POS Tags and Decision Trees for Language Modeling

Peter A. Heeman
Department of Computer Science and Engineering
Oregon Graduate Institute
PO Box 91000, Portland OR 97291
heeman@cse.ogi.edu

Abstract

Language models for speech recognition concentrate solely on recognizing the words that were spoken. In this paper, we advocate redefining the speech recognition problem so that its goal is to find both the best sequence of words and their POS tags, and thus incorporate POS tagging. To use POS tags effectively, we use clustering and decision tree algorithms, which allow generalizations between POS tags and words to be effectively used in estimating the probability distributions. We show that our POS model gives a reduction in word error rate and perplexity for the Trains corpus in comparison to word and class-based approaches. By using the Wall Street Journal corpus, we show that this approach scales up when more training data is available.

1 Introduction

For recognizing spontaneous speech, the acoustic signal is to weak to narrow down the number of word candidates. Hence, recognizers employ a language model to take into account the likelihood of word sequences. To do this, the recognition problem is cast as finding the most likely word sequence $\hat{W}$ given the acoustic signal A (Jelinek, 1985).

$$\begin{aligned}\hat{W} &= \arg\max_W \Pr(W|A) \\ &= \arg\max_W \frac{\Pr(A|W)\Pr(W)}{\Pr(A)} \\ &= \arg\max_W \Pr(A|W)\Pr(W) \qquad (1)\end{aligned}$$

The last line involves two probabilities that need to be estimated—the first due to the acoustic model $\Pr(A|W)$ and the second due to the language model $\Pr(W)$. The language model probability can be expressed as follows, where we rewrite the sequence W explicitly as the sequence of N words $W_{1,N}$.

$$\Pr(W_{1,N}) = \prod_{i=1}^{N} \Pr(W_i|W_{1,i-1}) \qquad (2)$$

To estimate the probability distribution $\Pr(W_i|W_{1,i-1})$, a training corpus is used to determine the relative frequencies. Due to sparseness of data, one must define *equivalence classes* amongst the contexts $W_{1,i-1}$, which can be done by limiting the context to an n-gram language model (Jelinek, 1985). One can also mix in smaller size language models when there is not enough data to support the larger context by using either interpolated estimation (Jelinek and Mercer, 1980) or a backoff approach (Katz, 1987). A way of measuring the effectiveness of the estimated probability distribution is to measure the *perplexity* that it assigns to a test corpus (Bahl et al., 1977). Perplexity is an estimate of how well the language model is able to predict the next word of a test corpus in terms of the number of alternatives that need to be considered at each point. The perplexity of a test set $w_{1,N}$ is calculated as 2^H, where H is the entropy, defined as follows.

$$H = -\frac{1}{N}\sum_{i=1}^{N} \log_2 \hat{\Pr}(w_i|w_{1,i-1}) \qquad (3)$$

1.1 Class-based Language Models

The choice of equivalence classes for a language model need not be the previous words. Words can be grouped into classes, and these classes can be used as the basis of the equivalence classes of the context rather than the word

identities (Jelinek, 1985). Below we give the equation usually used for a class-based trigram model, where the function g maps each word to its unambiguous class.

$$\begin{aligned}&\Pr(W_i|W_{1,i-1}) \\ &\approx \Pr(W_i|g(W_i))\Pr(g(W_i)|g(W_{i-1})g(W_{i-2}))\end{aligned}$$

Using classes has the potential of reducing the problem of sparseness of data by allowing generalizations over similar words, as well as reducing the size of the language model.

To determine the word classes, one can use the algorithm of Brown *et al.* (1992), which finds the classes that give high mutual information between the classes of adjacent words. In other words, for each bigram $w_{i-1}w_i$ in a training corpus, choose the classes such that the classes for adjacent words $g(w_{i-1})$ and $g(w_i)$ lose as little information about each other as possible. Brown *et al.* give a greedy algorithm for finding the classes. They start with each word in a separate class and iteratively combine classes that lead to the smallest decrease in mutual information between adjacent words. Kneser and Ney (1993) found that a class-based language model results in a perplexity improvement for the LOB corpus from 541 for a word-based bigram model to 478 for a class-based bigram model. Interpolating the word-based and class-based models resulted in an improvement to 439.

1.2 Previous POS-Based Models

One can also use POS tags, which capture the syntactic role of each word, as the basis of the equivalence classes (Jelinek, 1985). Consider the utterances "load the oranges" and "the load of bananas". The word "load" is being used as an untensed verb in the first example, and as a noun in the second; and "oranges" and "bananas" are both being used as plural nouns. The POS tag of a word is influenced by, and influences the neighboring words and their POS tags. To use POS tags in language modeling, the typical approach is to sum over all of the POS possibilities. Below, we give the derivation based on using trigrams.

$$\begin{aligned}&\Pr(W_{1,N}) \\ &= \sum_{P_{1,N}} \Pr(W_{1,N}P_{1,N}) \\ &= \sum_{P_{1,N}} \prod_{i=1}^{N} \Pr(W_i|W_{1,i-1}P_{1,i})\Pr(P_i|W_{1,i-1}P_{1,i-1}) \\ &\approx \sum_{P_{1,N}} \prod_{i=1}^{N} \Pr(W_i|P_i)\Pr(P_i|P_{1,i-1}) \qquad (4) \\ &\approx \sum_{P_{1,N}} \prod_{i=1}^{N} \Pr(W_i|P_i)\Pr(P_i|P_{i-2,i-1}) \qquad (5)\end{aligned}$$

Note that line 4 involves some simplifying assumptions; namely, that $\Pr(W_i|W_{1,i-1}P_{1,i})$ can be approximated by $\Pr(W_i|P_i)$ and that $\Pr(P_i|W_{1,i-1}P_{1,i-1})$ can be approximated by $\Pr(P_i|P_{1,i-1})$. These assumptions simplify the task of estimating the probability distributions. Relative frequency can be used directly for estimating the word probabilities, and trigram backoff and linear interpolation can be used for estimating the POS probabilities.

The above approach for incorporating POS information into a language model has not been of much success in improving speech recognition performance. Srinivas (1996) reported a 24.5% increase in perplexity over a word-based model on the Wall Street Journal; Niesler and Woodland (1996) reported an 11.3% increase (but a 22-fold decrease in the number of parameters of such a model) for the LOB corpus; and Kneser and Ney (1993) report a 3% increase on the LOB corpus. The POS tags remove too much of the lexical information that is necessary for predicting the next word. Only by interpolating it with a word-based model is an improvement seen (Jelinek, 1985).

1.3 Our Approach

In past work (Heeman and Allen, 1997; Heeman, 1998), we introduced an alternative formulation for using POS tags in a language model. Here, POS tags are elevated from intermediate objects to be part of the output of the speech recognizer. Furthermore, we do not use the simplifying assumptions of the previous approach. Rather, we use a clustering algorithm to find words and POS tags that behave similarly. The output of the clustering algorithm is used by a decision tree algorithm to build a

set of equivalence classes of the contexts from which the word and POS probabilities are estimated.

In this paper, we show that the perplexity reduction that we previous reported using our POS-based model on the Trains corpus does translate into a word error rate reduction. The Trains corpus is very small with only 58,000 words of data. Hence, we also report on perplexity results using much larger amounts of training data, as afforded by using the Wall Street Journal corpus. We discuss how we take advantage of the POS tags to both improve and expedite the clustering and decision tree algorithms.

2 Redefining the Problem

To add POS tags into the language model, we refrain from simply summing over all POS sequences as prior approaches have done. Instead, we redefine the speech recognition problem so that it finds the best word and POS sequence. Let P be a POS sequence for the word sequence W. The goal of the speech recognizer is to now solve the following.

$$\begin{aligned}\hat{W}\hat{P} &= \arg\max_{WP} \Pr(WP|A) \\ &= \arg\max_{WP} \frac{\Pr(A|WP)\Pr(WP)}{\Pr(A)} \\ &= \arg\max_{WP} \Pr(A|WP)\Pr(WP) \quad (6)\end{aligned}$$

The first term $\Pr(A|WP)$ is the acoustic model, which traditionally excludes the category assignment. In fact, the acoustic model can probably be reasonably approximated by $\Pr(A|W)$. The second term $\Pr(WP)$ is the POS-based language model and accounts for both the sequence of words and their POS assignment. We rewrite the sequence WP explicitly in terms of the N words and their corresponding POS tags, thus giving the sequence $W_{1,N}P_{1,N}$. The probability $\Pr(W_{1,N}P_{1,N})$ forms the basis for POS taggers, with the exception that POS taggers work from a sequence of given words.

As in Equation 2, we rewrite $\Pr(W_{1,N}P_{1,N})$ using the definition of conditional probability.

$$\begin{aligned}&\Pr(W_{1,N}P_{1,N}) \\ &= \prod_{i=1}^{N} \Pr(W_iP_i|W_{1,i-1}P_{1,i-1}) \\ &= \prod_{i=1}^{N} \Pr(W_i|W_{1,i-1}P_{1,i}) \Pr(P_i|W_{1,i-1}P_{1,i-1}) \quad (7)\end{aligned}$$

Equation 7 involves two probability distributions that need to be estimated. Previous attempts at using POS tags in a language model as well as POS taggers (i.e. (Charniak et al., 1993)) simplify these probability distributions, as given in Equations 8 and 9.

$$\Pr(W_i|W_{1,i-1}P_{1,i}) \approx \Pr(W_i|P_i) \quad (8)$$

$$\Pr(P_i|W_{1,i-1}P_{1,i-1}) \approx \Pr(P_i|P_{1,i-1}) \quad (9)$$

However, to successfully incorporate POS information, we need to account for the full richness of the probability distributions. Hence, as we will show in Table 1, we cannot use these two assumptions when learning the probability distributions.

3 Estimating the Probabilities

To estimate the probability distributions, we follow the approach of Bahl *et al.* (1989) and use a decision tree learning algorithm (Breiman et al., 1984) to partition the context into equivalence classes.

3.1 POS Probabilities

For estimating the POS probability distribution, the algorithm starts with a single node with all of the training data. It then finds a question to ask about the POS tags and word identities of the preceding words ($P_{1,i-1}W_{1,i-1}$) in order to partition the node into two *leaves*, each being more informative as to which POS tag occurred than the parent node. Information theoretic metrics, such as minimizing entropy, are used to decide which question to propose. The proposed question is then verified using heldout data: if the split does not lead to a decrease in entropy according to the heldout data, the split is rejected and the node is not further explored (Bahl et al., 1989). This process continues with the new leaves and results in a hierarchical partitioning of the context.

After growing a tree, the next step is to use the partitioning of the context induced by the

decision tree to determine the probability estimates. Using the relative frequencies in each node will be biased towards the training data that was used in choosing the questions. Hence, Bahl *et al.* smooth these probabilities with the probabilities of the parent node using interpolated estimation with a second heldout dataset.

Using the decision tree algorithm to estimate probabilities is attractive since the algorithm can choose which parts of the context are relevant, and in what order. Hence, this approach lends itself more readily to allowing extra contextual information to be included, such as both the word identifies and POS tags, and even hierarchical clusterings of them. If the extra information is not relevant, it will not be used.

3.2 Word Probabilities

The procedure for estimating the word probability is almost identical to the above. However, rather than start with all of the training data in a single node, we first partition the data by the POS tag of the word being estimated. Hence, we start with the probability $\Pr(W_i|P_i)$ as estimated by relative frequency. This is the same value with which non-decision tree approaches start (and end). We then use the decision tree algorithm to further refine the equivalence contexts by allowing it to ask questions about the preceding words and POS tags.

Starting the decision tree algorithm with a separate root node for each POS tag has the following advantages. Words only take on a small set of POS tags. For instance, a word that is a superlative adjective cannot be a relative adjective. For the Wall Street Journal, each token on average takes on 1.22 of the 46 POS tags. If we start with all training data in a single root node, the smoothing (no matter how small) will end up putting some probability for each word occurring as every POS tag, leading to less exact probability estimates. Second, if we start with a root node for each POS tag, the number of words that need to be distinguished at each node in the tree is much less than the full vocabulary size. For the Wall Street Journal corpus, there are approximately 42,700 different words in the training data, but the most common POS tag, proper nouns (**NNP**), only has 12,000 different words. Other POS tags have much fewer, such as the personal pronouns with only 36 words. Making use of this smaller vocabulary size results in a faster algorithm and less memory space.

A significant number of words in the training corpus have a small number of occurrences. Such words will prove problematic for the decision tree algorithm to predict. For each POS tag, we group the low occurring words into a single token for the decision tree to predict. This not only leads to better probability estimates, but also reduces the number of parameters in the decision tree. For the Wall Street Journal corpus, excluding words that occur less than five times reduces the vocabulary size to 14,000 and the number of proper nouns to 3126.

3.3 Questions about POS Tags

The context that we use for estimating the probabilities includes both word identities and POS tags. To make effective use of this information, we need to allow the decision tree algorithm to generalize between words and POS tags that behave similarly. To learn which words behave similarly, Black *et al.*(1989) and Magerman (1994) used the clustering algorithm of Brown *et al.* (1992) to build a hierarchical classification tree. Figure 1 gives the classification tree that we built for the POS tags from the Trains corpus. The algorithm starts with each token in a separate class and iteratively finds two classes to merge that results in the smallest lost of information about POS adjacency. Rather than stopping at a certain number of classes, one continues until only a single class remains. However, the order in which classes were merged gives a hierarchical binary tree with the root corresponding to the entire tagset, each leaf to a single POS tag, and intermediate nodes to groupings of tags that occur in statistically similar contexts. The path from the root to a tag gives the binary encoding for the tag. The decision tree algorithm can ask which partition a word belongs to by asking questions about the binary encoding. Of course it doesn't make sense to ask questions about the bits before the higher level bits are asked about. But we do allow it to ask complex bit encoding questions so that it can find more optimal ques-

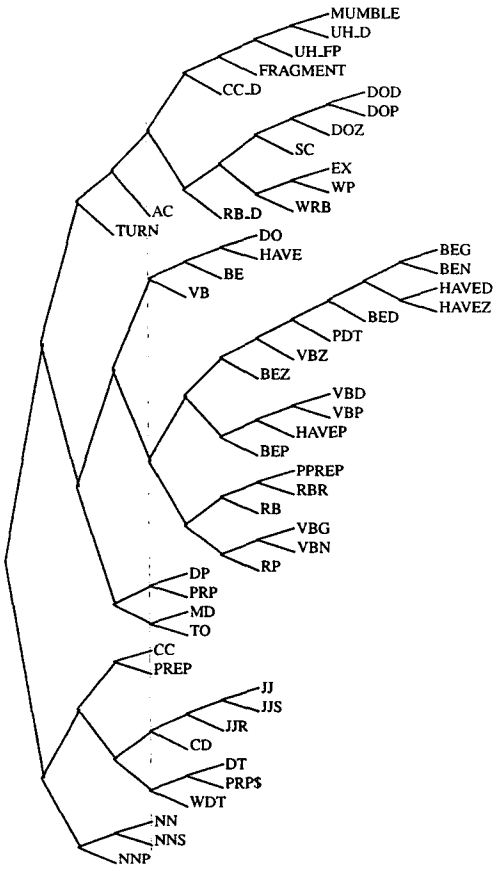

Figure 1: Classification Tree for POS Tags

tions (Heeman, 1997).

3.4 Questions about Word Identities

For handling word identities, one could follow the approach used for handling the POS tags (e.g. (Black et al., 1992; Magerman, 1994)) and view the POS tags and word identities as two separate sources of information. Instead, we view the word identities as a further refinement of the POS tags. We start the clustering algorithm with a separate class for each word and each POS tag that it takes on and only allow it to merge classes if the POS tags are the same. This results in a word classification tree for each POS tag. Using POS tags in word clustering means that words that take on different POS tags can be better modeled (Heeman, 1997). For instance, the word "load" can be used as a verb (**VB**) or as a noun (**NN**), and this usage affects with which words it is similar. Furthermore, restricting merges to those of

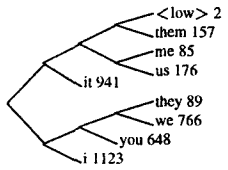

Figure 2: Classification Tree for Personal Pronouns

the same POS tag allows us to make use of the hand-annotated linguistic knowledge for clustering words, which allows more effective trees to be built. It also significantly speeds up the clustering algorithm. For the Wall Street Journal, only 13% of all merges are between words of the same POS tag, and hence do not need to be considered.

To deal with low occurring words in the training data, we follow the same approach as we do in in building the classification tree. We group all words that occur less than some freshhold into a single token for each POS tag before clustering. This not only significantly reduces the input size to the clustering algorithm, but also relieves the clustering algorithm from trying to statistically cluster words for which there is not enough training data. Since low occurring words are grouped by POS tag, we have better handling of this data than if all low occuring words were grouped into a single token.

Figure 2 shows the classification tree for the personal pronouns (**PRP**) from the Trains corpus. For reference, we list the number of occurrences of each word. Notice that the algorithm distinguished between the subjective pronouns "I", "we", and "they", and the objective pronouns "me", "us" and "them". The pronouns "you" and "it" take both cases and were probably clustered according to their most common usage in the corpus. Although we could have added extra POS tags to distinguish between these two types of pronouns, it seems that the clustering algorithm can make up for some of the shortcomings of the POS tagset.

Since words are viewed as a further refinement of POS information, we restrict the decision tree algorithm from asking about the word identity until the POS tag of the word is uniquely identified. We also restrict the deci-

sion tree from asking more specific bit questions until the less specific bits are unquely determined.

4 Results on Trains Corpus

We ran our first set of experiments on the Trains corpus, a corpus of human-human task oriented dialogs (Heeman and Allen, 1995).

4.1 Experimental Setup

To make the best use of the limited size of the Trains corpus, we used a six-fold cross-validation procedure: each sixth of the data was tested using the rest of the data for training. This was done for both acoustic and language models. Dialogs for each pair of speakers were distributed as evenly between the six partitions in order to minimize the new speaker problem.

For our perplexity results, we ran the experiments on the hand-collected transcripts. Changes in speaker are marked in the word transcription with the token <**turn**>. Contractions, such as "that'll" and "gonna", are treated as separate words: "that" and "'ll" for the first example, and "going" and "ta" for the second. All word fragments were changed to the token <**fragment**>. In searching for the best sequence of POS tags for the transcribed words, we follow the technique proposed by Chow and Schwartz (1989) and only keep a small number of alternative paths by pruning the low probability paths after processing each word.

For our speech recognition results, we used OGI's large vocabulary speech recognizer (Yan et al., 1998; Wu et al., 1999), using acoustic models trained from the Trains corpus. We ran the decoder in a single pass using cross-word acoustic modeling and a trigram word-based backoff model (Katz, 1987) built with the CMU toolkit (Rosenfeld, 1995). For the first pass, contracted words were treated as single tokens in order to improve acoustic recognition of them. The result of the first pass was a word graph, which we rescored in a second pass using our other trigram language models.

4.2 Comparison with Word-Based Model

Column two of Table 1 gives the results of the word-based backoff model and column three gives the results of our POS-based model. Both models were restricted to only looking at the previous two words (and POS tags) in the context, and hence are trigram models. Our POS-based model gives a perplexity reduction of 8.9% and an absolute word error rate reduction of 1.1%, which was found significant by the Wilcoxon test on the 34 different speakers in the Trains corpus (Z-score of -4.64). The POS-based model also achieves an absolute sentence error rate reduction of 1.3%, which was found significant by the McNemar test.

	Word Backoff	Full Context	Simple Content
POS Errors	–	1573	1718
POS Error Rate	–	2.69	2.94
Word Perplexity	24.8	22.6	42.4
Word Error Rate	26.0	24.9	28.9
Sentence Error Rate	56.6	55.2	58.1

Table 1: Comparison with Word-Based Model

One reason for the good performance of our POS-based model is that we use all of the information in the context in estimating the word and POS probabilities. To show this effect, we contrast the results of our model, which uses the full context, with the results given in column four of a model that uses the simpler context afforded by the approximations given in Equation 8 and 9, which ignore word co-occurence information. This simpler model uses the same decision tree techniques to estimate the probability distributions, but the decision tree can only ask questions of the simpler context, rather than the full context. In terms of POS tagging results, we see that using the full context leads to a POS error rate reduction of 8.4%.[1] But more importantly, using the full context gives a 46.7% reduction in perplexity, and a 4.0% absolute reduction in the word error rate. In fact, the simpler model does not even perform as well as the word-based model. Hence, to use POS tags in speech recognition, one must use a richer context for estimating the probabilities than what has been traditionally used, and must properly account for co-occurence information.

[1] POS errors were calculated by running both models against the actual transcripts, in the same way that perplexity is calculated.

4.3 Other Decision Tree Models

The differences between our POS-based model and the backoff word-based model are partially due to the extra power of the decision tree approach in estimating the probabilities. To factor out this difference, we compare our POS-based model to word and class-based models built using our decision tree approach for estimating the probabilities. For the word-based model, we treated all words as having the same POS tag and hence built a trivial POS classification tree and a single word hierarchical classification tree, and then estimated the probabilities using our decision tree algorithm.

We also built a class-based model to test out if a model with automatically learned unambiguous classes could perform as well as our POS-based model. The classes were obtained from our word clustering algorithm, but stopping once a certain number of classes has been reached. Unfortunately, the clustering algorithm of Brown *et al.* does not have a mechanism to decide an optimal number of word classes (cf. (Kneser and Ney, 1993)). Hence, to give an optimal evaluation of the class-based approach, we chose the number of classes that gave the best word error rate, which was 30 classes. We then used this class-assignment instead of the POS tags, and used our existing algorithms to build our decision tree models.

The results of the three decision tree models are given in Table 2, along with the results from the backoff word-based model. First, our

	Backoff	Decision Tree		
	Word	Word	Class	POS
Word Perplexity	24.8	23.7	23.4	22.6
Word Error Rate	26.0	25.5	25.4	24.9
Sentence Error Rate	56.6	55.8	55.6	55.2

Table 2: POS, Class and Word-Based Models

word-based decision tree model outperforms the word backoff model, giving an absolute word-error rate reduction of 0.5%, which was found significant by the Wilcoxon test (Z-score -3.26). Hence, some of the improvement of our POS-based model is because we use decision trees with word clustering to estimate the probabilities. Second, there is little improvement from using unambiguous word classes. This is because we are already using a word hierarchical classification tree, which allows the decision tree algorithm to make generalizations between words, in the same way that classes do (which explains for why so few classes gives the optimal word error rate). Third, using POS tags does lead to an improvement over the class-based model, with an absolute reduction in word error rate of 0.5%, an improvement found significant by the Wilcoxon test (Z-score -2.73). Hence, using shallow syntactic information, in the form of POS tags, does improve speech recognition since it allows syntactic knowledge to be used in predicting the subsequent words. This syntactic knowledge is also used to advantage in building the classification trees, since we can use the hand-coded knowledge present in the POS tags in our classification and we can better classify words that can be used in different ways.

5 Results on Wall Street Journal

In order to show that our model scales up to larger training data sizes and larger vocabulary sizes, we ran perplexity experiments on the Wall Street Journal corpus in the Penn Treebank, which is annotated with POS tags. We used one-eighth of the corpus as our test set, and the rest for training.

Figure 3 gives the results of varying the amount of training data from approximately 45,000 words up to 1.1 million words. We show both the perplexity of the POS-based model and the word-based backoff model.[2] We

[2]The perplexity measure only includes words known in the training data. As the training data size increases,

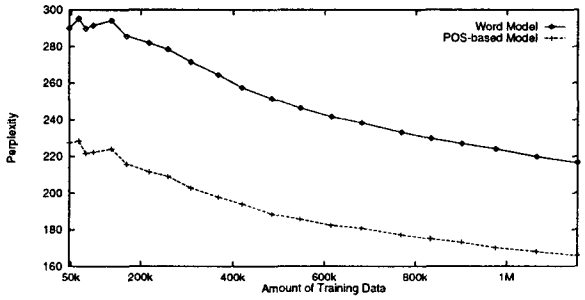

Figure 3: Wall Street Journal Results

see that the POS-based model shows a consistent perplexity reduction over the word-based model. When using all of the available training data, the POS-based model achieves a perplexity rate of 165.9, in comparison to 216.6 for the word-based backoff model, an improvement of 23.4%.

For the POS-based model, all word–POS combinations that occurred less than five times in the training data were grouped together for clustering the words and for building the decision tree. Thus, we built the word classification tree using 14,000 word/POS tokens, rather than the full set of 52,100 that occurred in the training data. Furthermore, the decision tree algorithm was not allowed to split a leaf with less than 6 datapoints. This gave us 103,000 leaf nodes (contexts) for the word tree, each with an average of 1277 probabilities, and 111,000 leaf nodes for the POS tree, each with 47 probabilities, for a total of 136 million parameters. In contrast, the word-based model was composed of 795K trigrams, 376K bigrams, and 43K unigrams and used a total of 2.8 million parameters.[3]

In the above, we compared our decision-tree based approach against the backoff approach. Although our approach gives a 23.4% reduction in perplexity, it also gives a 49-fold increase in the size of the language model. We have done some preliminary experiments in reducing the model size. The word and POS trees can be reduced by decreasing the number of leaf nodes. The word decision tree can also be reduced by decreasing the number of probablities in each leaf, which can be done by increasing the number of words put into the low-occurring group. We built a language model using our decision tree approach that uses only 2.8 million parameters by grouping all words that occur 40 times or fewer into the low occurring class, disallowing nodes to be split if they have 50 or fewer datapoints, and pruning back nodes that give the smallest improvement in node impurity. The resulting word tree has 13,700 leaf nodes, each with an average of 80 probabilities, and the POS tree has 12,800 leaf nodes, each with 47 probabilities. This model achieves a perplexity of 191.7, which is still a 11.5% improvement over the word backoff approach. Hence, even for the same model size, the decision tree approach gives a perplexity reduction over the word backoff approach.[4]

6 Conclusion

Unlike previous approaches that use POS tags, we redefined the speech recognition problem so that it includes finding the best word sequence and best POS tag interpretation for those words. Thus this work can be seen as a first-step towards tightening the integration between speech recognition and natural language processing.

In order to estimate the probabilities of our POS-based model, we use standard algorithms for clustering and growing decision trees; however, we have modified these algorithms to better use the POS information. The POS-based model results in a reduction in perplexity and in word error rate in comparison to a word-based backoff approach. Part of this improvement is due to the decision tree approach for estimating the probabilities.

7 Acknowledgments

We wish to thank James Allen, Geraldine Damnati, Chaojun Liu, Xintian Wu, and Yonghong Yan. This research work was partially supported by NSF under grant IRI-9623665, by the Intel Research Council, and by CNET France Télécom, while the author was visiting there.

the vocabulary increases from approximately 7500 to 42,700. Hence, fewer words of the test data are being excluded from the perplexity measure.

[3]The count of 2.8 million parameters includes 795K trigram probabilities, 376K bigram probabilities, 376K bigram backoff weights, 43K unigram probabilities and 43K unigram backoff weights. Since the trigrams and bigrams are sparse, we include 795K to indicate which trigrams are included, and 376K to indicate which bigrams are included.

[4]We compared a word-backoff model that does not exclude any trigrams or bigrams based on thresholds. Hence, the word-backoff approach can produce much smaller models. We still need to contrast our decision tree approach with smaller backoff models.

References

L. Bahl, J. Baker, F. Jelinek, and R. Mercer. 1977. Perplexity—a measure of the difficulty of speech recognition tasks. In *Proceedings of the 94th Meeting of the Acoustical Society of America.*

L. Bahl, P. Brown, P. deSouza, and R. Mercer. 1989. A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(7):1001–1008.

E. Black, F. Jelinek, J. Lafferty, D. Magerman, R. Mercer, and S. Roukos. 1992. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pg. 134–139.

L. Breiman, J. Friedman, R. Olshen, and C. Stone. 1984. *Classification and Regression Trees.* Wadsworth & Brooks.

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

E. Charniak, C. Hendrickson, N. Jacobson, and M. Perkowitz. 1993. Equations for part-of-speech tagging. In *Proceedings of the National Conference on Artificial Intelligence.*

Y. Chow and R. Schwartz. 1989. The n-best algorithm: An efficient procedure for finding top n sentence hypotheses. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pg. 199–202.

P. Heeman and J. Allen. 1995. The Trains spoken dialog corpus. CD-ROM, Linguistics Data Consortium.

P. Heeman and J. Allen. 1997. Incorporating POS tagging into language modeling. In *Proceedings of the 5th European Conference on Speech Communication and Technology*, pg. 2767–2770, Rhodes, Greece.

P. Heeman. 1997. Speech repairs, intonational boundaries and discourse markers: Modeling speakers' utterances in spoken dialog. Technical Report 673, Department of Computer Science, University of Rochester. Doctoral dissertation.

P. Heeman. 1998. POS tagging versus classes in language modeling. In *Sixth Workshop on Very Large Corpora*, pg. 179–187, Montreal.

F. Jelinek and R. Mercer. 1980. Interpolated estimation of markov source parameters from sparse data. In *Proceedings, Workshop on Pattern Recognition in Practice*, pg. 381–397, Amsterdam.

F. Jelinek. 1985. Self-organized language modeling for speech recognition. Technical report, IBM T.J. Watson Research Center, Continuous Speech Recognition Group.

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

R. Kneser and H. Ney. 1993. Improved clustering techniques for class-based statistical language modelling. In *Proceedings of the 3rd European Conference on Speech Communication and Technology*, pg. 973–976.

D. Magerman. 1994. Natural language parsing as statistical pattern recognition. Doctoral dissertation, Stanford University.

T. Niesler and P. Woodland. 1996. A variable-length category-based n-gram language model. In *Proceedings of the International Conference on Audio, Speech and Signal Processing (ICASSP)*, pg. 164–167.

R. Rosenfeld. 1995. The CMU statistical language modeling toolkit and its use in the 1994 ARPA CSR evaluation. In *Proceedings of the ARPA Spoken Language Systems Technology Workshop.*

B. Srinivas. 1996. "Almost parsing" techniques for language modeling. In *Proceedings of the 4th International Conference on Spoken Language Processing*, pg. 1169–1172.

X. Wu, C. Liu, Y. Yan, D. Kim, S. Cameron, and R. Parr. 1999. The 1998 ogi-fonix broadcast news transcription system. In *DARPA Broadcast News Workshop.*

Y. Yan, X. Wu, J. Shalkwyk, and R. Cole. 1998. Development of CSLU LVCSR: The 1997 DARPA HUB4 evaluation system. In *DARPA Broadcast News Transcription and Understanding Workshop.*