A Hierarchical EM Approach to Word Segmentation

Fuchun Peng and Dale Schuurmans
Department of Computer Science
University of Waterloo
200 University Avenue West
Waterloo, Ontario, Canada, N2L 3G1
{f3peng,dale}@ai.uwaterloo.ca

Abstract

We propose a simple two-level hierarchical probability model for unsupervised word segmentation. By treating words as strings composed of morphemes/phonemes which are themselves composed of character/phone strings, we use EM to first identify the important morphemes/phonemes in a corpus, and then use a second level of EM to identify words given a lower level morpheme/phoneme segmentation. To further improve performance of the basic method we employ a mutual information criterion to eliminate long word agglomerations and reduce the size of the inferred lexicon while moving EM out of poor local maxima. Experiments on the Brown corpus show that our method accurately recovers hidden word boundaries using less training data than current MDL based approaches, even though our method is only trained on raw unsupervised data.

1 Introduction

Word segmentation is an important problem in many natural language processing tasks; for example, in speech recognition where there is no explicit word boundary information given within a continuous speech utterance, or in interpreting written languages such as Chinese, Japanese and Thai where words are not delimited by white-space but instead must be inferred from the basic character sequence.

Many unsupervised methods have been proposed for segmenting raw character sequences with no boundary information into words (Brent and Cartwright, 1996; Brent, 1999; Deligne and Bimbot, 1995; Christiansen, 1997; Christiansen, et al., 1998; de Marcken, 1995; Kit and Wilks, 1999; Hua, 2000). Brent (1999) gives a good survey of the area. Most current approaches are based on using some form of EM to learn a probabilistic speech or text model and then employing Viterbi-decoding-like procedures to segment new speech or text into words. One reason that EM is widely adopted for unsupervised learning is that it is guaranteed to converge to a good probability model that locally maximizes the likelihood or posterior probability of the training data (Dempster et al., 1977). For the problem of word segmentation, EM is typically applied by first extracting a set of candidate multi-grams from a given training corpus (Deligne and Bimbot, 1995), initializing a probability distribution over this set, and then using the standard iteration to adjust the probabilities of the multi-grams to increase the posterior probability of the training data.

There are at least three problems with this standard approach. First, there is a significant sparse data problem in training large multi-gram models. For example, if one wished to model all English words of length up to 15 characters, $26 + 26^2 + \cdots + 26^{15}$ multi-grams would have to be considered (in a naive model that only considered lower case characters). Although EM will assign zero probability to any unseen multi-gram—and therefore eliminates most of them—it still typically yields a very large lexicon. Due to the limited amount of training data it remains difficult to estimate a probability distribution that is defined by so many free parameters.

Second, because likelihood is usually defined by a product of individual chunk probabilities (making the standard assumption that segments are independent), the more chunks a segmentation has, the smaller its likelihood will tend to be. For example, given a character sequence *sizeofthecity* and a uniform distribution over multi-grams, the segmentation *sizeof*|*thecity* will have higher likelihood than segmentation *size*|*of*|*the*|*city*. Therefore, the maximum likelihood training procedure will prefer fewer chunks in its segmentation and thus tend to put a large probability on long non-word character sequences such as *sizeof, longtime* and *computerscience*. If one can break such sequences into short legal words, such as *size, of, long, time, computer* and *science*, then the lexicon will be much smaller and both training and segmentation should be improved.

Third, EM is known to have problems with getting trapped poor local maxima (Dempster et al., 1977) and often achieves results that depend strongly on the distribution from which it is initialized.

We propose two simple modifications of the standard approach to mitigate all three of these problems. Our first idea is based on the simple intuition that words themselves are built out of an intermediate vocabulary of morphemes (or phonemes in speech) which are in turn defined by shorter structured character strings. For example, in English, words like *carelessly* are composed of one to three morphemes, *care, less, ly*, which are themselves composed of one to five characters. To exploit this feature of natural text, we apply EM hierarchically in two phases: first to learn a morpheme lexicon, and second to learn a word lexicon over the base morpheme vocabulary. This hierarchical approach dramatically reduces the number of free parameters in our probability model by reducing the number of candidate multi-grams to $26 + 26^2 + \cdots + 26^5 + |G| + |G|^2 + |G|^3$ (where $|G|$ is the number of morphemes retained in our model)—which substantially reduces the problem of sparse training data. Although the idea of using morphemes/phonemes to detect word boundaries is not new (Brent and Cartwright, 1996; Brent, 1999; Christiansen, et al., 1998), previous work all assumes that the set of morphemes/phonemes is fixed beforehand and therefore learns a word model over an established morpheme/phoneme vocabulary. Our work is different in that we automatically learn the underlying morpheme/phoneme vocabulary from a training set of unsegmented character/phone strings.

Our second idea is to *prune* the learned morpheme and word lexicons. We do this by detecting long multi-grams that can be decomposed into shorter multi-grams without significantly reducing data likelihood, and delete these weakly connected long sequences from the lexicon. The rationale for lexicon pruning is not merely to cope with sparse data, but also to reduce excessive word agglomerations (second problem) and help guide EM out of poor local minima (third problem). As pointed out by Brand (1999) effective parameter pruning can help move EM into better subspaces and enable further progress.

2 Standard EM segmentation

Assume we have a sequence of characters $C = c_1c_2...c_T$ that we wish to segment into chunks $S = s_1s_2...s_M$, where the chunks s_i are chosen from a lexicon $V = \{s_i, i = 1, ..., |V|\}$. If we already have a probability distribution $\theta = \{\theta_i | \theta_i = p(s_i), i = 1, ..., |V|\}$ defined over the lexicon, then we can compute the most likely segmentation of the sequence $C = c_1c_2...c_T$ into chunks $S = s_1s_2...s_M$ as follows. First, for any given segmentation S of C, we can calculate the joint likelihood of S and C by

$$prob(S, C|\theta) = \prod_{i=1}^{M} \theta_i$$

Our task is to find the segmentation S^* that achieves the maximum likelihood:

$$S^* = \underset{S}{argmax}\{prob(S|C;\theta)\}$$

$$= \underset{S}{argmax}\{prob(S, C|\theta)\} \quad (1)$$

Viterbi decoding can be used to efficiently decode (1); *learning* these probabilities from a training corpus is the job of the EM algorithm. Following Dempster et al. (1977), the update Q function that we use here is

$$Q(k, k+1) = \sum_{S} prob(S|C;\theta^k) log(prob(C, S|\theta^{k+1})) \quad (2)$$

By maximizing (2) under the constraint that $\sum_i \theta_i^{k+1} = 1$, we obtain the parameter re-estimation formula

$$\theta_i^{k+1} = \frac{\sum_S \#(s_i, S) \times prob(S, C|\theta^k)}{\sum_{s_j} \sum_S \#(s_j, S) \times prob(S, C|\theta^k)} \quad (3)$$

Here the numerator is a sum over all possible segmentations S of the number of occurrences of a word s_i, weighted by the probability of the segmentation. Similarly, the denominator is a weighted sum of the number of words in all possible segmentations. Thus, (3) is a weighted frequency count.

3 Hierarchical EM Segmentation

The first modification of the basic procedure we employ is to use a two level, hierarchical version of EM for unsupervised text segmentation. In the first level, we generate all morphemes with one to five characters from the training corpus C, use

EM to learn a probability distribution over morphemes (as described above), prune low probability morphemes, and segment the original training corpus C into a morpheme sequence G. In the second level, we generate a large word lexicon from G (multi-grams over morphemes), use EM to learn a probability distribution over words, and segment G into a word sequence W. The complete process is illustrated in Figure 1. In both phases, once EM converges, we employ additional lexicon pruning (discussed in the next section) which pulls EM out of local maxima and allows it to make further progress. Overall, the first level determines

$$G^* = \underset{G}{argmax}\{prob(G, C|\theta_g)\} \quad (4)$$

and the second level determines

$$W^* = \underset{W}{argmax}\{prob(W, G|\theta_w)\} \quad (5)$$

where θ_g and θ_w are the distributions over morpheme lexicon and word lexicon respectively. The EM algorithms in both levels are identical except that in the first level the basic observation unit is character and in the second level the basic unit is morpheme.

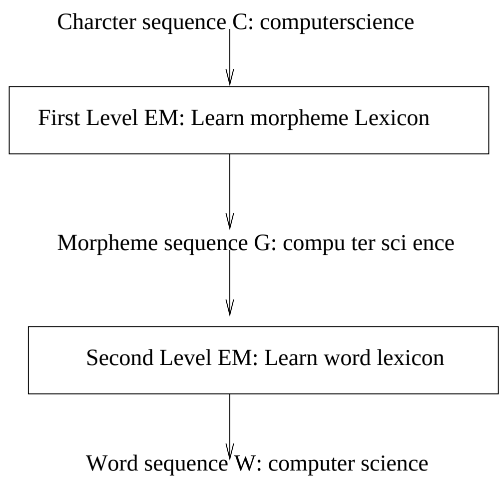

Figure 1: Hierarchical EM segmentation model

4 Lexicon pruning

Although a hierarchical training scheme mitigates some of the sparse data problems encountered in this problem, a naive application of EM still runs into the two other problems we identified: excessive agglomeration and poor local minima. To cope with both of these, we employ a simple lexicon pruning scheme that eliminates long agglomerations of short primitives.

Recall that the mutual information between two random variables X and Y is defined as

$$MI = log\frac{P(X,Y)}{P(X) \times P(Y)} \quad (6)$$

where a large value indicates strong dependence and zero indicates independence. To implement our pruning criteria we use a variant of this formula to evaluate the cohesiveness of strings. Specifically, given a long string s we consider splitting it into the two substrings s_1 and s_2 that maximize $p(s_1) \times p(s_2)$ over all two-chunk segmentations $s = s_1 s_2$. Let the probabilities of the original string and the two chunks be $p(s)$, $p(s_1)$ and $p(s_2)$ respectively. We define the *pointwise mutual information* (Manning and Schütze, 1999) between s_1 and s_2 to be

$$pMI = \frac{1}{T} \times log\frac{p(s)}{p(s_1) \times p(s_2)}. \quad (7)$$

where T is length of s. To apply this measure to pruning, we set two thresholds $\gamma_1 > \gamma_2$. If the mutual information is higher than the high threshold γ_1 we say that s_1 and s_2 are strongly correlated and do not split s. (That is, we do not remove s from the lexicon.) If the mutual information is lower than the lower threshold γ_2 we say that s_1 and s_2 are nearly independent, so we remove s from the lexicon and distribute its probability to s_1 and s_2. If the mutual information is between the two thresholds we say that s_1 and s_2 are weakly correlated and therefore shift some of the probability from s to s_1 and s_2 by keeping a portion of s's probability for itself (1/3 in experiments) and distribute its rest probability to the smaller chunks proportional to their probabilities. The idea is to shift the weight of the probability distribution toward shorter words. This lexicon pruning scheme works efficiently in Chinese segmentation (Peng and Schuurmans, 2001).

5 Experiments

We tested our procedure on segmenting the Brown corpus into words. Specifically, we converted the corpus to lowercase letters and removed all whitespace and punctuation. We then split the corpus into a training sequence C_1 (4292K characters, 891524 words, 37930 unique words) and a test sequence C_2 (317K characters; 64338 words; 9506 unique words, 2280 of which never occur in the training sequence). For pruning we used the thresholds $\gamma_1 = 3$ and $\gamma_2 = 0.5$.

5.1 Performance measures

We measured performance on the test corpus by recovering recall, precision and F-measures with respect to detecting the word boundaries in a test sequence (Kit and Wilks, 1999). A predicted word boundary which corresponds to white space or punctuation in the original test text is considered correct. Let N_1 denote the true number of word boundaries in the original test text, let N_2 denote the number of predicted boundaries, of which N_3 are correct. Then the precision, recall and F-measure are defined by

$$\text{precision: } p = \frac{N_3}{N_2}$$
$$\text{recall: } r = \frac{N_3}{N_1}$$
$$\text{F-measure: } F = \frac{2 \times precision \times recall}{precision + recall}$$

We also report the word identification performance on the test text. A word is said to be correctly recovered if and only if it has correct boundaries predicted immediately before it, immediately after it, and no predicted boundaries in between (Brent and Cartwright, 1996; Brent, 1999; Christiansen, et al., 1998).

5.2 Results of the first level model

We first train the morpheme model on the training sequence C_1 with multi-grams of lengths 1–5 characters using EM. With this morpheme lexicon, we then segmented the training sequence C_1 into a morpheme string G_1 using the Viterbi algorithm. With this initial morpheme lexicon in hand, we then conducted several stages of our recursive pruning procedure to remove large morphemes from the lexicon. For each pruned lexicon, EM was run again to re-estimate the probability parameters, and each model was then used to re-segment the training sequence.

Each of these models was tested on the separate test sequence C_2 by running Viterbi to segment it into a morpheme string G_2 using the learned models. Table 1 shows the results of using the various learned probability models to predict word boundaries in the test sequence C_2 (where # represents the size of the lexicon). Here, each reduced and trained model was used to segment the test sequence using Viterbi, and the morpheme boundaries were used to predict word boundaries in the initial text. Clearly, this segmentation should not work very well because proper morphemes only comprise sub-components of words. Nevertheless, we should see a high recall score (every word boundary corresponds to a morpheme boundary) along with a mediocre precision score—which is exactly what Table 1 shows.

5.3 Results of the second level model

Using the best morpheme model learned in the first level (G Pruned 5), we then trained the second level model over the Viterbi segmentation of

	#	p	r	F
G Start	13506	0.603	0.799	0.678
G Pruned 1	8274	0.595	0.867	0.695
G Pruned 2	7793	0.590	0.873	0.693
G Pruned 3	7625	0.589	0.876	0.694
G Pruned 4	6800	0.576	0.892	0.689
G Pruned 5	6758	**0.570**	**0.908**	**0.690**

Table 1: Test segmentation results with the first level (morpheme) model showing boundary detection scores

the training sequence, G_1. Using EM we learn a lexicon of words composed of morpheme strings and a probability distribution over these words. Viterbi decoding of the training and test morpheme strings, G_1 and G_2, yields the word segmentations W_1 and W_2. As above, we successively prune the word lexicon on the training sequence G_1 and re-estimate the probability distribution over words in the training sequence using EM. For each successive word model, we segment the training and test morpheme strings G_1 and G_2 into word strings W_1 and W_2. The results of applying Viterbi using the learned model to segment the test sequence G_2 into words W_2 are shown in Table 2.

	#	p	r	F
Start	812841	0.806	0.670	0.716
W Pruned 1	278608	0.754	0.720	0.721
W Pruned 2	224619	0.743	0.731	0.721
W Pruned 3	198723	**0.746**	**0.792**	**0.754**
W Pruned 4	181316	0.733	0.806	0.753
W Pruned 5	174806	0.725	0.806	0.749

Table 2: Test segmentation results with the second level (word) model showing boundary detection scores

In addition to these boundary detection results we also measured the word detection performance of the best model (W pruned 3) on the test sequence, obtaining **49.1%** word precision, **60.1%** word recall and **53.8%** F-measure.

5.4 Results of a flat model

To verify the effectiveness of our hierarchical approach, we re-ran the experiments using a basic flat model that is similar to the first level of our hierarchical model. Here, we generated all words up to 15 characters from the training sequence C_1, ran EM to learn a probability model over words, and tested the model by segmenting the test sequence C_2. The results for the original trained model as well as successively pruned versions are shown in Table 3.

Using this flat one level model, we obtained a

	#	p	r	F
Start	8578589	0.727	0.556	0.607
F Pruned 1	1316725	**0.846**	**0.513**	**0.620**
F Pruned 2	1232529	0.713	0.515	0.583

Table 3: Test segmentation results with the flat model showing boundary detection scores

best word detection performance (F Pruned 1) of: **32.4%** word precision, **48.1%** word recall and **38.1%** F-measure, which is substantially below that obtained by the hierarchical model.

5.5 Interpreting the results

The weakest results we obtained with the hierarchical model are those with no lexicon pruning: 80.6% precision, 67% recall and 71.6% F-measure. The best results were achieved after applying pruning to both the morpheme and word models. The combination G-Pruned-5 and W-Pruned-3 yielded 74.6% precision, 79.2% recall and F-measure 75.4%.

Compared to the flat model, the hierarchical model gains **13.4%** improvement on boundary detection F-measure and **15.7%** on word detection F-measure. The best known results on segmenting the Brown corpus that we are aware of are due to Kit and Wilks (1999) who use a description length gain method. They trained their model on the whole corpus (6.13M) and reported results on the *training* set, obtaining a boundary precision of 79.33% and a boundary recall of 63.01% (they did not report boundary F-measure, but we can calculate it to be 70.23% in this case). By comparison, we train our model on a much smaller subset of the corpus (4292K) and test on *unseen* data. Even the weakest (unpruned) results of the hierarchical model are better than those reported in (Kit and Wilks, 1999). After the lexicon is optimized, we obtain 16.19% higher recall and 4.73% lower precision; resulting in an improvement of **5.2%** in boundary F-measure.

De Marcken (1995) also uses an minimum description length (MDL) framework and a hierarchical model to learn a word lexicon from raw speech. However, this work does not explicitly yield word boundaries, but instead recursively decomposes an input string down to the level of individual characters. As pointed out by Brent (1999), this study gives credit for detecting a word if any node in the hierarchical decomposition spans the word. Under this measure (de Marcken, 1995) reports a word recall rate of 90.5% on the Brown corpus. However, his method creates numerous chunks and therefore only achieves a word precision rate of 17%.

Christianson et al. (1998) use a simple recurrent neural network approach and report a word precision rate of 42.7% and word recall rate of 44.9% on spontaneous child-directed British English.

Brent and Cartwright (1996) use an MDL approach and report a word precision rate of 41.3% and a word recall rate of 47.3% on the CHILDES collection. More recently, Brent (1999) achieves improved results (about 70% word precision and 70% word recall) by employing additional language modeling and smoothing techniques.

The best word recognition performance we obtain is 49.1% word precision and 60.1% word recall, hence 53.8% word F-measure on the Brown corpus. This is better than (Christiansen, et al., 1998; Brent and Cartwright, 1996) but worse than (Brent, 1999). However, it is difficult to draw a direct comparison between these results because of the different test corpora used. Nevertheless, our results seem to support the utility of exploiting a simple hierarchical model for word recognition.

6 Related Work

Our work is related to many other research efforts.

Word detection and lexical acquisition:

The work of (Kit and Wilks, 1999; Hua, 2000; Brent and Cartwright, 1996; Brent, 1999) are all based on the MDL principle, but differ in how the description length is encoded. Kit and Wilks (1999) uses a description length gain measure, but does not use hierarchical structure nor EM to learn a lexicon. (Brent and Cartwright, 1996; Brent, 1999) use Huffman codes to describe words and use a generative model in which the size of lexicon is predefined and then use this lexicon to generate observations. Our model uses a simple lexicon pruning scheme to automatically determine the size of the lexicon.

(Brent and Cartwright, 1996; Brent, 1999; Christiansen, et al., 1998) test their algorithms on phonemically transcribed corpora (the CHILDES collection and spontaneous child-directed British English) but in practice the phonemes are not explicitly identified in the utterances, and therefore the basic unit in speech is the phone. Here it is also necessary to detect phonemes automatically from a phone sequence. In essence, this corresponds to the second level of our model: learning words given phonemes.

Hierarchical models for sparse data: Many authors have proposed hierarchical structures to reduce the effects of sparse training data. Frietag and McCallum (1999) use hierarchical models to regularize hidden Markov model emission probability estimates. Slonim and Tishby (2000) use a similar two-step method to cluster documents. In this work, Slonim and Tishby cluster words in a document first, and then cluster documents based on the word clusters obtained from the first step. The main technique for exploiting a hierarchy in these cases is to share common information between related clusters of data. As long as there is common information, a hierarchical structure should prove beneficial. In our case, the common information is the morphemes that are shared between words.

7 Conclusions and future work

We presented a two-level hierarchical EM approach to word segmentation and word discovery by exploiting the internal structure of English words. The hierarchical structure we impose is natural and effectively deals with the sparse training data problem. Our model can learn phonemes and words completely automatically. We also use a lexicon pruning method based on mutual information which makes the lexicon more compact and helps guide EM out of local training maxima. We tested our model on the Brown corpus and obtained a noticeable improvement over MDL based methods on the same data.

Overall, these results show the potential advantage of hierarchical models for speech segmentation and text-to-speech synthesis, as well as compound and phrase detection in natural language processing.

There remain many open questions. As shown in many other areas of research, MDL or Bayesian estimators often yield better models than a straightforward maximum likelihood approach. It is therefore worthwhile to consider imposing a hierarchical model on description length gain (Kit and Wilks, 1999) or exploit a prior in EM training. Also, our use of EM sets the probability of all non-occurring words in the training sequence to zero. As shown in (Brent, 1999), it would be beneficial to employ language modeling and smoothing techniques.

8 Acknowledgments

We would like to thank the Waterloo Statistical NLP group and the anonymous referees for their helpful comments. Research supported by Bell University Labs, MITACS and NSERC.

References

Brand, M. 1999. Structure learning in conditional probability models via an entropic prior and parameter extinction. In *Neural Computation 11*, 1155-1182.

Brent, M. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning 34*, 71–106.

Brent, M. and Cartwright, T. 1996. Distributional regularity and phonotactics are useful for segmentation. *Cognition 61*, 93-125.

Christiansen M. and Allen, J. 1997. Coping with Variation in Speech Segmentation. In *Proceedings of GALA 1997: Language Acquisition: Knowledge Representation and Processing*, 327-332.

Christiansen, M., Allen, J. and Seidenberg, M. 1998. Learning to Segment Speech Using Multiple Cues: A Connectionist Model. *Language and Cognitive Processes 13*, 221-268.

Deligne, S. and Bimbot, F. 1995. Language Modeling by Variable Length Sequences: Theoretical Formulation and Evaluation of Multigrams. In *Proceedings ICASSP.*

de Marcken, C. 1995. The Unsupervised Acquisition of a Lexicon from Continuous Speech. *Technical Report AI Memo No. 1558, M.I.T., Cambridge, Massachusetts.*

Dempster, A., Laird, N. and Rubin, D. 1977. Maximum-likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. Ser. B 39.*

Frietag, D. and McCallum, A. 1999. Information Extraction with HMMs and Shrinkage. In *AAAI'99 Workshop on Machine Learning for Information Extraction.*

Hua, Y. 2000. Unsupervised word induction using MDL criterion. In *Proceedings ISCSL2000*, Beijing.

Kit, C. and Wilks, Y. 1999. Unsupervised Learning of Word Boundary with Description Length Gain. In *Proceedings CoNLL99 ACL Workshop.* Bergen.

Manning, C. and Schütze, H. 1999. Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, Massachusetts.

Peng, F. and Schuurmans, D. 2001. Self-supervised Chinese Word Segmentation. In *Proceedings of the 4th Internation Symposium on Intelligent Data Analysis(IDA2001).*

Slonim, N. and Tishby, N. Document Clustering using Word Clusters via the Information Bottleneck Method. In *Proceedings SIGIR-2000.*