w|h) + \lambda_2 p_{back_2}(w|h)] & \text{if } C(h,w) > 0 \\ \gamma(h)[\lambda_1 p_{back_1}(w|h) + \lambda_2 p_{back_2}(w|h)] & \text{otherwise} \end{cases} \tag{13}$$

5.3 EM estimation of the backoff weights

We show how to estimate the interpolation weights λ_i in the case of shadowing from some representative set of new data.

Let λ denotes the pair (λ_1, λ_2) with the constraint $\sum_i \lambda_i = 1$. Notice that λ appears in the normalization factor $\alpha(h, \lambda)$ of equation 12. We can write

$$\alpha(h, \lambda) = \frac{K(h)}{\lambda_1 S_1(h) + \lambda_2 S_2(h)} \tag{14}$$

where

$$K(h) = \sum_{w:C(h,w)>0} \frac{d_C}{C(h)},$$

$$S_1(h) = \sum_{w:C(h,w)=0} p_{back_1}(w|h) = 1 - \sum_{w:C(h,w)>0} p_{back_1}(w|h),$$

$$S_2(h) = \sum_{w:C(h,w)=0} p_{back_2}(w|h) = 1 - \sum_{w:C(h,w)>0} p_{back_2}(w|h).$$

Let $p_\lambda(w|h)$ denote the interpolated backoff distribution

$$p_\lambda(w|h) = \lambda_1 p_{back_1}(w|h) + \lambda_2 p_{back_2}(w|h)$$

The re-estimation formula can be derived from an auxiliary function Q representing the difference in the conditional expectation of the complete data log-likelihoods given the observed data. The hidden data is the actual sequence of backoff distributions used while predicting the new data. The function Q is defined as follows:

$$Q(\lambda'|\lambda) = \sum_B \tilde{p}(w|h) \log \frac{\alpha(\lambda', h) p_{\lambda'}(w|h)}{\alpha(\lambda, h) p_\lambda(w|h)} \tag{15}$$

where B denotes the set of backoff events on the new data and $\tilde{p}(h,w)$ denotes the relative frequency of the event (h,w) on this set. We can rewrite equation 15 as follows

$$\begin{aligned}
Q(\lambda'|\lambda) = & \sum_B \tilde{p}(w|h)\log\frac{p_{\lambda'}(w|h)}{p_\lambda(w|h)} - \sum_B \tilde{p}(w|h)\log\frac{\lambda'_1 S_1(h)+\lambda'_2 S_2(h)}{\lambda_1 S_1(h)+\lambda_2 S_2(h)} \\
= & \sum_B \tilde{p}(w|h)\log[\frac{\lambda_1 p_{back_1}(w|h)}{p_\lambda(w|h)}\frac{\lambda'_1}{\lambda_1} + \frac{\lambda_2 p_{back_2}(w|h)}{p_\lambda(w|h)}\frac{\lambda'_2}{\lambda_2}] - \\
& \sum_B \tilde{p}(w|h)\log\frac{\lambda'_1 S_1(h)+\lambda'_2 S_2(h)}{\lambda_1 S_1(h)+\lambda_2 S_2(h)} \\
\geq & \sum_B \tilde{p}(w|h)[\frac{\lambda_1 p_{back_1}(w|h)}{p_\lambda(w|h)}\log\lambda'_1 + \frac{\lambda_2 p_{back_2}(w|h)}{p_\lambda(w|h)}\log\lambda'_2 + C] - \\
& \sum_B \tilde{p}(w|h)\log\frac{\lambda'_1 S_1(h)+\lambda'_2 S_2(h)}{\lambda_1 S_1(h)+\lambda_2 S_2(h)} \\
\geq & \sum_B \tilde{p}(w|h)[\frac{\lambda_1 p_{back_1}(w|h)}{p_\lambda(w|h)}\log\lambda'_1 + \frac{\lambda_2 p_{back_2}(w|h)}{p_\lambda(w|h)}\log\lambda'_2 + C] + \\
& 1 - \sum_B \tilde{p}(w|h)\frac{\lambda'_1 S_1(h)+\lambda'_2 S_2(h)}{\lambda_1 S_1(h)+\lambda_2 S_2(h)}
\end{aligned}$$

where the first inequality is an application of Jensen's inequality[13], C does not depend on λ' and the second inequality results from the fact that $-\log x \geq 1-x$.
Computing the partial derivative of Q with respect to λ'_1

$$\begin{aligned}
\frac{\partial Q}{\partial \lambda'_1} = & \sum_B \tilde{p}(w|h)\frac{\lambda_1 p_{back_1}(w|h)}{p_\lambda(w|h)}\frac{1}{\lambda'_1} - \\
& \sum_B \tilde{p}(w|h)\frac{S_1(h)}{\lambda_1 S_1(h)+\lambda_2 S_2(h)}
\end{aligned} \tag{16}$$

and setting $\frac{\partial Q}{\partial \lambda'_1} = 0$, we get the re-estimation formula.

$$\lambda'_1 = \frac{\sum_B \tilde{p}(w|h)\frac{\lambda_1 p_{back_1}(w|h)}{p_\lambda(w|h)}}{\sum_B \tilde{p}(w|h)\frac{S_1(h)}{\lambda_1 S_1(h)+\lambda_2 S_2(h)}} \tag{17}$$

The numerator on the right hand side of equation 17 is analogous to that found in the classical re-estimation formula to compute interpolation weights between various language models. The difference here is that the sum is over the set B, which is the set of events in which a backoff occurred while predicting the new data. The main difference in this model lies in the denominator, as the normalization factor $\alpha(\lambda,h)$ is a function of λ.

In the case of non-shadowing the derivation is somewhat simpler and the re-estimation formula is given by equation 18

[13] If f is a convex function and X a random variable, then $f[E(X)] \leq E[f(X)]$.

Model	PP
Word 3g	84
Lattice	82
Lattice+linear interpolation	79
Linear interpolation	79

Table 3: Test set perplexity with shadowing

$$\lambda_1' = \frac{1}{C}\Big(\sum_{B} \tilde{p}(w|h)\frac{\lambda_1 p_{back_1}(w|h)}{p_\lambda(w|h)} + \sum_{\overline{B}} \tilde{p}(w|h)\frac{\gamma(h)\lambda_1 S_1(h)}{p_0(w|h)+\gamma(h)(\lambda_1 p_{back_1}(w|h)+\lambda_2 p_{back_2}(w|h))} \Big) \quad (18)$$

where C is a normalization constant that satisfies the constraint $\sum_i \lambda_i = 1$, B is the set of events (h, w) on the new data such that $C(h, w) = 0$ on the training data, $\overline{B}$ is the complement of B, and $p_0(w|h) = \frac{C(h,w)-d_C}{C(h)}$.

Generalization of the proposed approach to more than two backoff distributions is trivial and will not be detailed here. Moreover this model is not restricted to lattices of N-grams but can be applied to other cases where several backoff distributions are relevant. Finally, one should stress that backoff models are usually applied recursively to several predictors . The proposed re-estimation should then be applied first with less specific predictors and then iterated with more specific ones. There is no guarantee however that such an iterated re-estimation will globally maximize the likelihood of the most specific model.

5.4 Preliminary results

We describe in this section some preliminary results obtained on the Switchboard data with lattices of N-gram models smoothed with two-dimensional backoff. In these experiments we restricted the lattices to 3-gram models with 4 class levels. Figure 9 shows the interpolation weights (plain arrows) estimated on held-out data. For example, from a 3-gram at the word level the backoff weight of the word 2-gram is 0.58 while it would be 1.0 in a traditional one-dimensional backoff scheme. A conventional interpolation of the 4 higher order predictors can be performed on top of the two-dimensional backoff. The so-called global interpolation weights are represented by dashed arrows in figure 9.

Table 3 presents the test set perplexity obtained with these models in the case of shadowing. The first line corresponds to the reference model, that is

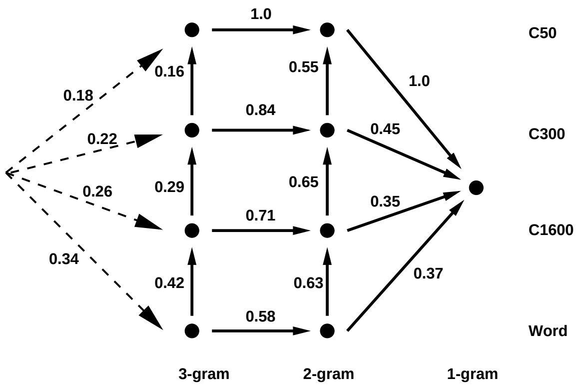

Figure 9: 2-D backoff interpolation weights (shadowing)

Model	PP
Word 3g	77
Lattice	77
Lattice+linear interpolation	74
Linear interpolation	74

Table 4: Test set perplexity with non-shadowing and KN distributions

a word 3-gram model. The second line gives the test set perplexity of the model obtained after estimation of the backoff weights on held-out data. The third line shows the additional improvement which can be obtained with the global weights. This last result represents a 6 % perplexity reduction over the reference model. However, the same reduced perplexity can be obtained with global interpolation only[14]. A similar conclusion can be drawn from table 4, in which non-shadowing is used and the bigram predictors are replaced by their corresponding KN distributions. As pointed out in section 5.3 there is no guarantee that the iterated reestimation globally maximizes the likelihood of the most specific model. We observe here a practical case where lattice based language models do not outperform linearly interpolated models.

This result is somewhat surprising, as the two-dimensional backoff model contains more free parameters. Additional experiments should be performed to confirm the source of this result. In particular, larger lattices could be used and two-dimensional backoff could be generalized to more than two predictors. Another interesting extension would rely on the definition of backoff weights depending on the history h. In such a case, the weights could

[14] In this case the backoff weights are fixed to 0.0 in the vertical direction, and 1.0 otherwise.

be optimized for each history instead of globally with only a few additional free parameters (6 in the previous example).

6 Dynamic combination of predictors

6.1 Predictor combination

Another way of looking at the lattice presented in figure 1 is to consider each node as the starting point for a recursive backoff scheme that is limited to progressing in one-dimension – the horizontal one. Figure 10 presents the perplexity obtained using this scheme on the held-out set from each lattice node. In this case, KN distributions are used as backoff distributions combined with non-shadowing. The reference word 3-gram model has a perplexity of 77 which is only slightly outperformed by the word 4-gram and 5-gram models. The dynamic combination of predictors can then be perceived as a way to combine these 17 predictors in order to improve over the reference model.

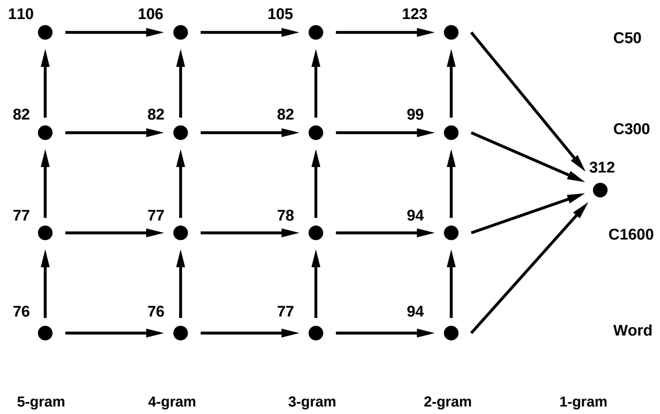

Figure 10: Held-out perplexity with KN distributions. Each node represents a predictor which starts at that node and backs off horizontally.

6.2 Predicting the Oracle decisions

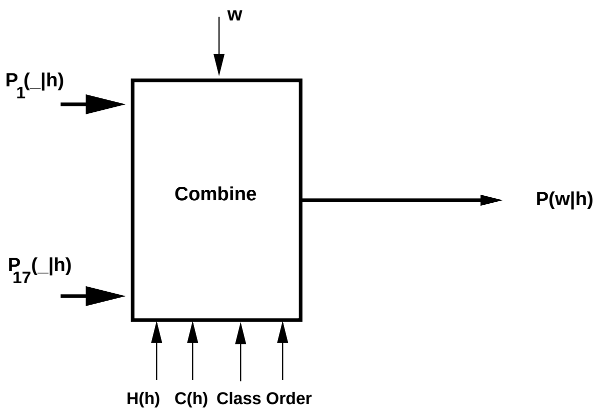

Figure 11: Use of predictor variables to choose the optimal combination

Suppose we had an oracle that knew which of the 17 predictors should be used in any given context to predict the next word. A very well informed oracle could look at the word to be predicted and pick the best predictor accordingly. The perplexity on the held-out set would then drop to 38.

A more realistic framework is to combine the predictors linearly but to adapt dynamically the interpolation weights between the predictors. To do this, we can rely on four predictor variables: the entropy $H(h)$, the count $C(h)$, the number of classes and the model order (see figure 11).

Using these predictor variables, a decision tree can be built from the held-out set to predict the probability that any predictor would outperform the reference model by some factor (typically set to 1.5). Such a decision tree is presented in figure 12 where left branches correspond to "yes" answers. For example the probability that any randomly selected predictor (except the word 3-gram itself) would significantly outperform the reference model is 0.18 in general. This probability drops to 0.01 when the number of classes is below 950 and the count of the history is above 126,204.

The relative weight of any predictor can be made proportional to the probability attached to the leaf into which it falls. Notice that even though the decision tree is fixed, the weight applied to a predictor changes dynamically, as $H(h)$ and $C(h)$ depend on the observed history.

Table 5 presents test set perplexities that result from experiments based on this approach[15]. The perplexity of the best reference model (word 3gram with non-shadowing and KN distributions) is 77. A static interpolation with

[15]The decision tree used in practice was more detailed than the one shown in figure 12.

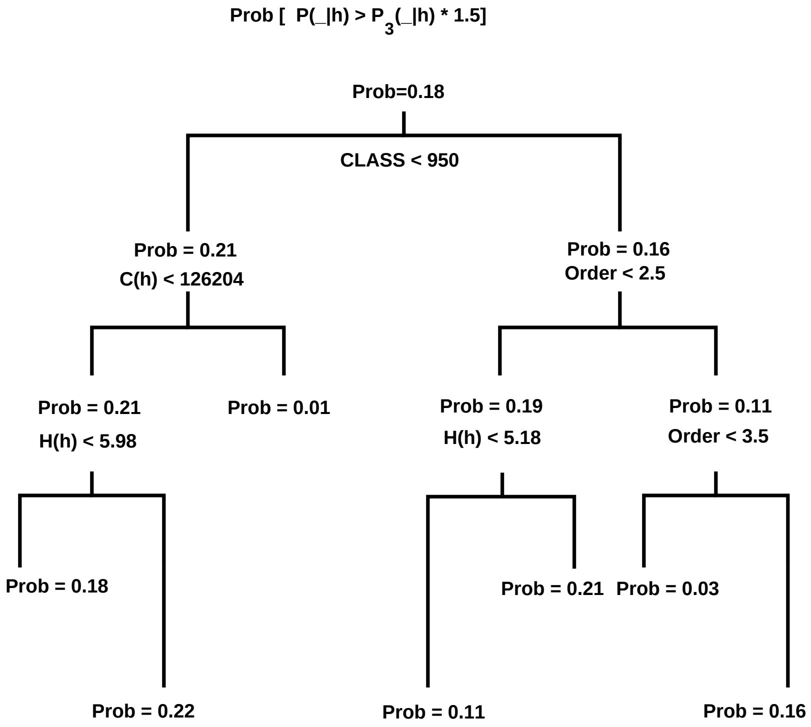

Figure 12: Decision tree

Model	PP
Word 3g	77
Lattice + Interpolation	74
Lattice + Decision Trees	72

Table 5: Test set perplexities

globally optimized but fixed weights yields a perplexity of 74. The dynamic combination of the lattice predictors gives a perplexity of 72.

7 Summary and Conclusions

In this report we introduced lattice based language models — an alternative language modeling paradigm which we have just started exploring. Lattice based models construct multi-dimensional hierarchies of partitions and then select the most promising partitions (nodes) to generate the estimated distributions.

We discussed a specific two dimensional lattice, where the first dimension is the length of the history equivalence class, and the second dimension is the position in a word class hierarchy. As originally defined such a lattice is in fact a DAG, since subsumption relations exist among neighboring vertices. Simple set subtraction operations can remove these data dependencies.

Next, we considered which features of the lattice nodes are indicative of their usefulness, and proposed the use of two primary ones: the training-set history count of the node, and the (smoothed) entropy of its prediction. Using the SWITCHBOARD corpus, we constructed a two dimensional, 17 node lattice, and calculated history and prediction hit ratios for all its nodes using held out data. We then demonstrated how the prediction hit ratio depends strongly on both the count of the history and its entropy, thus justifying our original choice.

After discussing various smoothing techniques, we proposed a straightforward generalization of the conventional backoff strategy to multiple dimensions, and derived the formula for calculating the optimal interpolation weights, using the Estimation-Maximization (EM) algorithm. This simple model provided a modest improvement over the baseline trigram. Another interpolation scheme, using the same predictor set, achieved the same performance.

The true strength of lattice models, we believe, lies in dynamic selection of a small subset of predictor nodes. How to select such a set is an open and interesting research problem, which we have just begun to look at. Oracle experiments suggest that significant improvements are possible if we choose the predictor set correctly. And indeed, an initial attempt at using a decision tree to make that selection yielded some improvement. We believe much more improvement is possible, and are hoping to explore this problem in greater detail in the future.

8 Acknowledgements

Thanks to John Lafferty for his contribution to the derivation of the re-estimation of the backoff weights, and to Lin Chase for her contribution to the design of the decision tree model and for her careful editing of this document.

References

[1] F. Jelinek and R. Mercer. Interpolated estimation of markov source parameters from sparse data. In E. Gelsema and L. Kanal, editors, *Pattern Recognition in Practice*, pages 381–397. North-Holland, Amsterdam, 1980.

[2] F. Jelinek. Up from trigrams, the struggle for improved language models. In *European Conference on Speech Communication and Technology*, pages 1037–1040, Berlin, 1993.

[3] I.J.Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237–264, 1953.

[4] H. Ney, U. Essen, and R. Kneser. On structuring probabilistic dependences in stochastic language modelling. *Computer Speech and Language*, 8:1–38, 1994.

[5] S.M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 35(3):400–401, 1987.

[6] P. Brown, V. Della Pietra, P. de Souza, J. Lai, and R. Mercer. Class-based N-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

[7] R. Kneser and H. Ney. Forming word classes by statistical clustering for statistical language modeling. In *Proceedings of the 1st QUALICO Conference*, Trier, Germany, September 1991.

[8] R. Kuhn. Speech recognition and the frequency of recently used words: A modified markov model for natural language. In *COLING 88, 12th International Conference on Computational Linguistics*, pages 348–350, Budapest, August 1988.

[9] R. Kuhn and R. De Mori. A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):570–583, 1990.

[10] R. Kuhn and R. De Mori. Correction to a cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(6):691–692, June 1992.

[11] F. Jelinek, B. Merialdo, S. Roukos, and M. Strauss. A dynamic language model for speech recognition. In *Proceedings of the DARPA Workshop on Speech and Natural Language*, pages 293–295, February 1991.

[12] L. Bahl, P. Brown, P. de Souza, and R. Mercer. A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 37(7):1001–1008, 1989.

[13] S. Della Pietra, V. Della Pietra, R. Mercer, and S. Roukos. Adaptive language modeling using minimum discriminant estimation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 633–636, San Francisco, March 1992.

[14] R. Lau, R. Rosenfeld, and S. Roukos. Trigger-based language models: a maximum entropy approach. In *International Conference on Acoustic, Speech and Signal Processing*, volume II, pages 45–48, 1993.

[15] R. Rosenfeld. *Adaptive Statistical Language Modeling: a Maximum Entropy Approach.* Ph. D. dissertation, TR CMU-CS-94-138, Computer Science Department, Carnegie Mellon University, 1994.

[16] A. Berger, S. Della Pietra, and V. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1), March 1996.

[17] R. Rosenfeld. A maximum entropy approach to adaptive statistical language modeling. *Computer, Speech and Language*, 10:187–228, 1996.

[18] K. Seymore and R. Rosenfeld. Scalable backoff language models. In *International Conference on Spoken Language Processing*, pages 232–235, 1996.

[19] R. Kneser and H. Ney. Improved clustering techniques for class-based language modelling. In *European Conference on Speech Communication and Technology*, pages 973–976, Berlin, 1993.

[20] J.J. Godfrey, E.C. Holliman, and J. McDaniel. Switchboard: Telephone speech corpus for research and development. In *International Conference on Acoustic, Speech and Signal Processing*, pages 517–520, Detroit, 1995.

[21] I.J. Good. The population frequencies of species and the estimation of population parmaters. *Biometrika*, 40:237–264, 1953.

[22] H. Ney, U. Essen, and R. Kneser. On structuring probabilistic dependences in stochastic language modelling. *Computer Speech and Language*, 8:1–38, 1994.

[23] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *International Conference on Acoustic, Speech and Signal Processing*, pages 181–184, Detroit, 1995.

[24] S. Besling and H.-G. Meier. Language model speaker adaptation. In *European Conference on Speech Communication and Technology*, Madrid, 1995.