Probabilistic Models of Short and Long Distance Word Dependencies in Running Text

Julian Kupiec
XEROX PALO ALTO RESEARCH CENTER
3333 Coyote Hill Road
Palo Alto, CA 94304

Abstract

This article describes two complementary models that represent dependencies between words in local and non-local contexts. The type of local dependencies considered are sequences of part of speech categories for words. The non-local context of word dependency considered here is that of word recurrence, which is typical in a text. Both are models of phenomena that are to a reasonable extent domain independent, and thus are useful for doing prediction in systems using large vocabularies.

Modeling Part of Speech Sequences

A common method for modeling local word dependencies is by means of second order Markov models (also known as trigram models). In such a model the context for predicting word w_i at position i in a text consists of the two words w_{i-1}, w_{i-2} that precede it. The model is built from conditional probabilities: $P(w_i \mid w_{i-1}, w_{i-2})$. The parameters of a part of speech (POS) model are of the form: $P(w_i \mid C_i) \times P(C_i \mid C_{i-1}, C_{i-2})$. That is, for word w_i a POS category C_i is first predicted, based on the POS categories of the two previous words. The word w_i is then predicted in terms of C_i. If the vocabulary consists of the set of N words $\{v_1, v_2 ... v_N\}$, then w_i, w_{i-1} and w_{i-2} range over the elements of this set. For a model containing M parts of speech, $\{S_1, S_2 ... S_M\}$, the variables C_i, C_{i-1} and C_{i-2} likewise range over the M elements.

POS language models have been used in speech recognition systems (Dumouchel, Gupta, Lennig and Mermelstein, 1988; Shikano, 1987) and for phoneme-to-text transcription (Derouault and Merialdo, 1986). In these systems the parameters are obtained from the analysis of an annotated training corpus. To create the training corpus a set of POS categories is first defined. A word in the vocabulary may be associated with several POS categories depending on the roles it can play in a sentence. A suitably large corpus of training text is then manually analyzed and each word of the corpus is annotated with an unambiguous POS category according to its function in the text. The Brown Corpus has been analyzed this way, using a set of 87 main categories (Francis and Kucera, 1982). To obtain the parameters of a language model, frequency counts are made and normalized to produce the required sets of parameters. The problem of training a model, and the reliability of the resulting parameters rests on a laborious manual annotation of a necessarily large amount of training text. To reduce this burden, a bootstrap method can be used (Derouault and Merialdo, 1986). First a relatively small amount of text is annotated, to create a partial model. The partial model is then used to automatically annotate more text, which is then corrected manually, and used to re-train the model.

Here, an alternative approach has been taken to the training problem, which is based on work by Jelinek (Jelinek, 1985a). In this approach, a POS language model is viewed as a Hidden Markov model (HMM). That is, a word sequence reflects an underlying sequence of parts of speech, which is hidden from the observer. The advantage of considering the model in such terms is that the need for an annotated training corpus is eliminated, resulting in greater flexibility. The model can be trained with alternative sets of POS categories, and can accommodate special POS categories that are defined for specialized domains. Another advantage is that the method can be applied to other languages. To train a model requires the following:

1. A suitably large training corpus of text (that is not annotated).
2. A vocabulary of words occurring in the corpus, where each word is associated with a list of all the possible parts of speech it can assume.
3. Estimates of the frequency of occurrence $P(v_1) ... P(v_N)$ of the words in the vocabulary. These are used to set initial probability values in the model.

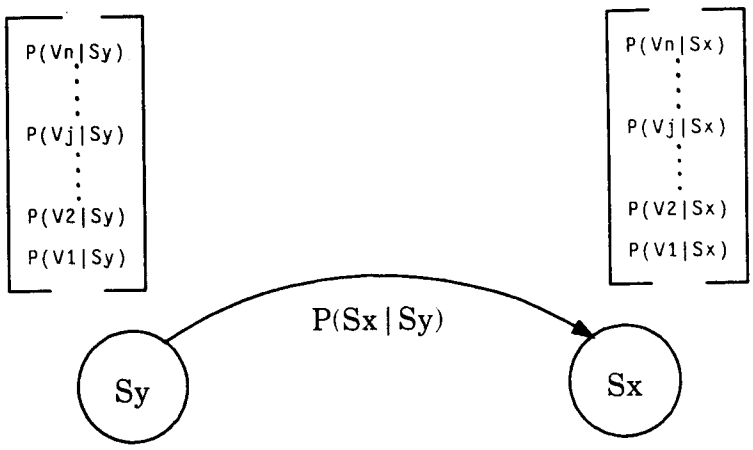

Figure 1: A typical state transition in a first order HMM

States in the HMM correspond to POS categories, and are labeled by the category they represent. The observations generated at a state are those words that can function as the POS category associated with the state. The probability $P(C_i = S_x \mid C_{i-1} = S_y)$ labels the transition from state S_y to state S_x, and $P(w_i = v_j \mid C_i = S_x)$ represents the jth element of the output matrix for state S_x. These are shown in Figure 1. If word v_k cannot occur as POS category S_x then the kth element of the output matrix for state S_x is zero. This provides a strong source of constraint on the estimation process. Each state is connected to all other states, which permits the modeling of any sequence of parts of speech.

The HMM used for the work is based on the implementation of Rabiner, Levinson and Sondhi (Rabiner et al., 1983). A first order HMM was used as a starting point because the resources required to train it are significantly less than for a second order model, and thus facilitate experimentation. Twenty-one parts of speech were used, corresponding to traditional POS categories such as *determiner, adverb, preposition, noun, noun-plural, verb-progressive, verb-past-participle* etc.

In the model described here, the elements of the output matrix have been assigned to *word equivalence classes* rather than individual words. This is due to the observation that in unrestricted domains, the number N of different word types that can occur is so large it precludes practical estimation of the required number of parameters $P(w_i \mid C_i)$. Words are partitioned into L equivalence classes $W_1 ... W_L$. All words that can function as a noun only are in one equivalence class, and all words that can function as either a noun or adjective are in another class, and so on. A total of $L = 129$ equivalence classes were used in the model. Each equivalence class has an index in the output matrix. Words have an uneven distribution within these classes, e.g. in the dictionary that was used the word "more" is the only member of the class *noun-or-comparative-adjective-or-adverb*. Prior to training, initial values of probabilities in all the matrices must be chosen. The transition matrix is set so that all the transitions from a state are equiprobable. Word occurrence probabilities $P(v_j)$ are used to obtain the initial output matrix probabilities. They are first converted to probabilities of word equivalence classes $P(W_k)$. The probability of each equivalence class W_k is then divided equally among the POS categories that are in the equivalence class, to give weights $F(W_k, C_i)$. This reflects the assumption that all words in an equivalence class can initially function equiprobably as any POS category of the class. The output matrix elements for each state are constructed using the various $F(W_k, C_i)$. For each state, the elements are then normalized to sum to unity.

The process of training a HMM is to statistically infer parameters that maximize the likelihood of the training text, and the Baum-Welch algorithm (Baum, 1972) was used for this purpose. Typically, the model is trained on several iterations over the training data, until the parameter estimates converge. Then, for any given test sentence, the Viterbi algorithm (Viterbi, 1967) can be used to find the most likely state sequence through the model, which maximizes the probability of seeing the test sentence. The state sequence is used as the annotation for the sentence.

A text corpus containing 1,180,000 words was used for training and testing the model. The corpus is a

I	THINK	THAT	IT	IS	A	FAR
subj-pro	**verb**	**conj**	**pro**	**verb**	**det**	**adj**
noun		*adj*	*noun*		*noun*	*adverb*
		pro				
		adverb				
		det				
MORE	COMPLICATED	ISSUE	THAN	YOU	SEEM	TO
comp-adj	**adj**	**noun**	**conj**	**pro**	**verb**	**prep**
adverb	*past part*	*verb*				*adverb*
noun						
THINK	TO	PUT	IN	ELEMENTARY	FUNCTIONS	
verb	**prep**	**verb**	**prep**	**adj**	**noun-plural**	
	adverb		*adj*		*verb 3rd person*	
			noun			
			adverb			

Figure 2: An Example of Annotation

collection of electronic mail messages concerning the design of the Common Lisp programming language. It contains 23,190 different words, of which 8,300 only occur once. A dictionary of approximately 30,000 words was also available, each word tagged with its possible parts of speech. An annotated corpus vocabulary was constructed by intersecting the dictionary with the corpus. This resulted in 90% coverage of the total number of words in the corpus, and 45% coverage of the total word types that appear in the corpus. Words that did not have vocabulary entries giving their parts of speech were marked with a special POS category called *unlabeled.* A total of 21 POS categories were used initially, resulting in 441 transitions. The output matrix contained 129 elements per state (corresponding to the number of word equivalence classes in the annotated corpus vocabulary). The model was trained initially on 3000 sentences from the corpus, using 10 iterations. Preliminary results are good, even for a first order model. An example of annotation of a sentence is shown in Figure 2. The annotation provided by the Viterbi algorithm is shown in bold on the line below the words of the sentence. The other possible parts of speech present in the dictionary are shown below the annotation. It can be seen that the major dependencies have been inferred correctly; the main difference from the preferred annotation lies in the substitution of adjectival for adverbial categories.

Modeling Word Recurrence

The previous method is useful for modeling sequences of parts of speech. Words were grouped into equivalence classes, and it was noted that there was an uneven distribution of words in different classes. The classes for topic independent words such as *the, of, in,* etc. are relatively small (some are unique) and as a result these words are easily predicted given their equivalence class. The equivalence classes for topic dependent words such as nouns, adjectives and verbs are very much larger. This section describes a method which addresses one aspect of the problem of predicting topic dependent words. The method is concerned with modeling word recurrence, which is typical in discourse. The use of previous discourse history as a means of prediction has been recognized for some time. Barnett (Barnett, 1973) describes a system which allocates higher scores to "content" words if they have already been mentioned. In the MINDS spoken language dialogue system (Young, Hauptmann, Ward, Smith and Werner; 1989) extensive use is made of previous dialogue history, which is integrated with high level knowledge sources. These include dialogue knowledge, user goals, plans and focus.

Our model makes use of previous history at the lower level of word transition probabilities, where automatic statistical estimation can be performed. Word recurrence takes place at greater distances than the two word context used in the previous model. To account for this, a long range memory called a *word cache*

is used. A word cache may be described as *static* or *dynamic* depending on whether its contents are fixed. A frequency ordered list of words occurring in a corpus represents a static word cache. In this type of word cache the overall probability of occurrence of a word related to a specific topic tends to be small, as a large corpus of text may be composed of material on a large number of diverse topics. The contents of a dynamic word cache may be some function of the previous text history. In a small section of the corpus concerned with a topic, words related to the topic tend to be repeated. A word's probability may then be higher than its overall probability. A dynamic word cache can track this recurrence effect. Dynamic caches can be further split into two kinds. A *token* cache contains the previous n words (i.e. word tokens) seen in a text, and acts as a window containing the most recent words encountered. A *type* cache contains the previous n different words (i.e. word types) found, and can be implemented as a linked list of words. When a word is encountered that is already in the cache, the word is simply moved to the front of the list. When a new word type is seen it is placed at the front of the list. If the cache is full, the least recently used word is removed from the tail of the list. An interesting comparison between a static and dynamic word cache is given by the amount of coverage they provide. A static word cache containing the n overall most frequent words in the corpus of Common Lisp mail messages was compared with an equally sized dynamic type cache. Table 1 shows the coverage given over the whole corpus by the two kinds of cache, for various n. For cache sizes from 90 to 4000 words the dynamic type cache gave slightly better coverage than the optimum frequency ordered cache. This characteristic was also observed when using subsections of the corpus. The dynamic cache does not give 100% coverage when its size is equal to the vocabulary, because at the outset it is empty. Words entering the cache for the first time are not covered. The dynamic type cache has the advantage that it is adaptable over different domains whereas the optimum contents of a static cache must be determined by prior analysis. A dynamic token cache was also compared, and it gave inferior performance to either of the other kinds of cache because space in the cache is occupied by multiple tokens of the same word type.

Dynamic type caches have been considered as a means of vocabulary personalization (Jelinek, 1985b). In conjunction with interactive word entry, they have also been viewed as a way of obtaining high coverage in a speech recognition system using a very large vocabulary (Merialdo, 1988). The dynamic cache can also be considered as a means for improving word prediction. For this purpose, it is necessary to quantify its effect and allow direct comparison with an alternative. A first order (bigram) model of word dependency was compared with a similar model into which a dynamic cache had been incorporated. A first order model is composed of parameters of the form $P(w_i \mid w_{i-1})$. Both w_i and w_{i-1} range over all words in the vocabulary. If (v_x, v_y) is such a word pair, the conditional probability $P(w_i = v_x \mid w_{i-1} = v_y)$ denotes the corresponding model parameter. A dynamic cache D is a function of previous history, which can be evaluated at $i-1$ for any choice v_x of w_i, and its inclusion is modeled as $P(w_i \mid w_{i-1}, D_{i-1})$. D is binary valued and indicates for any word type v_x whether or not it is currently in the cache. This results in two sets of probabilities: $P(v_x \mid v_y, D = true)$, $P(v_x \mid v_y, D = false)$. These probabilities are obtained from frequency counts as shown in Figure 3. In the figure, $N(a, b)$ represents the count of the number of times the joint event a and b occurred in the training text.

The dynamic cache is likewise used to select probabilities when doing prediction. The corpus of Common

	Percent. Coverage of Corpus	
Cache Size	Static Cache	Dynamic Cache
19	30	21
90	50	50
512	75	78
1000	83	86
2000	90	91
4000	95	95
8000	98	97
12000	98.4	97.7
16000	99.4	97.9
23191	100	98

Table 1: Coverage provided by various sizes of Static and Dynamic Caches

$$P(x \mid y, D = true) = \frac{\text{N(Vx follows Vy, and Vx is in cache)}}{\text{N(Vy is seen, and Vx is in cache)}}$$

$$P(x \mid y, D = false) = \frac{\text{N(Vx follows Vy, and Vx is not in cache)}}{\text{N(Vy is seen, and Vx is not in cache)}}$$

Figure 3: Conditional Probabilities for a Dynamic Cache

Lisp mail messages was divided in the ratio 80% - 20% for training and testing respectively. Experiments were done with cache sizes ranging from 128 to 4096 words. The criterion of average rank of the correct word was used to compare the "dynamic" model based on $P(w_i \mid w_{i-1}, D)$ to the "static" model based on ordinary bigrams $P(w_i \mid w_{i-1})$. When a word pair v_x, v_y occurred in the test text, which had zero probability in the training text, unigrams were used to rank v_x. The dynamic model used $P(v_x \mid D)$ whereas the static model used $P(v_x)$. Several runs were made using different sections of the corpus for training and testing. For any run, less than 5% of words in the test text were absent from the training text. Over different runs, the average rank of the correct word in the static model ranged from 520 to 670. In each run, the dynamic model consistently produced a lower average rank, ranging from 7% to 17% less than that of the static model. The performance of the model varied by less than 1% for cache sizes between 350 and 750.

Another method of combining the bigram and unigram probabilities is by means of an interpolated estimator (Jelinek and Mercer, 1980). An interpolated estimator is a weighted linear combination of other estimates. Optimum weights are derived by the Baum-Welch algorithm, and their values give an indication of the overall utility of each component. Parameters of both the static and dynamic models were used in an interpolated estimator. The corpus was divided in the proportion 60% - 20% - 20% respectively for training the model, obtaining the weights, and test text. The use of the interpolated estimator contributed a further reduction in average rank of a few percent. A typical set of values that were obtained for weights of each component are shown below:

$$P_{int}(v_x \mid v_y) = 0.3P(v_x \mid v_y, D) + 0.25P(v_x \mid v_y) + 0.35P(v_x \mid D) + 0.1P(v_x)$$

Discussion

POS categories provide reasonable models of local word order in English. Further work is necessary to understand the sensitivity of the model to factors such as the granularity of POS classifications. States representing sentence boundary and punctuation marks would also be useful. Our model would benefit from further refinement of some POS categories. The category *auxiliary-verb* is currently absent, and the words "might" and "will" are in the equivalence class labeled *noun-or-verb*. Accordingly, they are classed as exhibiting the same behaviour as words like "thought", "feel" and "structure". It may also be advantageous to assign unique equivalence classes to common words, retaining the same POS categories, but allowing them to assume different parameter values in the output matrix. This enables the modeling of words that function uniquely. Moreover, common words tend to be topic independent and their estimation is not impaired by lack of data. Higher order conditioning is also an obvious area for further work. Due to the limitations of local context and the simplicity of the model, it cannot resolve all kinds of ambiguities found in text. Consider, for instance, the disambiguation of the word "that" in the following sentences:

"I think that function is more important than form".
"I think that function is more important than this one".

Traditional parts of speech have been used deliberately, to enable the model to interface to other tools of computational linguistics. In particular, a morphological analyzer is being integrated into the model to aid POS category predictions for words that are not in the vocabulary. The output from the model can also be used as input to other types of linguistic processing.

Results indicate that word prediction can be improved by making use of word recurrence, which can be considered as a simple focus-of-attention strategy. For speech recognition it would be a suitable component in a language model that must cover two or more application areas, where each area has its own set of commonly used topic dependent words. The improvements provided by the model depend on the extent to which the usage of a word is repeatedly clustered, resulting in a non-uniform distribution throughout a text. Unlike the part of speech model, common "function words" do not contribute to the word recurrence model. The two models can be viewed as addressing complementary topic-independent and topic-dependent aspects, and could be integrated into a combined model. The word recurrence results would benefit from verification on other corpora, as a corpus consisting of electronic mail is not the ideal choice for such experiments, and must be used with caution. The phenomenon of word recurrence is subsumed by the more general interaction of topic related words, which may be used to predict each other as well as exhibiting recurrence. This would be an interesting direction for future effort.

Acknowledgement

I would like to thank Jan Pedersen of Xerox PARC, for fruitful discussion and his comments. This work was sponsored in part by the Defense Advanced Research Projects Agency (DOD), under the Information Science and Technology Office, contract #N00140-86-C-8996.

References

P. Dumouchel, V. Gupta, M. Lennig & P. Mermelstein. Three Probabilistic Language Models for a Large-Vocabulary Speech Recognizer. Proc. 1988 Int. Conf. on Acoustics, Speech and Signal Processing.

A.M. Derouault, B. Merialdo. Natural Language Modeling for Phoneme-to-Text Transcription. IEEE Trans. on Pattern Analysis and Machine Intelligence Vol. PAMI-8, No. 6, November 1986.

K. Shikano. Improvement of Word Recognition Results by Trigram Model. Proc. 1987 Int. Conf. on Acoustics, Speech and Signal Processing.

W. N. Francis, H. Kucera. Frequency Analysis of English Usage. Houghton Mifflin, 1982.

L.R. Rabiner, S.E. Levinson, and M.M. Sondhi. An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition. Bell System Technical Journal, Vol. 62, No. 4, April 1983. pp 1035-1074.

(1985a) F. Jelinek. Self-Organized Language Modeling for Speech Recognition. Unpublished Technical Report, 1985. IBM T.J. Watson Research Center, Yorktown Heights, N.Y.

L.E. Baum. An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of a Markov Process. Inequalities, 3, 1972. pp. 1-8.

A. J. Viterbi. Error Bounds for Convolutional Codes and an Asymptotically Optimal Decoding Algorithm. IEEE Trans. on Information Theory Vol. IT-13, April 1967. pp. 260-269.

J. Barnett. A Vocal Data Management System. IEEE Trans. on Audio and Electroacoustics Vol. AU-21 No. 3, June 1973.

S. Young, A. Hauptmann, W. Ward, E. Smith, P. Werner. High Level Knowledge Sources in Usable Speech Recognition Systems. CACM Vol. 32 No. 2, February 1989.

(1985b) F. Jelinek. The Development of an Experimental Discrete Dictation Recognizer. Proc. IEEE, Vol. 73, No. 11, November 1985.

B. Merialdo. Multilevel Decoding for Very-Large-Size Dictionary Speech Recognition. IBM J. Res. Develop., Vol. 32, No. 2, March 1988.

F. Jelinek, R.L. Mercer. Interpolated Estimation of Markov Source Parameters from Sparse Data. Proc. Workshop Pattern Recognition in Practice, May 21-23, 1980. Amsterdam, The Netherlands. North-Holland.