POS Tagging versus Classes in Language Modeling

Peter A. Heeman
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. The use of POS tags allows more sophisticated generalizations than are afforded by using a class-based approach. Furthermore, if we want to incorporate speech repair and intonational phrase modeling into the language model, using POS tags rather than classes gives better performance in this task.

1 Introduction

For recognizing spontaneous speech, the acoustic signal is to weak to narrow down the number of word candidates. Hence, speech recognizers employ a language model that prunes out acoustic alternatives by taking into account the previous words that were recognized. In doing this, the speech recognition problem is viewed as finding the most likely word sequence $\hat{W}$ given the acoustic signal (Jelinek, 1985).

$$\hat{W} = \arg\max_{W} \Pr(W|A) \quad (1)$$

We can rewrite the above using Bayes' rule.

$$\hat{W} = \arg\max_{W} \frac{\Pr(A|W)\Pr(W)}{\Pr(A)} \quad (2)$$

Since $\Pr(A)$ is independent of the choice of W, we simplify the above as follows.

$$\hat{W} = \arg\max_{W} \Pr(A|W)\Pr(W) \quad (3)$$

The first term, $\Pr(A|W)$, is the acoustic model and the second term, $\Pr(W)$, is the language model, which assigns a probability to the sequence of words W. We can rewrite W explicitly as a sequence of words $W_1W_2W_3 \ldots W_N$, where N is the number of words in the sequence. For expository ease, we use the notation $W_{i,j}$ to refer to the sequence of words W_i to W_j. We now use the definition of conditional probabilities to rewrite $\Pr(W_{1,N})$ as follows.

$$\Pr(W_{1,N}) = \prod_{i=1}^{N} \Pr(W_i|W_{1,i-1}) \quad (4)$$

To estimate the probability distribution, a training corpus is typically used from which the probabilities can be estimated using 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, which is defined as follows.

$$H = -\frac{1}{N}\sum_{i=1}^{N} \log_2 \hat{\Pr}(w_i|w_{1,i-1}) \quad (5)$$

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.

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

Using classes has the potential of reducing the problem of sparseness of data by allowing generaliza-

tions 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 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 sequence of words "hello can I help you". Here, "hello" is being used as an acknowledgment, "can" as a modal verb, "I" as a pronoun, "help" as an untensed verb, and "you" as a pronoun. 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.

$$
\begin{aligned}
&\Pr(W_{1,N}) \\
&\quad = \sum_{P_{1,N}} \Pr(W_{1,N}P_{1,N}) \\
&\quad = \sum_{P_{1,N}} \prod_{i=1}^{N} \Pr(W_i|W_{1,i-1}P_{1,i}) \Pr(P_i|W_{1,i-1}P_{1,i-1}) \\
&\quad \approx \sum_{P_{1,N}} \prod_{i=1}^{N} \Pr(W_i|P_i) \Pr(P_i|P_{i-2,i-1}) \qquad (6)
\end{aligned}
$$

The above approach for incorporating POS information into a language model has not been of much success in improving speech recognition performance. Srinivas (1996) reports that such a model results in a 24.5% increase in perplexity over a word-based model on the Wall Street Journal; Niesler and Woodland (1996) report 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).

In the rest of the paper, we first describe the annotations of the Trains corpus. We next present our POS-based language model and contrast its performance with a class-based model. We then augment these models to account for speech repairs and intonational phrase, and show that the POS-based one performs better than the class-based one for modeling speech repairs and intonational phrases.

2 The Trains Corpus

As part of the TRAINS project (Allen et al., 1995), a long term research project to build a conversationally proficient planning assistant, we collected a corpus of problem solving dialogs (Heeman and Allen, 1995). The dialogs involve two human participants, one who is playing the role of a user and has a certain task to accomplish, and another who is playing the role of a planning assistant. The collection methodology was designed to make the setting as close to human-computer interaction as possible, but was not a *wizard* scenario, where one person pretends to be a computer. Table 1 gives information about the corpus.

Dialogs	98
Speakers	34
Turns	6163
Words	58298
Fragments	756
Distinct Words	859
Distinct Words/POS	1101
Singleton Words	252
Singleton Words/POS	350
Intonational Phrases	10947
Speech Repairs	2396

Table 1: Size of the Trains Corpus

2.1 POS Annotations

Our POS tagset is based on the Penn Treebank tagset (Marcus et al., 1993), but modified to include tags for discourse markers and end-of-turns, and to provide richer syntactic information (Heeman, 1997). Table 2 lists our tagset with differences from the Penn tagset marked in bold. Contractions are annotated using '∧' to conjoin the tag for each part; for instance, "can't" is annotated as '**MD∧RB**'.

AC	Acknowledgement	**DP**	Pro-form	NNPS	Plural proper Noun	**TURN**	Turn marker
BE	Base form of "be"	DT	Determiner	PDT	Pre-determiner	**UH_D**	Discourse interjection
BED	Past tense	EX	Existential "there"	POS	Possessive	**UH_FP**	Filled pause
BEG	Present participle	**HAVE**	Base form of "have"	**PPREP**	Pre-preposition	VB	Verb base form (other than 'do', 'be', or 'have')
BEN	Past participle	**HAVED**	Past tense	**PREP**	Preposition	VBD	Past tense
BEP	Present	**HAVEP**	Present	PRP	Personal pronoun	VBG	Present participle
BEZ	3rd person sing. pres.	**HAVEZ**	3rd person sing. pres.	PRP$	Possessive pronoun	VBN	Past participle
CC	Co-ordinating conjunct	JJ	Adjective	RB	Adverb	VBP	Present tense
CC_D	Discourse connective	JJR	Relative Adjective	RBR	Relative Adverb	VBZ	3rd person sing. pres.
CD	Cardinal number	JJS	Superlative Adjective	RBS	Superlative Adverb	WDT	Wh-determiner
DO	Base form of "do"	MD	Modal	**RB_D**	Discourse adverbial	WP	Wh-pronoun
DOD	Past tense	NN	Noun	RP	Reduced particle	WRB	Wh-adverb
DOP	Present	NNS	Plural noun	SC	Subordinating conjunct	WP$	Processive Wh-pronoun
DOZ	3rd person sing. present	NNP	Proper Noun	TO	To-infinitive		

Table 2: Part-of-Speech Tags used in the Trains Corpus

2.2 Speech Repair Annotations

Speech repairs occur where the speaker goes back and changes or repeats what was just said (Heeman, 1997), as illustrated by the following.

Example 1 (d92a-2.1 utt29)

the one with the bananas I mean that's taking the bananas

reparandum *ip* *et* *alteration*

Speech repairs have three parts (some of which are optional): the *reparandum*, which are the words the speaker wants to replace, an *editing term*, which helps mark the repair, and the *alteration*, which is the replacement of the reparandum. The end of the reparandum is referred to as the *interruption point.*

For annotating speech repairs, we have extended the scheme proposed by Bear *et al.* (1992) so that it better deals with overlapping and ambiguous repairs. Like their scheme, ours allows the annotator to capture the word correspondences that exist between the reparandum and the alteration. Below, we illustrate how a speech repair is annotated. In this example, the reparandum is "engine two from Elmi(ra)-", the editing term is "or", and the alteration is "engine three from Elmira". The word matches on "engine" and "from" are annotated with '**m**' and the word replacement of "two" by "three" is annotated with '**r**'.

Example 2 (d93-15.2 utt42)

engine two from Elmi(ra)- or engine three from Elmira

m1 r2 m3 m4 et m1 r2 m3 m4

ip:mod+

2.3 Intonation Annotations

Speakers break up their speech into intonational phrases. This segmentation serves a similar purpose as punctuation does in written speech. The ToBI annotation scheme (Silverman et al., 1992) involves labeling the accented words, intermediate phrases and intonational phrases with high and low accents. Since we are currently only interested in the intonational phrase segmentation, we only label the intonational phrase endings.

3 POS-Based Language Model

In this section, we present an alternative formulation for using POS tags in a statistical language model. Here, POS tags are viewed as part of the output of the speech recognizer, rather than intermediate objects (Heeman and Allen, 1997a; Heeman, 1997).

3.1 Redefining the Recognition Problem

To add POS tags into the language model, we refrain from simply summing over all POS sequences as illustrated in Section 1.2. 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.

$$\begin{aligned}\hat{W}\hat{P} &= \arg\max_{W,P} \Pr(WP|A) \\ &= \arg\max_{WP} \frac{\Pr(A|WP)\Pr(WP)}{\Pr(A)} \\ &= \arg\max_{WP} \Pr(A|WP)\Pr(WP) \qquad (7)\end{aligned}$$

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 this accounts for both the sequence of words and the POS assignment for those words. We rewrite the sequence WP explicitly in terms of the N words and their corresponding POS tags, thus giving us 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 4, we rewrite the probability $\Pr(W_{1,N}P_{1,N})$ as follows using the definition of conditional probability.

$$\begin{aligned}
&\Pr(W_{1,N}P_{1,N}) \\
&= \prod_{i=1}^{N} \Pr(W_iP_i|W_{1,i-1}P_{1,i-1}) \\
&= \prod_{i=1}^{N} \Pr(W_i|W_{1,i-1}P_{1,i}) \Pr(P_i|W_{1,i-1}P_{1,i-1}) \qquad (8)
\end{aligned}$$

Equation 8 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 9 and 10. However, to successfully incorporate POS information, we need to account for the full richness of the probability distributions. Hence, we cannot use these two assumptions when learning the probability distributions.

$$\Pr(W_i|W_{1,i-1}P_{1,i}) \not\approx \Pr(W_i|P_i) \qquad (9)$$

$$\Pr(P_i|W_{1,i-1}P_{1,i-1}) \not\approx \Pr(P_i|P_{1,i-1}) \qquad (10)$$

3.2 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. The algorithm starts with a single node. It then finds a question to ask about the node in order to partition the node into two *leaves*, each being more informative as to which event 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. The approach of using decision trees will become even more critical in the next two sections where the probability distributions will be conditioned on even richer context.

3.2.1 Simple Questions

One important aspects of using a decision tree algorithm is the form of the questions that it is allowed to ask. We allow two basic types of information to be used as part of the context: numeric and categorical. For a numeric variable N, the decision tree searches for questions of the form 'is $N >= n$', where n is a numeric constant. For a categorical variable C, it searches over questions of the form 'is $C \in S$' where S is a subset of the possible values of C. We also allow restricted boolean combinations of elementary questions (Bahl et al., 1989).

3.2.2 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. 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 are statistically similar. 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.

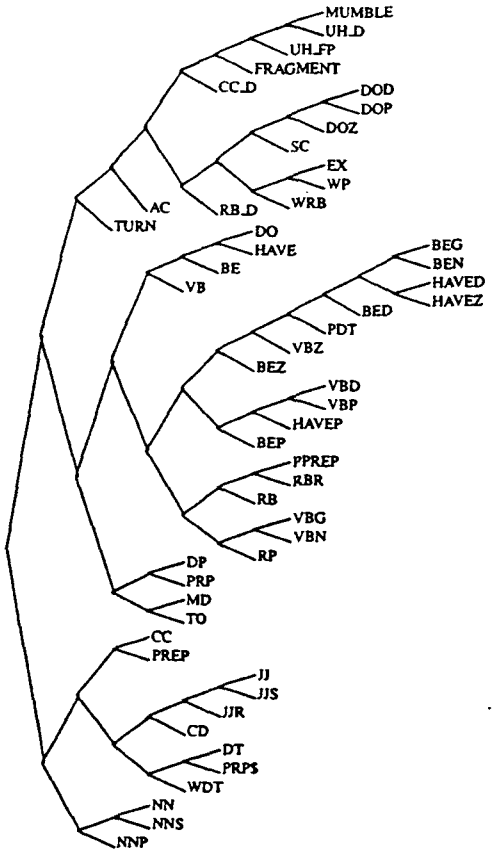

Figure 1: Classification Tree for POS Tags

3.2.3 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. Building a word classification tree for each POS tag means that the tree will not be polluted by words that are ambiguous as to their POS tag, as exemplified by the word "loads", which is used in the Trains corpus as both a third-person present tense verb **VBZ** and as a plural noun **NNS**. Furthermore, building a tree for each POS tag simplifies the task because the hand annotations of the POS tags resolve a lot of the difficulty that the algorithm would otherwise have to handle. This allows effective trees to be built even when only a small amount of data is available.

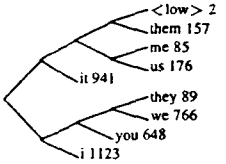

Figure 2: Classification Tree for Personal Pronouns

Figure 2 shows the classification tree for the personal pronouns (**PRP**). 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. The class **low** is used to group singleton words.

3.3 Results

Before giving a comparison between our POS-based model and a class-based model, we first describe the experimental setup and define the perplexity measures that we use to measure the performance.

3.3.1 Experimental Setup

To make the best use of our limited data, we used a six-fold cross-validation procedure: each sixth of the data was tested using a model built from the remaining data. Changes in speaker are marked in the word transcription with the special token **<turn>**. We treat contractions, such as "that'll" and "gonna", as separate words, treating them as "that" and "'ll" for the first example, and "going" and "ta" for the second.[1] We also changed all word fragments into the token **<fragment>**.

Since current speech recognition rates for spontaneous speech are quite low, we have run the experiments on the hand-collected transcripts. 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.

3.3.2 Branching Perplexity

Our POS-based model is not only predicting the next word, but its POS tag as well. To estimate

[1] See Heeman and Damnati (1997) for how to treat contractions as separate words in a speech recognizer. .

the branching factor, and thus the size of the search space, we use the following formula for the entropy, where d_i is the POS tag for word w_i.

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

3.3.3 Word Perplexity

In order to compare a POS-based model against a traditional language model, we should not penalize the POS-based model for incorrect POS tags, and hence we should ignore them when defining the perplexity. Just as with a traditional model, we base the perplexity measure on $\Pr(w_i|w_{1,i-1})$. However, for our model, this probability is not estimated. Hence, we must rewrite it in terms of the probabilities that we do estimate. To do this, our only recourse is to sum over all possible POS sequences.

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

3.3.4 Using Richer Context

Table 3 shows the effect of varying the richness of the information that the decision tree algorithm is allowed to use in estimating the POS and word probabilities. The second column uses the approximations given in Equation 9 and 10. The third column gives the results using the full context. The results show that adding the extra context has the biggest effect on the perplexity measures, decreasing the word perplexity from 43.22 to 24.04, a reduction of 44.4%. The effect on POS tagging is less pronounced, but still gives an error rate reduction of 3.8%. Hence, to use POS tags during speech recognition, one must use a richer context for estimating the probabilities than what is typically used.

Context for W_i Context for D_i	D_i $D_{i-2,i-1}$	$D_{i-2,i}W_{i-2,i-1}$ $D_{i-2,i-1}W_{i-2,i-1}$
POS Errors	1778	1711
POS Error Rate	3.04	2.93
Word Perplexity	43.22	24.04
Branching Perplexity	47.25	26.35

Table 3: Using Richer Context

3.3.5 Class-Based Decision-Tree Models

In this section, we compare the POS-based model against a class-based model. To make the comparison as focused as possible, we use the same methodology for estimating the probability distributions as we used for the POS-based model. The classes were obtained from the 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 choose the number of classes that gives the best perplexity results, which was 100 classes. We then built word classification trees, just as we did for the POS-based approach, where words from different classes are not allowed to be merged. The resulting class-based model achieved a perplexity of 25.24 in comparison to 24.04 for the POS-based model. This improvement is due to two factors. First, tracking the syntactic role of each word gives valuable information for predicting the subsequent words. Second, the classification trees for the POS-based approach, which the decision tree algorithm uses to determine the equivalence classes, are of higher quality. This is due to the POS-based classification trees using the hand-annotated POS information, since they take advantage of the hand-coded knowledge present in the POS tags and are not polluted by words that take on more than one syntactic role.

3.3.6 Preliminary Wall Street Journal Results

For building a system that partakes in dialogue, read-speech corpora, such as the Wall Street Journal, are not appropriate. However, to make our results more comparable to the literature, we have done preliminary tests on the Wall Street Journal corpus in the Penn Treebank, which has POS annotations. This corpus has a significantly larger vocabulary size (55800 words) than the Trains corpus. Our current algorithm for clustering the words takes space in proportion to the square of the number of unique word/POS combinations (minus any that get grouped into the low occurring class). More work is needed to handle larger vocabulary sizes. Using 78,800 words of data, with a vocabulary size of 9711, we achieved a perplexity of 250.75 on the known words in comparison to a trigram word-based backoff model (Katz, 1987) built with the CMU toolkit (Rosenfeld, 1995), which achieved a perplexity of 296.43. More work is needed to see if these results scale up to larger vocabulary and training data sizes.

4 Adding Repairs and Phrasing

Just as we redefined the speech recognition problem so as to account for POS tagging, we do the same for modeling intonational phrases and speech

repairs. We introduce null tokens between each pair of words w_{i-1} and w_i (Heeman and Allen, 1997b), which will be tagged as to the occurrence of these events. The variable T_i indicates if word w_{i-1} ends an intonational phrase (T_i=%), or not (T_i=**null**).

For detecting speech repairs, we have the problem that repairs are often accompanied by an editing term, such as "um", "uh", "okay", or "well", and these must be identified as such. Furthermore, an editing term might be composed of a number of words, such as "let's see" or "uh well". Hence we use two tags: an editing term tag E_i and a repair tag R_i. The editing term tag indicates if w_i starts an editing term (E_i=**Push**), if w_i continues an editing term (E_i=**ET**), if w_{i-1} ends an editing term (E_i=**Pop**), or otherwise (E_i=**null**). The repair tag R_i indicates whether word w_i is the onset of the alteration of a fresh start (R_i=C), a modification repair (R_i=M), or an abridged repair (R_i=A), or there is no repair (R_i=**null**). Note that for repairs with an editing term, the repair is tagged after the extent of the editing term has been determined. Below we give an example showing all non-null tone, editing term and repair tags.

Example 3 (d93-18.1 utt47)
it takes one **Push** you **ET** know **Pop** M two hours %

If a modification repair or fresh start occurs, we need to determine the extent (or the onset) of the reparandum, which we refer to as *correcting* the speech repair. Often, speech repairs have strong word correspondences between the reparandum and alteration, involving word matches and word replacements. Hence, knowing the extent of the reparandum means that we can use the reparandum to predict the words (and their POS tags) that make up the alteration. In our full model, we add three variables to account for the correction of speech repairs (Heeman and Allen, 1997b; Heeman, 1997). We also add an extra variable to account for silences between words. After a silence has occurred, we can use the silence to better predict whether an intonational boundary or speech repair has just occurred.

Below we give the redefinition of the speech recognition problem (without speech repair correction and silence information). The speech recognition problem is redefined so that its goal is to find the maximal assignment for the words as well as the POS, intonational, and repair tags.

$$\hat{W}\hat{P}\hat{R}\hat{E}\hat{T} = \arg\max_{WPRET} \Pr(WPRET|A)$$

Just as we did in Equation 8, we rewrite the above in terms of five probability distributions, each of which need to be estimated. The context for each of the probability distributions includes all of the previous context. In principal, we could give all of this context to the decision tree algorithm and let it decide what information is relevant in constructing equivalence classes of the contexts. However, the amount of training data is limited (as are the learning techniques) and so we need to encode the context in order to simplify the task of constructing meaningful equivalence classes. Hence we restructure the context to take into account the speech repairs and boundary tones (Heeman, 1997).

4.1 Results

We now contrast the performance of augmenting the POS-based model with speech repair and intonational modeling versus augmenting the class-based model. Just as in Section 3, all results were obtained using a six-fold cross-validation procedure from the the hand-collected transcripts. We ran these transcripts through a word-aligner (Ent, 1994), a speech recognizer constrained to recognize what was transcribed, in order to automatically obtain silence durations. In predicting the end of turn marker <**turn**>, we do not use any silence information.

4.1.1 Recall and Precision

We report results on identifying speech repairs and intonational phrases in terms of *recall*, *precision* and *error rate*. The recall rate is the number of times that the algorithm correctly identifies an event over the total number of times that it actually occurred. The precision rate is the number of times the algorithm correctly identifies it over the total number of times it identifies it. The error rate is the number of errors in identifying an event over the number of times that the event occurred.

4.1.2 POS Tagging and Perplexity

The first set of experiments, whose results are given in Table 4, explore how POS tagging and word perplexity benefit from modeling boundary tones and speech repairs. The second column gives the results of the POS-based language model, introduced in Section 3. The third column adds in speech repair detection and correction, boundary tone identification, and makes use of silence information in detecting speech repairs and boundary tones. We see that this results in a perplexity reduction of 7.0%, and a POS error reduction of 8.1%. As we further improve the modeling of the user's utterance, we

	POS	Full Model
POS Errors	1711	1572
POS Error Rate	2.93	2.69
Word Perplexity	24.04	22.35
Branching Perplexity	26.35	30.26

Table 4: POS Tagging and Perplexity

expect to see further improvements in the language model. Of course, there is a penalty to pay in terms of increased search space size, as the increase in the branching perplexity shows.

4.1.3 Intonational Phrases

In Table 5, we demonstrate that modeling intonational phrases benefits from modeling POS tags. Column two gives the results of augmenting the class-based model of Section 3.3.5 with intonational phrase modeling and column three gives the results of augmenting the POS-based model. Contrasting the results in column two with those in column three, we see that using the POS-based model results in a reduction in the error rate of 17.2% over the class-based model. Hence, we see that modeling the POS tags allows much better modeling of intonational phrases than can be achieved with a class-based model. The fourth column reports the results using the full model, which accounts for interactions with speech repairs and the benefit of using silence information (Heeman and Allen, 1997b).

	Class-Based Tones	POS-Based Tones	Full Model
Errors	4859	4024	3632
Error Rate	44.38	36.75	33.17
Recall	74.55	81.72	84.76
Precision	79.74	81.55	82.53

Table 5: Detecting Intonational Phrase Boundaries

4.1.4 Detecting Speech Repairs

In Table 6, we demonstrate that modeling the detection of speech repairs (and editing terms) benefits from modeling POS tags. In the results below, we ignore errors that are the result of improperly identifying the type of repair, and hence score a repair as correctly detected as long as it was identified as either an abridged repair, modification repair or fresh start. Column two gives the results of augmenting the class-based model of Section 3.3.5 with speech repair modeling and column three gives the results of augmenting the POS-based model. In

	Class-Based Repairs	POS-Based Repairs	Full Model
Errors	1246	1106	839
Error Rate	52.00	46.16	35.01
Recall	64.98	68.61	76.79
Precision	79.27	82.28	86.66

Table 6: Detecting Speech Repairs

terms of overall detection, the POS-based model reduces the error rate from 52.0% to 46.2%, a reduction of 11.2%. This shows that speech repair detection profits from being able to make use of syntactic generalizations, which are not available from a class-based approach. The final column gives the results of the full model, which accounts for interactions with speech repair correction and intonational phrasing, and uses silence information.

5 Conclusion

In this paper, we presented a POS-based language model. Unlike previous approaches that use POS tags in language modeling, we redefine 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 make use of the POS tags, we use a decision tree algorithm to learn the probability distributions, and a clustering algorithm to build hierarchical partitionings of the POS tags and the word identities. Furthermore, we take advantage of the POS tags in building the word classification trees and in estimating the word probabilities, which both results in better performance and significantly speeds up the training procedure. We find that using the rich context afforded by decision tree results in a perplexity reduction of 44.4%. We also find that the POS-based model gives a 4.2% reduction in perplexity over a class-based model, also built with the decision tree and clustering algorithms. Preliminary results on the Wall Street Journal corpus are also encouraging. Hence, using a POS-based model results in an improved language model as well as accomplishes the first part of the task in linguistic understanding.

We also see that using POS tags in the language model aids in the identification of boundary tones and speech repairs, which we have also incorporated into the model by further redefining the speech recognition problem. The POS tags allow these two

processes to generalize about the syntactic role that words are playing in the utterance rather than using crude class-based approaches which does not distinguish this information. We also see that modeling these phenomena improves the POS tagging results as well as the word perplexity.

6 Acknowledgments

We wish to thank Geraldine Damnati. The research involved in this paper was done while the first author was visiting at CNET, France Télécom.

References

J. Allen, L. Schubert, G. Ferguson, P. Heeman, C. Hwang, T. Kato, M. Light, N. Martin, B. Miller, M. Poesio, and D. Traum. 1995. The Trains project: A case study in building a conversational planning agent. *Journal of Experimental and Theoretical AI*, 7:7–48.

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

J. Bear, J. Dowding, and E. Shriberg. 1992. Integrating multiple knowledge sources for detection and correction of repairs in human-computer dialog. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 56–63.

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*, pages 134–139.

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

P. Brown, V. Della Pietra, P. deSouza, J. Lai, and Robert L. 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*, pages 199–202.

Entropic Research Laboratory, Inc., 1994. *Aligner Reference Manual.* Version 1.3.

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

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

P. Heeman and J. Allen. 1997b. Intonational boundaries, speech repairs, and discourse markers: Modeling spoken dialog. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 254–261.

P. Heeman and G. Damnati. 1997. Deriving phrase-based language models. In *IEEE Workshop on Speech Recognition and Understanding*, pages 41–48.

P. Heeman. 1997. Speech repairs, intonational boundaries and discourse markers: Modeling speakers' utterances in spoken dialog. TR 673, Dept. of Computer Science, U. of Rochester. Doctoral dissertation.

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

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

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 European Conference on Speech Communication and Technology*, pages 973–976.

D. Magerman. 1994. Natural language parsing as statistical pattern recognition. Doctoral dissertation, Dept. of Computer Science, Stanford.

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

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*, pages 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.*

K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, and J. Hirschberg. 1992. ToBI: A standard for labelling English prosody. In *Proceedings of the 2nd International Conference on Spoken Language Processing*, pages 867–870.

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