On Combining Language Models : Oracle Approach*

Kadri Hacioglu and Wayne Ward
Center for Spoken Language Research
University of Colorado at Boulder
{hacioglu,whw}@cslr.colorado.edu

ABSTRACT

In this paper, we address the problem of combining several language models (LMs). We find that simple interpolation methods, like log-linear and linear interpolation, improve the performance but fall short of the performance of an oracle. The oracle knows the reference word string and selects the word string with the best performance (typically, word or semantic error rate) from a list of word strings, where each word string has been obtained by using a different LM. Actually, the oracle acts like a dynamic combiner with hard decisions using the reference. We provide experimental results that clearly show the need for a dynamic language model combination to improve the performance further. We suggest a method that mimics the behavior of the oracle using a neural network or a decision tree. The method amounts to tagging LMs with confidence measures and picking the best hypothesis corresponding to the LM with the best confidence.

1. INTRODUCTION

Statistical language models (LMs) are essential in speech recognition and understanding systems for high word and semantic accuracy, not to mention robustness and portability. Several language models have been proposed and studied during the past two decades [8]. Although it has turned out to be a rather difficult task to beat the (almost) standard class/word n-grams (typically $n = 2$ or 3), there has been a great deal of interest in grammar based language models [1]. A promising approach for limited domain applications is the use of semantically motivated phrase level stochastic context free grammars (SCFGs) to parse a sentence into a sequence of semantic tags which are further modeled using n-grams [2, 9, 10, 3]. The main motivation behind the grammar based LMs is the inability of n-grams to model longer-distance constraints in a language. With the advent of fairly fast computers and efficient parsing and search schemes several researchers have focused on incorporating relatively complex language models into speech recognition and understanding systems at different levels. For example, in [3], we report a significant perplexity improvement with a moderate increase in word/semantic accuracy, at N-best list (rescoring) level, using a dialog-context dependent, semantically motivated grammar based language model.

Statistical language modeling is a "learning from data" problem. The generic steps to be followed for language modeling are

- preparation of training data
- selection of a model type
- specification of the model structure
- estimation of model parameters

The training data should consist of large amounts of text, which is hardly satisfied in new applications. In those cases, complex models fit to the training data. On the other hand, simple models can not capture the actual structure. In the Bayes' (sequence) decision framework of speech recognition/understanding we heavily constrain the model structure to come up with a tractable and practical LM. For instance, in a class/word n-gram LM the dependency of a word is often restricted to the class that it belongs and the dependency of a class is limited to n-1 previous classes. The estimation of the model parameters, which are commonly the probabilities, is another important issue in language modeling. Besides data sparseness, the estimation algorithms (e.g. EM algorithm) might be responsible for the estimated probabilities to be far from optimal.

The aforementioned problems of learning have different effects on different LM types. Therefore, it is wise to design LMs based on different paradigms and combine them in some optimal sense. The simplest combination method is the so called linear interpolation [4]. Recently, the linear interpolation in the logarithmic domain has been investigated in [6]. Perplexity results on a couple of tasks have shown that the log-linear interpolation is better than the linear interpolation. Theoretically, a far more powerful method for LM combination is the maximum entropy approach [7]. However, it has not been widely used in practice, since it is computationally demanding.

In this research, we consider two LMs:

- class-based 3-gram LM (baseline).
- dialog dependent semantic grammar based 3-gram LM [3].

After N-best list rescoring experiments with linear and log-linear interpolation, we realized that the performance in terms of word and semantic accuracies fall considerably short of the performance of an oracle. We explain the set-up for the oracle experiment and point out that the oracle is a dynamic LM combiner. To fill the performance gap, we suggest a method that can mimic the oracle.

*The work is supported by DARPA through SPAWAR under grant #N66001-00-2-8906.

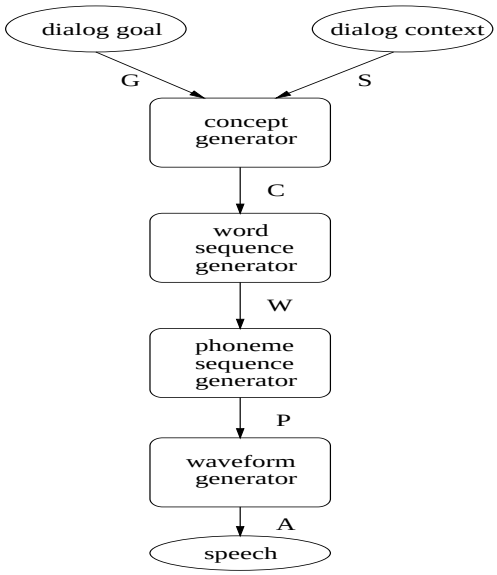

Figure 1: A speech production model

The paper is organized as follows. Section 2 presents the language models considered in this study. In Section 3, we briefly explain combining of LMs using linear and log-linear interpolation. Section 4 explains the set up for the oracle experiment. Experimental results are reported in Section 5. The future work and conclusions are given in the last section.

2. LANGUAGE MODELS

In language modeling, the goal is to find the probability distribution of word sequences, i.e. $P(W)$, where $W = w_1, w_2 \cdot \cdots, w_L$. We first describe a model for sentence generation in a dialog [5] on which our grammar LM is based. The model is illustrated in Figure 1. Here, the user has a specific goal that does not change throughout the dialog. According to the goal and the dialog context the user first picks a set of concepts with respective values and then use phrase generators associated with concepts to generate the word sequence. The word sequence is next mapped into a sequence of phones and converted into a speech signal by the user's vocal apparatus which we finally observe as a sequence of acoustic feature vectors.

Assuming that

- the dialog context S is given,
- W is independent of S but the concept sequence C, i.e. $P(W/C, S) = P(W/C)$,
- (W,C) pair is unique (possible with either Viterbi approximation or unambigious association between C and W),

one can easily show that $P(W)$ is given by

$$P(W) = P(W/C)P(C/S) \tag{1}$$

In (1) we identify two models:

- Concept model: $P(C/S)$
- Syntactic model : $P(W/C)$

<s> I WANT TO FLY FROM MIAMI FLORIDA TO SYDNEY AUSTRALIA ON OCTOBER FIFTH </s>
<s> [i_want] [depart_loc] [arrive_loc] [date] </s>

<s> I DON'T TO FLY FROM MIAMI FLORIDA TO SYDNEY AFTER AREA ON OCTOBER FIFTH </s>
<s> [Pronoun] [Contraction] [depart_loc] [arrive_loc] [after] [Noun] [date] </s>

Figure 2: Examples of parsing into concepts and filler classes

The concept model is conditioned on the dialog context. Although there are several ways to define a dialog context, we select the last question prompted by the system as the dialog context. It is simple and yet strongly predictive and constraining.

The concepts are classes of phrases with the same meaning. Put differently, a concept class is a set of all phrases that may be used to express that concept (e.g. [i_want], [arrive_loc]). Those concept classes are augmented with single word, multiple word and a small number of broad (and unambigious) part of speech (POS) classes. In cases where the parser fails, we break the phrase into a sequence of words and tag them using this set of "filler" classes. Two examples in Figure 2 clearly illustrate the scheme.

The structure of the concept sequences is captured by an n-gram LM. We train a seperate language model for each dialog context. Given the context S and $C = c_0 c_1 \cdots c_K, c_{K+1}$, the concept sequence probabilities are calculated as (for $n = 3$)

$$P(C/S) = P(c_1/ < s >, S)P(c_2/ < s >, c_1, S) \prod_{k=3}^{K+1} P(c_k/c_{k-2}, c_{k-1}, S)$$

where c_0 and c_{K+1} are for the sentence-begin and sentence-end symbols, respectively.

Each concept class is written as a CFG and compiled into a stochastic recursive transition network (SRTN). The production rules define complete paths beginning from the start-node through the end-node in these nets. The probability of a complete path traversed through one or more SRTNs initiated by the top-level SRTN associated with the concept is the probability of the phrase given that concept. This probability is calculated as the multiplication of all arc probabilities that defines the path. That is,

$$\begin{aligned} P(W/C) &= \textstyle\prod_{i=1}^{K} P(s_i/c_i) \\ &= \textstyle\prod_{i=1}^{K} \prod_{j=1}^{M_i} P(r_j/c_i) \end{aligned}$$

where s_i is a substring in $W = w_1, w_2..w_L = s_1, ..s_2, s_K$ ($K \leq L$) and $r_1, r_2, ...r_{M_i}$ are the production rules that construct s_i. The concept and rule sequences are assumed to be unique in the above equations. The parser uses heuristics to comply with this assumption.

SCFG and n-gram probabilities are learned from a text corpus by simple counting and smoothing. Our semantic grammars have a low degree of ambiguity and therefore do not require computationally intensive stochastic training and parsing techniques.

The class based LM can be considered as a very special case of our grammar based model. Concepts (or classes) are restricted to those that represent a list of semantically similar words, like [city_name] , [day_of_week], [month_day] and so forth. So, instead of rule probabilities we have given the class the word probabilities, $P(w_i/c_j)$. For simplicity, each word belongs to at most one class.

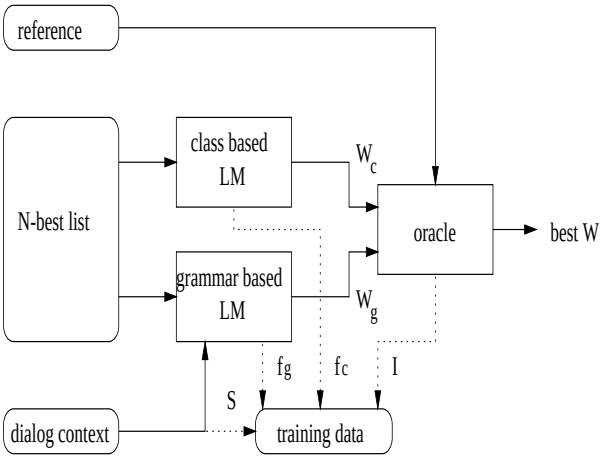

Figure 3: The set up for oracle experiments

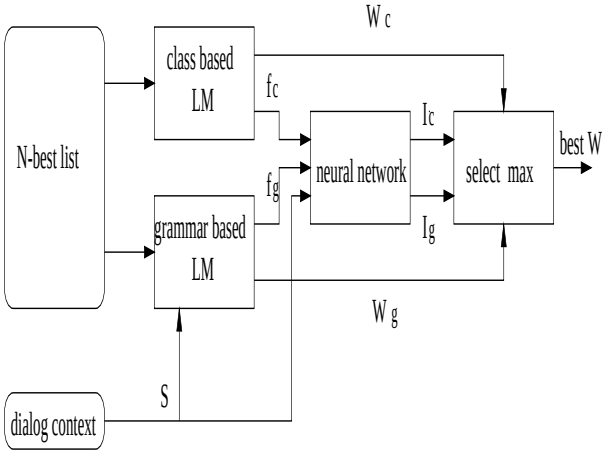

Figure 4: The LM combining system based on the oracle approach.

3. LINEAR AND LOG-LINEAR INTERPOLATION

Assuming that we have M language models, $P_i(W), i = 1, 2, \cdots, M$, the combined LM obtained using the linear interpolation (at sentence level) is given by

$$P(W) = \sum_{i=1}^{M} \lambda_i P_i(W) \tag{2}$$

where λ_i are positive interpolation weights that sum up to unity.

The log-linear interpolation suggests an LM, again at sentence level, given by

$$P(W) = \frac{1}{Z(\lambda)} \prod_{i=1}^{M} P_i(W)^{\lambda_i} \tag{3}$$

where $Z(\lambda)$ is the normalization factor and it is a function of the interpolation weights. The linearity in logarithmic domain is obvious if we take the logarithm of both sides. In the sequel, we omit the normalization term, as its computation is very expensive. We hope that its impact on the performance is not significant. Yet, it prevents us from reporting perplexity results.

4. THE ORACLE APPROACH

The set-up for oracle experiments is illustrated in Figure 3. The purpose of this set-up is twofold. First, we use it to evaluate the oracle performance. Second, we use it to prepare data for the training of a stochastic decision model. For the sake of simplicity, we show the set-up for two LMs and do experiments accordingly. Nonetheless, the set-up can be extended to an arbitrary number of LMs.

The language models are used for N-best list rescoring. The N-best list is generated by a speech recognizer using a relatively simpler LM (here, a class-based trigram LM) . The framework for N-best list rescoring is the following MAP decision:

$$W^* = \underset{W \in L_N}{\text{argmax}} \; p_A P(W/C_W) P(C_W/S) \tag{4}$$

where p_A is the acoustic probability from the first pass, C_W is the unique concept sequence associated with W, and L_N denotes the N-best list. Each rescoring module supplies the oracle with their best hypothesis after rescoring. The oracle compares each hypothesis to the reference and pick the one with the best word (or semantic) accuracy.

For training purposes, we create the input feature vector by augmenting features from each rescoring module (f_g, f_c) and the dialog context (S). The output vector is the LM indicator I from the oracle. The element that corresponds to the LM with the best final hypothesis is unity and the rest are zeros. After training the oracle combiner (here, we assume a neural network), we set our system as shown in Figure 4. The input to the neural network (NN) is the augmented feature vector. The output of the NN is the LM indicator probably with fuzzy values. So, we first pick the max output, and then, we select and output the respective word string.

5. EXPERIMENTAL RESULTS

The models were developed and tested in the context of the CU Communicator dialog system which is used for telephone-based flight, hotel and rental car reservations [11]. The text corpus was divided into two parts as training and test sets with 15220 and 1220 sentences, respectively. The test set was further divided into two parts. Each part, in turn, was used to optimize language and interpolation weights to be used for the other part in a "jacknife paradigm". The results were reported as the average of the two results. The average sentence length of the corpus was 4 words (end-of-sentence was treated as a word). We identified 20 dialog contexts and labeled each sentence with the associated dialog context.

We trained a dialog independent (DI) class based LM and dialog dependent (DD) grammar based LM. In all LMs n is set to 3. It must be noted that the DI class-based LM served as the LM of the baseline system with 921 unigrams including 19 classes. The total number of the distinct words in the lexicon was 1681. The grammar-based LM had 199 concept and filler classes that completely cover the lexicon. In rescoring experiments we set the N-best list size to 10. We think that the choice of $N = 10$ is a resonable tradeoff between performance and complexity.

The perplexity results are presented in Table 1. The perplexity of the grammar-based LM is 36.8% better than the baseline class-based LM.

We did experiments using 10-best lists from the baseline recognizer. We first determined the best possible performance in WER

Table 1: Perplexity results

LM	Perplexity
DI class 3-gram	22.0
DD SCFG 3-gram	13.9

offered by 10-best lists. This is done by picking the hypothesis with the lowest WER from each list. This gives an upperbound for the performance gain possible from rescoring 10-best lists . The rescoring results in terms of absolute and relative improvements in WER and semantic error rate (SER) along with the best possible improvement are reported in Table 2. It should be noted that the optimizations are made using WER. The slight drop in SER with interpolation might be due to that. Actually this is good for text transcription but not for a dialog system. We believe that the results will reverse if we replace the optimization using WER with the optimization using SER.

Table 2: The WER and SER results of the 10-best list rescoring with different LMs: the baseline WER is 25.9% and SER is 23.7%

Method	WER	SER
Class based LM alone	**0.0**%	**0.0**%
Grammar based LM alone	**1.4**(5.4)%	**1.4**(5.9)%
Linear interpolation	**1.6**(6.2)%	**1.3**(5.5)%
Log-linear interpolation	**1.7**(6.6)%	**1.2**(5.1)%
Oracle	**3.0**(11.6)%	**2.7**(11.4) %
Best	**6.4**(24.1)%	**5.5**(23.2)%

The performance gap between the oracle and interpolation methods promotes the system in Figure 4. We expect that, based on the universal approximation theory, a neural network with consistent features, sufficiently large training data and proper training would approximate fairly well the behavior of the oracle. On the other hand, the performance gap between the oracle and the best possible performance from 10-best lists suggests the use of more than two language models and dynamic combination with the acoustic model.

6. CONCLUSIONS

We have presented our recent work on language model combining. We have shown that although a simple interpolation of LMs improves the performance, it fails to reach the performance of an oracle. We have proposed a method for LM combination that mimics the behavior of the oracle. Although our work is not complete without a neural network that mimics the oracle, we argue that the universal approximation theory ensures the success of such a method. However, extensive experiments are required to reach the goal with the main focus on the selection of features. At the moment, the number of concepts, the number of filler classes and the number of 3-gram hits in a sentence (all normalized by the length of the sentence) and the behavior of n-grams in a context are the features that we consider to use. Also, it has been observed that the performance of the oracle is still far from the best possible performance. This is partly due to the very small number of LMs used in the rescoring, partly due to the oracle's hard decision combining strategy and partly due to the static combination with the acoustic model. The work is in progress towards the goal of filling the performance gap.

7. REFERENCES

[1] J. K. Baker. Trainable grammars for speech recognition. In *Speech Communications for th 97th Meeting of the Acoustical Society of America*, pages 31–35, June 1979.

[2] J. Gillett and W. Ward. A language model combining trigrams and stochastic context-free grammars. In *5-th International Conference on Spoken Language Processing*, pages 2319–2322, Sydney, Australia, 1998.

[3] K. Hacioglu and W. Ward. Dialog-context dependent language models combining n-grams and stochastic context-free grammars. In *submitted toInternational Conference of Acoustics, Speech, and Signal Processing*, Salt-Lake, Utah,, 2001.

[4] F. Jelinek and R. Mercer. Interpolated estimation of markov source parameters from sparse data. *Pattern Recognition in Practice*, 23:381, 1980.

[5] A. Keller, B. Rueber, F. Seide, and B. Tran. PADIS - an automatic telephone switchboard and directory information system. *Speech Communication*, 23:95–111, 1997.

[6] D. Klakow. Log-linear interpolation of language models. In *5-th International Conference on Spoken Language Processing*, pages 1695–1699, Sydney, Australia, 1998.

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

[8] R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278, August 2000.

[9] B. Souvignier, A. Keller, B. Rueber, H.Schramm, and F. Seide. The thoughtful elephant: Strategies for spoken dialog systems. *IEEE Transactions on Speech and Audio Processing*, 8(1):51–62, January 2000.

[10] Y. Wang, M. Mahajan, and X.Huang. A unified context-free grammar and n-gram model for spoken language processing. In *International Conference of Acoustics, Speech, and Signal Processing*, pages 1639–1642, Istanbul, Turkey, 2000.

[11] W. Ward and B. Pellom. The CU communicator system. In *IEEE Workshop on Automatic Speech Recognition and Understanding*, Keystone, Colorado, 1999.