OPTIMIZING SPEECH RECOGNITION GRAMMARS USING A MEASURE OF SIMILARITY BETWEEN HIDDEN MARKOV MODELS

Binit Mohanty[1], *John Hershey*[2], *Peder Olsen*[2], *Suleyman Kozat*[2], *Vaibhava Goel*[2]

[1]Center for Language & Speech Processing, Johns Hopkins University, Baltimore, MD
[2]IBM T. J. Watson Research Center, Yorktown Heights, NY

ABSTRACT

In this paper we discuss a method of optimizing weights in a stochastic finite state grammar using a measure of similarity between hidden Markov models. We compute the similarity using an edit distance and weights that are derived from the Bhattacharyya error between pairs of Gaussian mixture models. Forward-backward procedures are used to carry out the similarity computation, and to obtain the derivatives needed in gradient descent based optimization. We apply this procedure to the problem of estimating parameters of garbage models that are often included in SRGS grammars. Experimental results indicate that the method improves the garbage models and naturally results in models that are a function of their context in the grammar.

Index Terms— Stochastic finite state grammars, Garbage models, HMM similarity measures, Bhattacharyya distance, Forward-backward procedures.

1. INTRODUCTION

In spoken dialog systems, stochastic finite state grammars (FSG) are often used for speech recognition at dialog states. In order to handle words and phrases outside the grammar, these grammars typically include some sort of a "garbage" model; grammars written in a speech recognition grammar specification language (SRGS) [1] contain explicit references to a special GARBAGE rule, and grammars written in other proprietary formats contain implicit or explicit references to garbage models.

To determine the optimal model to be used for modeling garbage speech there has been a lot of research, especially in the context of grammar design, key word or phrase spotting, and out-of-vocabulary word modeling [2, 3, 4]. Most of these approaches start with generic word or sub-word based models and optimize them under maximum likelihood or a discriminative criterion using relevant domain data.

In this paper we propose to use an alternative approach that does not require any audio or text data, but instead uses a measure of similarity between HMMs to determine optimal models for garbage speech. These models could then be further optimized once some relevant data is available.

Various methods for computing the similarity or confusability between HMMs have been proposed in the literature [5, 6, 7]. Several applications of HMM similarity have also been proposed in areas such as texture image classification, handwriting recognition and machine learning. In speech recognition, HMM distances have been applied to such tasks as vocabulary selection, grammar design, phoneme clustering, measuring language modeling perplexity, locating occurrences of out-of-vocabulary words in indexed audio databases, matching acoustic tags, and pronunciation variation analysis.

Our proposed method of optimizing garbage models utilizes a Bhattacharyya error based similarity measure proposed recently [6].

2. PROBLEM FORMULATION

Figure 1 panel (a) depicts an example grammar. In this example garbage tags are used at three places: the garbage tags denoted by `garbage*` are expected to match "filler" utterances that callers may speak before or after the core request, and `garbage+` is expected to match "background" caller utterances that are completely out of the core grammar. These garbage models are expected to match arbitrary word strings; `garbage*` is expected to match word strings of length zero or more and `garbage+` is expected to match word strings of length one or more.

The set of all sentences accepted by this grammar can be divided in two potentially overlapping subsets: one where the entire sentence is formed by words from non-garbage portion of the grammar, and another where each sentence has at least one instance of garbage. The first subset corresponding to the grammar of Figure 1 panel (a) is represented as `core.fsm` in Figure 1 panel (b), and the second subset is represented as `diff.fsm` in panel (b).

Once the `core.fsm` and `diff.fsm` subsets are identified, our formulation aims to estimate the parameters of the garbage models to minimize the similarity between these subsets.

3. PARAMETER ESTIMATION

We first briefly review a measure of similarity between two finite state grammars [6]. Following that we discuss our parameter estimation method utilizing this similarity measure.

3.1. Computing Similarity Between Two Finite State Grammars

To compute similarity between two word level grammars, the first step is to expand them to context dependent HMM state level graphs. For the two FSMs of Figure 1 panel (b), we shall use `core.hmm` to denote the state level network obtained from `core.fsm` and `diff.hmm` to denote the corresponding network obtained from `diff.fsm`. The expansion is carried out such that each node in the expanded graphs corresponds to a unique HMM state.

Each HMM state has an associated Gaussian mixture model (GMM). For two states s_1 and s_2, the Bhattacharyya error between their GMMs is used as the measure of similarity between these states. We denote this by $b(s_1, s_2)$. Since exact Bhattacharyya error can not be computed, a variational bound is used instead [6] to approximate $b(s_1, s_2)$.

Let $\mathbf{T_C}$ denote the matrix of transition probabilities for `core.hmm`. If there are m states in this network then this is an $m \times m$ matrix,

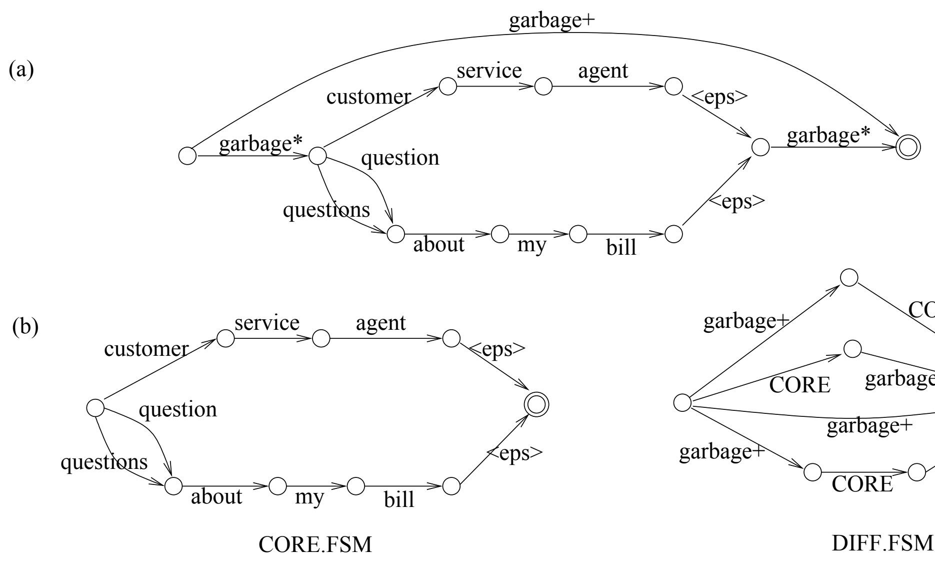

Fig. 1. (a) An example grammar with garbage tags. (b) "core" and "diff" FSMs created from the grammar of panel (a).

which is typically sparse. Let π_C denote the initial state distribution, and f_C denote the vector of exit probabilities from the final state of `core.hmm`. Similarly, let $\mathbf{T_D}$, π_D, and f_D denote the transition probability matrix, initial state distribution, and final state exit probabilities for `diff.hmm`. These are n dimensional matrix and vectors, again typically sparse, where n is the number of states in `diff.hmm`.

Furthermore, let $\mathbf{B}$ denote the $n \times m$ matrix such that the i, j element of $\mathbf{B}$ is the square of similarity between HMM state corresponding to state i of `diff.hmm` and HMM state corresponding to state j of `core.hmm`; $B(i, j) = b^2(i, j)$.

Hershey et. al. [6] show that a lower bound on the Bhattacharyya error (similarity) between distributions specified by `core.hmm` and `diff.hmm` can be computed on the Kronecker product HMM which contains all pairs of state sequences from `diff.hmm` and `core.hmm`. Using $vec(X)$ to denote the column major vector representation of matrix X and $diag(v)$ to denote a diagonal matrix whose diagonal is vector v, the Kronecker product HMM is defined by

$$\begin{aligned} e_I &= vec(\pi_D \otimes \pi_C) \\ e_F &= vec(f_D \otimes f_C) \\ A &= (\mathbf{T_D} \otimes \mathbf{T_C})\, diag(vec(\mathbf{B})) \end{aligned} \tag{1}$$

where $\otimes$ denotes matrix Kronecker product.

Similarity between `core.hmm` and `diff.hmm` is computed as

$$F(C, D) = e_I^T (I + A + A^2 + A^3 + ...)e_F \tag{2}$$

where the i^{th} term in the sum on r.h.s. above, $e_I^T A^i e_F$, represents similarity between all state sequence pairs of length i.

A forward (or backward) procedure can be used to compute similarity of Equation 2, using the following recursion, and using $mat(v)$ to denote the inverse operation of vec for $n \times m$ matrices

$$\begin{aligned} f_n &= e_I^T A^n \\ f_{n+1} &= vec(T_D\, mat(f_n)\, T_C)\, diag(vec(\mathbf{B})) \end{aligned} \tag{3}$$

Our experiments with $F(C, D)$ of Equation 2 showed that it was dominated (exponentially) by the first few terms in the sum; i.e. dominated by short sequences. So the optimization tended to assign higher weight to short words at the expense of longer words, irrespective of the grammars. To alleviate this, we considered a related similarity measure

$$\begin{aligned} G(C, D) &= \log(e_I^T I e_F) + \log(e_I^T A e_F) + \\ &\quad \log(e_I^T A^2 e_F) + ... + \log(e_I^T A^N e_F) \\ &\leq N \log(F(C, D)) - N \log(N) \end{aligned} \tag{4}$$

As the second inequality above shows, $G(C, D)$ provides another approximation to similarity between two HMMs. Using the recursion of Equation 3, this can also be computed using a forward or a backward procedure.

3.2. Model Parameter Optimization

Ideally the parameters of the `garbage` models should be tuned so that they capture most of the "garbage" while not infringing on the actual "core" grammar. It is logical to expect that maximizing the distance(or equivalently minimizing the similarity) between `core.fsm` and `diff.fsm` models should aid in this cause by decreasing the the overlap of garbage words with in-grammar words. However we face the danger of tuning the `garbage` model parameters to the extent of losing its ability to capture the actual garbage words (since we are only trying to decrease its similarity to the in-grammar model). Hence we need to include a penalty term to ensure that we do not over-tune the garbage model.

To capture these considerations, we add a quadratic term to $G(C, D)$ of Equation 4; thus our objective function to minimize is

$$Q(w) = G(C, D, w) + \alpha(w - w_0)'(w - w_0), \tag{5}$$

where w denotes the vector of garbage model parameters that we wish to optimize. w_0 indicates the initial value of these parameters. We add w to $G(C, D)$ to indicate that this similarity is also a function of the garbage model parameters. The quadratic term in the objective function acts as a penalty term, preventing these parameters from getting too far away from the initial value. The parameter α is chosen heuristically, as discussed in the experiments section.

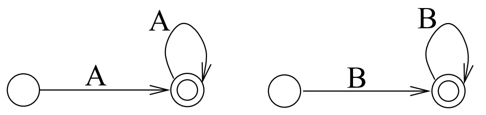

Fig. 2. The fsms to compute similarity between two words A & B

To minimize $Q(w)$, we use a simple gradient descent procedure where we first compute the gradient of the objective function at the current parameter value and then take a small step in the direction of the gradient to reduce the objective function value. The gradient of $G(C, D, w)$ is

$$\frac{\partial G(C, D, w)}{\partial w_i} = \frac{1}{e_I^T A e_F} e_I^T \frac{\partial A}{\partial w_i} e_F + \frac{1}{e_I^T A^2 e_F} e_I^T \frac{\partial A^2}{\partial w_i} e_F + \ldots + \frac{1}{e_I^T A^N e_F} e_I^T \frac{\partial A^N}{\partial w_i} e_F + 2\alpha w_i \quad (6)$$

The gradient can be computed using a forward procedure that utilizes the recursion

$$e_I^T \frac{\partial A^k}{\partial w_i} = e_I^T \frac{\partial A^{k-1}}{\partial w_i} A + e_I^T A^{k-1} \frac{\partial A}{\partial w_i} \quad (7)$$

Once the gradient vector is computed, we do a line search along the gradient. The line search proceeds by choosing the step size along the gradient direction where either a “knee” is observed in the decrease of value of objective function or the lowest value of the objective function is obtained, whichever is earlier within a given upper bound on step size. Furthermore after every step the weight of the parameters are re-normalized.

4. WORD SIMILARITY BASED GARBAGE MODEL OPTIMIZATION

The garbage model is a word unigram constructed from commonly occurring words in English. The word unigram probabilities form the parameters w that we wish to adjust in the garbage model.

A simpler alternative to the garbage model parameter estimation procedure presented in Sections 3.1 and 3.2 is to first compute similarity between words of `core.fsm` and `diff.fsm` and then reduce the unigram probabilities of words of `diff.fsm` that are similar to words in `core.fsm`. In our experiments, we assigned a very low (floor) probability to these confusable words of `garbage` model.

To compute the similarity between a word in `core.fsm` and a word in `garbage` model, we use the HMM similarity procedure of Section 3.1 on the HMMs constructed from FSMs shown in Figure 2. The advantage of the word loop on final state is that we can capture confusability between words of unequal length, by allowing repetitions.

5. EXPERIMENTAL SETUP

The experiments we report in this paper were conducted with a grammar used at a specific dialog module in a conversational self-service system. At this dialog module callers are presented with four choices: they can ask a question about their bill, get their account balance, find the location of a store, or make a payment. Additionally, they can also ask for an operator.

We experimented with two grammar structures: one similar to the example shown in Figure 1, except the core portion is slightly larger: it accepts 59 sentences corresponding to the five request types mentioned above and has a vocabulary of 23 words. The second structure was simpler; it did not have the filler garbage models before and after the core and only had the background `garbage+` model.

All garbage models we use in our experiments consist of the 100 most frequent words in the Fisher [8] corpus. The unigram probabilities of these words in the corpus are taken as the initial value of the garbage model parameters.

The test set consists of 7137 sentences. We partitioned these into three subsets: a) in-grammar or IG: sentences with no filler or background words and belonging completely to the core; b) filler or FL: sentences that have some filler words in addition to a portion that belongs to core; and c) background or BG: all other sentences.

Speech recognition was carried out using an acoustic model built on about 1000 hours acoustic data. The acoustic feature vectors were obtained by first computing 13 Mel-cepstral coefficients (including energy) for each time slice under a 25.0 msec. window with a 15 msec. shift. Nine such vectors were concatenated and projected to a 39 dimensional space using LDA. The acoustic models had 156 context independent states from 52 phones; these states were decision tree clustered to 2198 context dependent states and a total of 167929 Gaussians were used to model these states.

6. RESULTS

6.1. Grammar with Only Background Garbage Model

Our first set of experiments were with grammar structure that only had the core and background `garbage+` model. For this structure, the `diff.fsm` consists only of `garbage+`. The HMM created from `core.fsm` had 684 states, and the HMM created from `diff.fsm` had 1529 states.

The baseline recognition performance, as measured by total false accepts (FA) and false rejects (FR) at optimal threshold level (one that minimizes total FA+FR), was 11.7% for this grammar. The recognition performance on the three test set subsets mentioned above was: 2.6% on IG, 37.7% on FL, and 40.3% on BG.

Experiment	Overall	IG	FL	BG
Baseline	11.7	2.6	37.7	40.3
4 floored	11.7	2.6	34.2	41.5
8 floored	11.5	2.4	33.2	41.6
12 floored	11.5	2.4	31.6	42.2
16 floored	11.5	2.4	31.6	42.0

Table 1. Background garbage only grammar: FA + FR values for garbage model optimization based on word confusabilities

We first evaluated the word similarity based garbage model optimization described in Section 4. We chose to floor the probability of the n most confusable words. Table 1 shows the false accept + false reject rates on our test set for $n = 4, 8, 12, 16$.

Next we evaluated the performance of the garbage model optimization procedure described in Section 3. Table 2 shows FA+FR results for various values of the quadratic term weight α (Equation 5). For each α, 5 iterations of the gradient descent procedure described in Section 3 were carried out. For the largest best performing value, $\alpha = 1e8$, five more iterations (total 10) were carried out, as indicated in the table.

From Tables 1 and 2, we note that with either garbage model optimization method there is a small gain in overall FA+FR. The gain is relatively larger (7% relative) on the IG portion of the test set.

Comparing numbers of Table 1 with those in Table 2, we note that similar results are obtained by the two methods. It appears that using gradient descent for the case where only the background garbage model is used performs no better then just flooring words in the `garbage` model based on individual word-pair confusability scores. This can be explained by the fact that the fsm model for this experiment allows a sentence to be recognized as either coming from `core` grammar only or from the background `garbage+` only. Hence we expect that the words in `garbage+` model which are most similar to words in core to cause potential mislabeling of the sentence. And so reducing the weights of these similar words leads to better recognition by the `core` model.

6.2. Grammar with Background & Filler Garbage Models

In the next set of experiments we used the grammar with filler and background garbage models. The baseline FA+FR performance of this grammar was 9.4%. On the three subsets the performance was: 4.2% on IG, 12.5% on FL, and 30.4% on BG.

Table 3 shows the results of word similarity based probability flooring experiments. Since the same garbage model was used for two filler instances of garbage and one background instance of garbage, the total number of floored words is a multiple of 3.

Results of gradient optimization of the two filler and one background garbage model are shown in Table 4. These results are obtained using $\alpha = 1e8$ as that was found to be the optimal α in Section 6.1. For each iteration of the optimization we chose a step size that floored 4 variables.

We observe that downweighting the acoustically similar words leads to a marginal degradation in the error rates. Since the current model has context dependent `garbage` model - we have no reason to believe that just downweighting acoustically similar words would result in any better performance. However the results for the gradient descent method are not conclusive. The numbers are only marginally different from baseline. However, we note that an optimal penalty parameter α is not heuristically chosen as in section 6.1.

Experiment	Overall	IG	FL	BG
$\alpha = 0.1e8$	11.5	2.5	32.4	41.7
$\alpha = 0.5e8$	11.5	2.5	32.4	41.7
$\alpha = 1e8$	11.5	2.5	32.4	41.7
$\alpha = 1e8$(10 iter)	11.5	2.4	31.4	42.2
$\alpha = 5e8$	11.6	2.5	32.4	42.2

Table 2. Background garbage only grammar: FA + FR of gradient optimization for various α values

Experiment	Overall	IG	FL	BG
Baseline	9.4	4.2	12.5	30.4
3 floored	9.4	4.2	12.5	30.3
6 floored	9.5	4.1	12.3	31.3
9 floored	9.5	4.2	12.1	31.2
12 floored	9.5	4.2	11.9	31.1

Table 3. Background & filler garbage: FA + FR values for garbage model optimization based on word confusabilities

Experiment	Overall	IG	FL	BG
iter 1 (4 floored)	9.4	4.1	12.3	30.7
iter 2 (8 floored)	9.3	4.2	12.3	29.8
iter 3 (12 floored)	9.3	4.3	12.9	29.3

Table 4. Background & filler garbage: FA + FR for iterations of gradient optimization ($\alpha = 1e8$)

7. CONCLUSIONS

In this paper we have proposed a method for tuning garbage models without using any training data. This method uses gradient descent based on a distance measure between HMMs to tune the garbage model parameters. We also propose a tuning method based on downweighting of individual words in the garbage model that are acoustically similar with words in the core grammar.

We present experiments with two different grammar structures, one involving only the background garbage model and another involving both background and filler garbage models. The results indicate a small improvement in the case of background garbage model using both gradient descent as well as downweighting of individual words in the garbage model. As expected there is a relatively larger improvement in the error rates for in-grammar sentences compared to the overall improvement. The experiments with the model containing both filler and background garbage models are, however, less conclusive. While acoustic confusability based downweighting seems to marginally degrade performance, the gradient descent based method gives a very small improvement but does not perform much better either. Though it should be noted that we do not optimize for α, the quadratic penalty parameter in this case.

The method described in this paper does not require any training data and uses simple gradient descent to tune parameters of an HMM. While no claims of optimality are made, the proposed method does give a means of tuning an HMM model directly based on the distance between competing models and in that respect it is general enough to encompass tuning of any HMM model.

8. REFERENCES

[1] W3C Speech Recognition Grammar Specification Version 1.0. http://www.w3.org/TR/speech-grammar/. 2004.

[2] C. Ma and C.-H. Lee, "A study on word detector design and knowledge-based pruning and rescoring," in *Proc. Interspeech*, 2007.

[3] I. Bazzi and J. Glass, "Modeling out-of-vocabulary words for robust speech recognition," in *Proc. ICSLP*, 2000.

[4] A. Manos and V. Zue, "A segment-based wordspotter using phonetic filler models," in *Proc. ICASSP*, 1997.

[5] P. Olsen and J. Hershey, "Bhattacharyya error and divergence using variational importance sampling," in *Proceedings of Interspeech 2007*, August 2007.

[6] J. Hershey and P. Olsen, "Bhattacharyya error and divergence between two hidden Markov models," in *submitted to ICASSP 2008*, 2008.

[7] B.-H. Juang and L. R. Rabiner, "A probabilistic distance measure for hidden Markov models," *AT&T Technical Journal*, vol. 64, no. 2, pp. 391–408, February 1985.

[8] C. Cieri, D. Miller, and K. Walker, "From Switchboard to Fisher: Telephone Collection Protocols, their Uses and Yields," in *Proc. Eurospeech*, 2003.