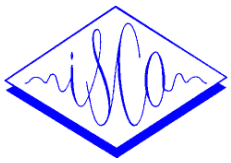

3rd European Conference on
Speech Communication and Technology
EUROSPEECH'93
Berlin, Germany, September 19-23, 1993

IMPROVED CLUSTERING TECHNIQUES FOR CLASS-BASED STATISTICAL LANGUAGE MODELLING

Reinhard Kneser, Hermann Ney

Philips GmbH Forschungslaboratorien
Weißhausstraße 2, D-52066 Aachen, Germany
e-mail: kneser@pfa.philips.de

ABSTRACT

The topic of this paper is to introduce a certain type of structure into a bigram language model by using the concept of word equivalence classes. We train these classes automatically, using an iterative clustering algorithm which finds a (local) optimum of some clustering criterion. We show that the conventional maximum-likelihood criterion performs well, but has the disadvantage that one has to specify the number of word classes in advance. We therefore modify this criterion using a special form of cross-validation, the leaving-one-out technique. The resulting algorithm is able to find both the unknown classification and the unknown number of classes at the same time. Clustering experiments were carried out on an English and a German text corpus comprising 1.1 million and 100,000 words, respectively. Compared to a word bigram model we could reduce the perplexity by more than 10% using a class model with automatically clustered classes. Combinations with the word model and with linguistically defined parts of speech lead to a further improvement of up to 37%.

Keywords: Stochastic Language Modelling, Statistical Clustering, Leaving-One-Out Method

1. INTRODUCTION

The task of a stochastical language model in automatic speech recognition is to provide estimates of the probability $Pr(w_1 \ldots w_N)$ of a word sequence $w_1 \ldots w_N$. This joint probability is computed from the conditional probabilities $Pr(w_i|w_1 \ldots w_{i-1})$ [1]. In this paper, we consider only conditional bigram probabilities $Pr(w_i|w_{i-1})$; the concepts presented, however, are more general and can easily be extended to trigram and m-gram models. Even in the relatively simple case of bigrams, we are faced with the problem of sparse data, i.e. most of the possible word pairs could not be seen in the training data since there are too many of them. The introduction of a certain type of structure into the model by using the concept of word equivalence classes reduces the number of free parameters and so one might expect to get more robust estimates. A further advantage of the use of word-class models in the speech recognition process is the typical reduction of the potential search space. Assuming that each word w belongs to exactly one class $G(w)$ of 'similar' or 'equivalent' words, we factorize the conditional bigram probability:

$$Pr(w|v) = p_0(w|G(w))\, p_1(G(w)|G(v)), \qquad (1)$$

where $p_0(.)$ denotes the class membership probability and $p_1(.)$ denotes the bigram probability at the level of the word classes.

Parts-of-Speech (POS), defined by linguistic experts serve usually as word classes[2]. Although working well, this employs manual work each time a new word is added to the vocabulary. We pursue another approach avoiding this disadvantage by learning word equivalence classes automatically from a training corpus $w_1 \ldots w_N$. We first introduce an optimization criterion $F(G)$ which indicates how well a classification $G : w \rightarrow G(w)$ 'explains' the training data. The classification problem can then be formulated as the optimization problem of finding that classification G of words that optimizes this criterion:

$$G = \underset{G' \in \mathcal{G}}{\operatorname{argmax}}\, F(G'). \qquad (2)$$

Here $\mathcal{G}$ denotes the set of possible classifications which usually will contain all subdivisions of the vocabulary into equivalence classes. In the following sections we derive two optimization criteria based on the maximum-likelihood concept and present a clustering algorithm that searches for an optimum.

Even in the case of a word-class bigram model it might happen that not all possible bigrams occurred in the training data. Interpolation with less specific models is therefore necessary. In all our experiments we use nonlinear discounting as described in [3].

2. MAXIMUM-LIKELIHOOD CRITERION

There are two sets of parameters in the class model, the unknown classification G and the estimates $p_G(w_i|w_{i-1})$ of the conditional probabilities which depend on G. Given these parameters we can calculate the likelihood of the training data, i.e. the probability that the training data would be generated by the model given the parameters:

$$L = \prod_{i=1}^{N} p_G(w_i|w_{i-1}). \qquad (3)$$

The likelihood is a measure of how well the training data is represented by the parameters and can be used as optimization criterion [4]. Since we want to maximize this likelihood we should take p_G to be the maximum-likelihood (ML) estimator, i.e. relative frequencies. This leads to the conventional ML criterion

$$F_{ML} = \prod_{i=1}^{N} \frac{N(G(w_{i-1}), G(w_i))}{N(G(w_{i-1}))} \frac{N(w_i)}{N(G(w_i))} \qquad (4)$$

where $N(.)$ is the count of the concerning event in the training data.
In order to keep computational cost low we may remove in Eq.(4) terms not depending on G since they do not affect the result of the optimization in Eq.(2). Further, taking the logarithm, with the convention of '$0 \ln(0) = 0$', and rearranging the terms, we obtain the equivalent optimization criterion

$$\begin{aligned} F'_{ML} &= \sum_{g_1,g_2} N(g_1,g_2) \ln N(g_1,g_2) \\ &\quad -2 \sum_{g} N(g) \ln N(g), \end{aligned} \qquad (5)$$

where we take the sums over all word classes g and all pairs g_1, g_2 of word classes in G, respectively.
The likelihood of every class-based bigram model, which at the same time is also a word model, must be less or equal than the maximal likelihood of a word bigram model. Thus, using the ML criterion, the word bigram model would be the optimal result if it were contained in the set $\mathcal{G}$ of admitted classifications. It is therefore necessary to limit the set $\mathcal{G}$ of possible classifications. A suitable method is to impose an upper limit M on the number of classes. This extra parameter must then also be trained.

3. LEAVING-ONE-OUT METHOD

In the case of the conventional ML criterion, both the classification G and the probability estimates p_G are trained on the same training set. Unlike in an independently drawn corpus, all class bigrams have been observed in the training data, a fact that overestimates the generalization power of the class model. We shall avoid this by incorporating a cross-validation technique directly into the optimization criterion. The basic principle of cross validation is to simulate unseen events by splitting the training data into 2 parts, a 'retained' part T_R and a 'held-out' part T_H. Let us assume that we have an estimator $p_{G,T}$ for the probabilities of the class model G, given a training set T. We can then use the 'retained' part to estimate the probabilities and the 'held-out' part to optimize the classification G:

$$G = \operatorname*{argmax}_{G' \in \mathcal{G}} p_{G',T_H}(T_R). \qquad (6)$$

The so-called leaving-one-out method is a special type of cross-validation [5, pp.75]. Here, the training text is divided into $N-1$ samples as the 'retained' part and only 1 sample as the 'held-out' part. This process is repeated N times so that all N partitions with 1 'held-out' are considered. The basic advantage of this approach is that all samples are used both in the 'retained' part and in the 'held-out' part and thus a very efficient exploitation of the training text is achieved. With the notation of T_i being the training corpus $w_1 \ldots w_N$ with the event (w_{i-1}, w_i) removed, we get the leaving-one-out likelihood

$$L_{LO} = \prod_{i=1}^{N} p_{G',T_i}(w_i|w_{i-1}), \qquad (7)$$

which is to be maximized.
In order to get the final optimization criterion we have to specify the estimator $p_{G,T}$. This must be an interpolated model since we might have to predict unseen events. Taking the same interpolation as used in the final model would result in optimal performance, but for computational reasons we chose the following scheme to modify maximum-likelihood estimates. First we rewrite $p_{G,T}(w|v)$ as

$$p_{G,T}(w|v) = \frac{p_{G,T}(G(v), G(w))}{p_{G,T}(G(w))} \frac{p_{G,T}(v)}{p_{G,T}(G(w))}. \qquad (8)$$

We can guarantee, as we will see later, that in our experiments every class has been seen at least once in the 'retained' part such that we can take relative counts as estimates for the class unigrams:

$$p_{G,T}(g) = \frac{N_T(g)}{N_T}, \qquad (9)$$

where N_T denotes the occurrence count of an event in the part T. In the bigram case, where we really have to predict unseen events, we use the absolute discounting method [6], where we 'discount' the counts by a constant value $b < 1$ and redistribute the thus gained probability mass over all unseen events. With the notation of $n_{0,T}$ being the number of unseen and $n_{+,T}$ for the number of seen bigrams this is:

$$p_{G,T}(g_1,g_2) = \begin{cases} \dfrac{N_T(g_1,g_2) - b}{N_T} & \text{if } N_T(g_1,g_2) > 0 \\ \dfrac{n_{+,T}\, b}{n_{0,T}\, N_T} & \text{if } N_T(g_1,g_2) = 0 \end{cases} \qquad (10)$$

In our experiments we used the empirically determined constant value of $b = 0.75$, rather than training this parameter for each classification separately. The estimate of $p_{G,T}(w)$ may be left unspecified since it is actually independent of G and thus is a constant in the optimization Eq.(7).
Leaving one event out reduces the count of this event in the whole training corpus by 1. Similarly, leaving out an event that occurs exactly once, increments the number n_0 of unseen bigrams and decrements the number n_1 of bigrams seen once. It is thus easy to specify the 'leaving-one-out' estimates $p_{G,T_i}(w_i|w_{i-1})$ in terms of counts $N(.)$, n_+, n_1 and n_0 taken over the whole training set. Substituting these estimates in Eq.(7) and performing similar

modifications as for Eq.(5) leads to the final leaving-one-out optimization criterion

$$F_{LO} = \sum_{g_1,g_2} N(g_1,g_2)\ln[N(g_1,g_2) - 1 - b]$$

$$+ n_1 \ln\left[\frac{(n_+ - 1)b}{n_0 + 1}\right] - 2\sum_{g} N(g)\ln[N(g) - 1]. \quad (11)$$

4. CLUSTERING ALGORITHM

Now as we have defined the optimization criteria Eqs.(5) and (11), we have to specify an algorithm in order to find the optimum. In the spirit of decision-directed learning [5, p.210], the basic concept of the algorithm is to improve the value of the optimization criterion by making local optimizations, which means moving a word from one class to another in order to improve the optimization criterion. Thus we obtain the following clustering algorithm:

Start with some initial mapping $G : w \rightarrow G(w)$
Iterate until some convergence criterion is met
Loop over all words w
Loop over all clusters g'
Check how the optimization criterion changes if w is moved from its current cluster $g = G(w)$ to g'
Move word w to the cluster g' which results in the highest optimization criterion

From this description, it can be seen that the optimization criterion is bound to increase (or stay constant) in every iteration step. Since the optimization criterion is, up to constants, the logarithm of a probability, it has some upper limit and thus the obtained monotone sequence is bound to converge. However, the resulting solution is only locally optimal since it depends on the start conditions.
During each iteration, moving a word w from its present cluster $g = G(w)$ to a new cluster g' will change only the counts of $g = G(w)$ and g'. In addition, regarding the leaving-one-out case, the values n_0, n_1 and n_+ might change. Therefore we can get the new value of the optimization criterion from the old one by adjusting only these counts in Eqs.(5) and (11), respectively. The complexity of this algorithm is hence of the order $M^2 * W * I$, where W is the vocabulary size, i.e. the number of words to be clustered, M is the maximal number of clusters and I the number of iterations.
Since the move of a word to another cluster affects the clustering of the subsequent words in the training corpus, the order in which words are moved is crucial. We know most about the frequent words and should cluster them first. Hence we sorted the words in descending order of frequency so that in each iteration the most frequent words were processed first.
The algorithm requires some initial mapping or partition G to start with. Knowing nothing about the words to be clustered in the case of the English corpus, we simply put all into one single cluster. For the German corpus we start with three clusters: a class for capitalized words (nouns), non-capitalized words and numbers, respectively.
Words of the vocabulary that do not occur in the training data do not affect the optimization criteria of Eqs.(5) and (11). They are not moved in the clustering procedure and remain in the cluster as specified by the initial partition. We also exclude words with small occurrence counts, say less than 5, from the clustering since they are not very reliable and they give a good estimate for the unseen words. This also guarantees that all classes have always been observed, even if we leave out an event.

5. EXPERIMENTAL RESULTS

We performed experiments on a German newspaper corpus (100,000 words) and an English database (the LOB corpus, 1.1 million words). The size of the vocabulary, consisting af all full-form words that occurred in the whole corpus is 14,000 for the German and 50,000 for the English database. The words in both corpora were labelled with exactly one out of 302 and of 153 parts of speech for the German and the English database, respectively. We took 1/4 of the corpora as test set and 3/4 as training set. All models were interpolated using the nonlinear discounting technique. The test set perplexity as defined in [1] was used as performance criterion.

Table 1: Test set perplexity with clustered categories

M	German	English
1 (word unigram)	1185	1138
30	674	647
60	589	561
90	571	538
120	→ **557**	514
150	563	500
200	566	486
250	576	479
350	579	→ **478**
700	586	484
49615 (word bigram)		541
14080	650	

We performed clustering experiments using both, the conventional ML and the leaving-one-out criterion. The maximal number of clusters M had to be specified in advance for the conventional ML criterion. Table 1 shows the test-set perplexities obtained for various numbers M. Starting with the special case of one cluster, the word unigram, the perplexity drops fast with increasing number of clusters. Depending on the corpus we have a flat optimum at about 120 respectively 350. From there the perplexity increases again slowly toward the word-bigram value. We took the number of classes optimized on the test-set as an optimistic estimate of the performance of the conventional ML

criterion (ML-Class). In contrast to the conventional ML criterion, the leaving-one-out criterion was able to find the optimal number of classes together with the classification (LO-Class). These numbers, 179 and 477, respectively, were slightly higher than the optimal numbers found for the conventional ML criterion.

Table 2: Test set perplexity of cluster models

Model	German	English
Word	650	541
POS	485	556
LO-Class	555	479
LO-Class + Word	494	442
LO-Class + Word + POS	407	420
ML-Class	557	478
ML-Class + Word	492	439
ML-Class + Word + POS	408	420

Table 2 shows the perplexities of the clustering experiments in comparison with the word bigram model (Word) which serves as baseline model, and the class model based on parts-of-speech (POS). In addition it shows the results of a further experiment, where we combined the cluster models with the word bigram and the parts-of-speech model. The combination was achieved by linear interpolation [4], with the interpolation parameters estimated on a small additional training set comprising about 4,000 words. We see that there is almost no difference in the perplexity of the two clustering criteria. In the case of the smaller German corpus, where the word bigram model is not so well trained and thus cluster models in general perform better, the clustering model reduced the perplexity of the baseline model by 15%. The combination of the cluster model with the word bigram model increased the improvement to 24%, and further adding the POS model finally resulted in a 37% lower perplexity. The experiments on the English corpus lead to lower, but still significant improvements, ranging between 11% and 24%.
Table 3 lists three typical examples of clusters created by the clustering algorithm. We see that most of the words belong to the same syntactic class, in our example past tense verbs, plural nouns, and adjectives, respectively. Furthermore there are semantical similarities between the words. The majority of the words in cluster A are verbs expressing some kind of movement, and cluster B contains mainly words specifying a group of people. Although most of the clusters are more or less intuitively satisfying, there are also clusters which are difficult to understand from a linguistic point of view.

Table 3: Most frequent words of some sample clusters from the English corpus

Cluster A: *turned carried opened sent moved caught laid drew pulled lifted threw pushed handed pressed crossed burst thrust slipped swept stretched poured plunged wrapped weighed switched dragged waved ruled rolled rang ...*

Cluster B: *people men children women boys girls persons students animals teachers officers soldiers stars Americans informants doctors employees Indians Africans prisoners individuals couples servants farmers doors conservatives critics folk artists visitors ...*

Cluster C: *important serious interesting effective useful popular significant suitable dangerous familiar successful powerful appropriate positive expensive excellent attractive odd complex satisfactory exciting angry vital complicated valuable unexpected outstanding exact improved critical ...*

6. SUMMARY

We presented a clustering algorithm to learn automatically abstract word equivalence classes from a training corpus. We derived a modified maximum-likelihood optimization criterion using the leaving-one-out technique which avoids the shortcomings of the conventional maximum-likelihood criterion. We could significantly improve the statistical language model by using the automatically clustered word classes. Further improvements were achieved by combination of the class model with word models and with part-of-speech models.

REFERENCES

[1] L. R. Bahl, F. Jelinek, R. L. Mercer: "A maximum likelihood approach to continuous speech recognition", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 5, pp. 179-190, March 1983.

[2] A. M. Derouault, B. Merialdo: "Natural language modeling for phoneme-to-text transcription", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 8, pp. 742-749, Nov. 1986.

[3] H. Ney, U. Essen: "On smoothing techniques for bigram-based natural language modelling", *Proc. ICASSP*, Vol. 2, pp. 825-828, May 1991.

[4] F. Jelinek: "Self-organized language modeling for speech recognition", pp. 450-506, in A. Waibel, K.-F. Lee (eds.): *Readings in Speech Recognition*, Morgan Kaufman Publishers, 1991.

[5] R. O. Duda, P. E. Hart: *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[6] H. Ney, U. Essen: "Estimating 'small' probabilities by leaving-one-out", *Proc. Eurospeech*, Sept. 1993.