Bootstrap Voting Experts

Daniel Hewlett
University of Arizona
Tucson, AZ, USA
dhewlett@cs.arizona.edu

Paul Cohen
University of Arizona
Tucson, AZ, USA
cohen@cs.arizona.edu

Abstract

BOOTSTRAP VOTING EXPERTS (BVE) is an extension to the VOTING EXPERTS algorithm for unsupervised chunking of sequences. BVE generates a series of segmentations, each of which incorporates knowledge gained from the previous segmentation. We show that this method of bootstrapping improves the performance of VOTING EXPERTS in a variety of unsupervised word segmentation scenarios, and generally improves both precision and recall of the algorithm. We also show that Minimum Description Length (MDL) can be used to choose nearly optimal parameters for VOTING EXPERTS in an unsupervised manner.

1 Introduction

Word segmentation can be considered as a uniquely linguistic phenomenon, or it can be regarded as an instance of a much more general cognitive ability: chunking. Intuitively, chunking is simply the process of identifying sequences of things that "go together." The key to chunking then, is to specify precisely what it means for a sequence of things to go together.

In designing the VOTING EXPERTS algorithm, Cohen and Adams [2001] provided a formal answer to this question: Chunks are sequences that have low *internal entropy*, and high *boundary entropy*, meaning that items within a chunk can predict one another, but not items outside the chunk. This answer, and hence VOTING EXPERTS, can be applied to a wide variety of sequential domains, including word segmentation but also including such diverse areas as robot actions and visual character recognition [Miller and Stoytchev, 2008].

This paper improves VOTING EXPERTS in two ways. The first is a bootstrapping method that improves segmentation performance by iteratively reusing high-precision segmentations. We show that VOTING EXPERTS can generate high-precision segmentations, and that the information gained from these segmentations can productively be incorporated into future segmentations. The second is an application of the Minimum Description Length (MDL) technique for parameter setting, to eliminate the few parameters that VOTING EXPERTS has. We also revisit experiments by Cohen et al. [2006] comparing VOTING EXPERTS and another algorithm, MBDP-1, and provide a clearer picture of the relative performance of the two algorithms.

2 Voting Experts

The VOTING EXPERTS algorithm can be described quite simply. The algorithm's name refers to the two "experts" that vote on possible boundary locations: One expert votes to place boundaries after sequences that have low internal entropy, given by $H_I(seq) = -log(p(seq))$, the other places votes after sequences that have high boundary entropy, given by $H_B(seq) = -\sum_{c \in S} p(c|seq) log(p(c|seq))$, where S is the set of successors to seq. All sequences are evaluated locally, within a sliding window, allowing the algorithm to be very efficient.

The statistics required to calculate H_I and H_B are stored efficiently using an n-gram trie, which is constructed in a single pass over the corpus. The trie depth is 1 greater than the size of the sliding window. Importantly, all statistics in the trie are normalized so as to be expressed in standard deviation units. This allows statistics from sequences of different lengths to be compared to one another.

The sliding window is then passed over the corpus, and each expert votes once per window for the boundary location that best matches that expert's criteria. After voting is complete, the algorithm yields an array of vote counts, each element of which is the number of times some expert voted to segment at that location. The result of voting on the string `thisisacat` could be represented in the following way, where the number in between each letter is the number of votes that location received, as in `t0h0i1s3i1s4a4c1a0t`.

With the final vote totals in place, the final segmentation consists of locations that meet two requirements: First, the number of votes must be locally maximal (this is called the *zero crossing rule*). Second, the number of votes must exceed a chosen *threshold*. Thus, VOTING EXPERTS has three parameters: the window size, the vote threshold, and whether to enforce the zero crossing rule. We will return to these parameters in Section 7. For further details of the VOTING EXPERTS algorithm see [Cohen *et al.*, 2006], and also [Miller and Stoytchev, 2008].

3 Related Work

Since its introduction in 2001 by Cohen and Adams, VOTING EXPERTS has been extended more than once by the addition of a third expert, the natural method of extending VOTING EXPERTS. The first of these was the Markov Expert [Cheng and Mitzenmacher, 2005], which uses a model of the quality of a cut point to refine the results of the other two experts. The third expert added by HVE-3E [Miller and Stoytchev, 2008] works as a two pass system: the first pass is a standard run of VOTING EXPERTS, and on the second pass this third expert votes for locations following frequent chunks from the first pass.

Much of the other work in unsupervised segmentation exists specifically within the context of word segmentation. Tanaka-Ishii and Jin [2006] developed a method based on boundary entropy alone, analogous to VOTING EXPERTS without the internal entropy expert. A separate line of work is based on bootstrapping, the principle of using previously discovered words to segment future utterances, and begins with Brent's MBDP-1 [1999], a Bayesian, dynamic programming algorithm discussed in further detail in Section 6.1. Venkataraman [2001] refined MBDP-1 by simplifying some of its mathematical assumptions. Several more recent algorithms have been developed that use probabilistic language modeling frameworks similar to Brent's, including Goldwater et al.'s HDP [2008] and Fleck's WORDENDS [2008].

4 Bootstrap Voting Experts

BOOTSTRAP VOTING EXPERTS proceeds by segmenting the corpus repeatedly, with each new segmentation incorporating knowledge gained from previous segmentations. As with many bootstrapping methods, three essential components are required: some initial seed knowledge, a way to represent knowledge, and a way to leverage that knowledge to improve future performance. For BOOTSTRAP VOTING EXPERTS, the seed knowledge consists of a high-precision segmentation generated by VOTING EXPERTS, as described in Section 4.1. Knowledge gained from prior segmentations is represented in a data structure we call the *knowledge trie*, described in Section 4.2. During voting, this knowledge trie provides statistics for the *Knowledge Expert*, a third expert added to VOTING EXPERTS, which is presented in Section 4.3. Details of the main bootstrapping loop itself are given in Section 4.4.

Figure 1 below illustrates how precision and recall typically change during bootstrapping. The final result is typically a segmentation with a higher F-measure, recall, and precision than VOTING EXPERTS.

4.1 Initial Seed Segmentation

Generating some initial high-precision cut points is easy with VOTING EXPERTS, because higher threshold values trade-off recall for precision. A single high-threshold execution of VOTING EXPERTS would suffice, and as a rule of thumb a threshold value equal to the window size produces precision above 90%. However, higher precision can be obtained with a slightly more sophisticated method: First, a high-precision execution of VOTING EXPERTS generates a set of "forward" cut points F. Reversing the corpus, and segmenting the reversed text with VOTING EXPERTS, generates a different set of "backward" cut points, B. This is because the trie constructed on the reversed corpus will contain different statistics than the standard corpus. By taking the intersection $F \cap B$, the result is a set of high-precision cut points where the forward and backward segmentations agree.

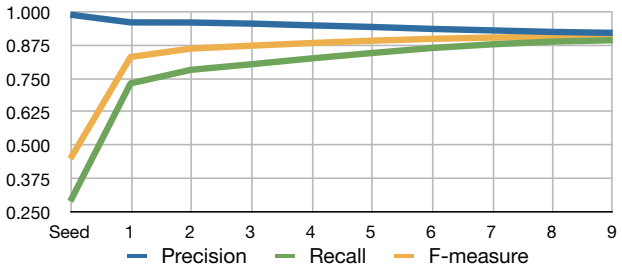

Figure 1: A characteristic run of BOOTSTRAP VOTING EXPERTS, showing recall increasing faster than precision decreases, with each iteration. Results are from a CHILDES corpus, discussed in the following section.

4.2 The Knowledge Trie

A *knowledge trie* is simply a trie that explicitly stores word boundaries. The BOOTSTRAP VOTING EXPERTS knowledge trie is built in the standard manner, except that word boundaries are treated as a character in the alphabet. A portion of the knowledge trie for the sequence `#the#cat#sat#on#the#mat#` (`#` represents the boundary character) is shown below in Figure 2. While this example includes the full set of correct boundaries, the knowledge trie in BOOTSTRAP VOTING EXPERTS can only store boundaries that were *discovered* by an earlier iteration of the algorithm, as no supervision is provided. This trie can serve as a prefix trie and a suffix trie simultaneously: the frequency of a sequence such as `at` in word-initial position is given by `#at`, and its word-final frequency is given by `at#`. The only statistic the knowledge trie must support for BOOTSTRAP VOTING EXPERTS is the internal entropy of each sequence, which is computed and normalized exactly as in the main VOTING EXPERTS trie.

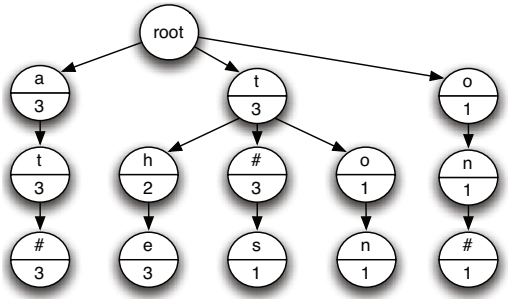

Figure 2: A portion of the knowledge trie built from `#the#cat#sat#on#the#mat#`. Numbers within each node are frequency counts.

4.3 The Knowledge Expert

To incorporate the information stored in the knowledge trie into the segmentation process, a third expert is added to VOTING EXPERTS. This new expert is called the Knowledge Expert, and votes for locations that are the most likely to be word boundaries given the boundary information stored in the knowledge trie. For each window under consideration, it considers potential boundary locations within the window explicitly as word boundaries. For a given split, say `it|was`, the sequence before the split is considered as the end of a word (as in `it#`), and the sequence after the split is considered as the beginning of a word (as in `#was`). The internal entropy of the "before" segment, $H_I(\text{before\#})$ and of the "after" segment, $H_I(\text{\#after})$, are stored in the knowledge trie. Thus, each calculation is simply a look-up into the knowledge trie. The Knowledge Expert votes for the location where the sum of these two internal entropies is minimal, as shown in Equation (1).

$$\underset{\text{before,after}}{\operatorname{argmin}} \big(H_I(\text{before\#}) + H_I(\text{\#after}) \big) \tag{1}$$

4.4 Iteration

For the first bootstrapping iteration, the knowledge trie for the Knowledge Expert is built from the initial seed segmentation. Each subsequent iteration i uses the segmentation generated by iteration $i-1$ to populate its knowledge trie, meaning that knowledge gained by previous iterations is conserved. This knowledge is also augmented, since the vote threshold is lowered by 1 with each iteration, effectively backing off from the initial high precision segmentations by increasing recall. Iteration continues in this manner until a minimum threshold value is reached. This minimum threshold must be specified as an additional parameter to BOOTSTRAP VOTING EXPERTS.

5 Results

We now present results detailing the performance of BOOTSTRAP VOTING EXPERTS on several corpora, as well as comparisons with VOTING EXPERTS, and other algorithms where possible. The primary metric for evaluating segmentation quality is the boundary F-measure:

$$\text{F} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2}$$

where precision is the percentage of the induced boundaries that are correct, and recall is the percentage of the correct boundaries that were induced. When comparing BOOTSTRAP VOTING EXPERTS against other algorithms, we will denote the boundary-based precision, recall, and F-score as BP, BR, and BF, respectively, because we will also provide the word-based precision, recall, and F-score (WP, WR, and WF, respectively).

Cohen et al. [2006] already demonstrated that VOTING EXPERTS performance is superior to a common compression-based algorithm, SEQUITUR [Nevill-Manning and Witten, 1997], as well as random and ALL-LOCATIONS baselines. We report results for the ALL-LOCATIONS method, which simply segments the corpus by cutting at every possible location, as a baseline only because it provides a quick reference for the complexity of the domain. All results reported for VOTING EXPERTS and BOOTSTRAP VOTING EXPERTS (as well as the ALL-LOCATIONS baseline) were obtained through direct execution of those algorithms; results from other algorithms are taken from published figures, except for the HVE-3E results, which were obtained by executing the source code generously provided by Miller and Stoytchev.

5.1 Orthographic English

Every paper about VOTING EXPERTS has included an evaluation of performance on some portion of George Orwell's *1984*. We continue this tradition here, by presenting results on the first 50,000 characters of *1984*, shown in Figure 3.

Algorithm	BP	BR	BF	WP	WR	WF
VE	0.817	0.731	0.772	0.505	0.452	0.477
BVE	**0.832**	**0.792**	**0.812**	**0.556**	**0.529**	**0.542**
HVE-3E	0.800	0.769	0.784	-	-	-
Markov Exp.	0.809	0.787	0.798	-	-	-
All Locations	0.185	1.000	0.313	0.008	0.041	0.013

Figure 3: Results obtained for Orwell's *1984*, as well as results from selected other algorithms. Dashes indicate a score that was not reported in published results. The highest performance figure in each column (other than ALL-LOCATIONS) is highlighted in bold.

5.2 Phonetic CHILDES

CHILDES [MacWhinney and Snow, 1985] is a set of transcribed conversations between caretakers (typically mothers) and young children. Here we examine the performance of BOOTSTRAP VOTING EXPERTS on a corpus used by several word segmentation researchers, called BR87. For the purposes of this paper, BR87 represents not only the mother's utterances from the portion of CHILDES compiled by Bernstein Ratner [1987], but also a particular phonemic encoding of those utterances owing to Brent [1999]. The BR87 corpus has been used not only by Brent, but also by other researchers inspired by Brent's work, including Goldwater et al. [2008], and Fleck [2008], allowing for direct comparison with these other published results (shown below in Figure 4).

5.3 Orthographic Chinese

Chinese is traditionally written without spaces between words, though today most written Chinese includes some punctuation. Because of this, Chinese presents an excellent real-world challenge for unsupervised segmentation. The results shown in Figure 5 were obtained using the first 100,000 words of the Chinese Gigaword corpus [Huang, 2007].

6 The Incremental Paradigm

All versions of VOTING EXPERTS are trained on a single text with all sentence and word boundaries removed. However, other researchers often train their algorithms by presenting

Algorithm	BP	BR	BF	WP	WR	WF
VE	0.867	0.854	0.860	0.652	0.642	0.647
BVE	0.928	**0.905**	**0.916**	**0.791**	**0.794**	**0.793**
MBDP-1[i]	0.803	0.843	0.823	0.670	0.694	0.682
HDP[i]	0.903	0.808	0.852	0.752	0.696	0.723
WordEnds[i]	**0.946**	0.737	0.829	-	-	0.707
All Locations	0.258	1.000	0.411	0.013	0.051	0.021

Figure 4: Results obtained for the BR87 corpus, as well as published results from selected other algorithms. Algorithms marked with [i] operate within the incremental paradigm, discussed in Section 6.

Algorithm	BP	BR	BF	WP	WR	WF
VE	**0.890**	0.836	0.862	0.670	0.630	0.649
BVE	0.871	**0.900**	**0.885**	**0.695**	**0.717**	**0.706**
All Locations	0.575	1.000	0.730	0.139	0.381	0.204

Figure 5: Results obtained for the Chinese Gigaword corpus.

sentences from the text incrementally, a condition we will refer to as the *incremental paradigm*. Algorithms operating in the incremental paradigm are provided two "free" word boundaries per sentence (the ones at the beginning and the end of the sentence), which essentially makes the segmentation problem *semi-supervised*.

6.1 MBDP-1 Revisited

In their 2006 paper [Cohen *et al.*, 2006], Cohen et al. present a comparison of VOTING EXPERTS with MBDP-1, a bootstrapping word segmentation algorithm due to Brent [1999]. On the basis of this comparison, they conclude that MBDP-1 is able to outperform VOTING EXPERTS for large corpora. However, the results for MBDP-1 presented by Cohen et al. were taken from Brent et al.'s study of the impact of *supervised* training on the performance of MBDP-1 [Brent and Tao, 2001], while VOTING EXPERTS was operating in a fully unsupervised environment. With a sufficiently large amount of supervised training, MBDP-1 was able to achieve close to perfect performance. Thus, the performance measures used were not comparable because the two algorithms were working under very different conditions.

To provide a fairer test, we developed incremental versions of both VOTING EXPERTS and BOOTSTRAP VOTING EXPERTS, and compared them to MBDP-1. The same requirements were present for all algorithms: No hand-segmented text was given, and each sentence must be segmented before the following sentence can be analyzed. Thus, the information available for the segmentation of the current sentence is the set of previous utterances and their proposed segmentations. The adaptation of both versions of VOTING EXPERTS to the incremental paradigm was straightforward: Each sentence was segmented using whatever prior knowledge was available, and then the sentence (and its segmentation, for BOOTSTRAP VOTING EXPERTS) could be analyzed to construct tries and compute statistics. All algorithms segmented the same corpus, the full text of Orwell's *1984*. The results of this comparison are shown below in Figure 6.

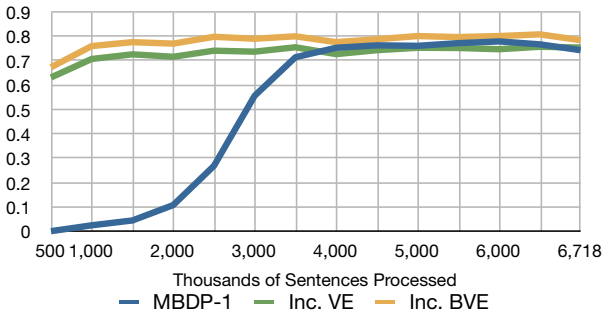

Figure 6: Results of comparison in the incremental paradigm. Each point represents the performance (boundary-based F-score) of that algorithm on the previous 500 sentences.

Both versions of VOTING EXPERTS start out with relatively high performance and stabilize after only 2,500 sentences, while MBDP-1 slowly builds performance and needs about 4,000 sentences to approach its performance plateau. After this point, VOTING EXPERTS and MBDP-1 are in close competition, with BOOTSTRAP VOTING EXPERTS leading both of them. This result is very different from Cohen et al's comparison, where MBDP-1 pulls away from VOTING EXPERTS [Cohen *et al.*, 2006].

7 Parameter Setting

In describing VOTING EXPERTS, we noted that the algorithm requires specification of a window size, a vote threshold, and whether to apply the zero crossing rule. Each of these choices can affect the performance of the algorithm in potentially serious ways. Here we consider the effects of these parameters, and propose a way to set them automatically, using a method called *Minimum Description Length*, or MDL.

Cohen et al. [2006] investigated the effect of the window size parameter in English and found that larger window sizes reduced the proportion of "lost" words, words for which neither boundary was recovered. Unfortunately, no general relationship between window size and performance holds across languages. For example, in Chinese, where words are typically much shorter than in English, performance peaks at a window size of 3. The effect of the threshold parameter is more straightforward: increasing the threshold produces higher precision and lower recall, so the threshold provides a powerful way to tune the performance of VOTING EXPERTS. In BOOTSTRAP VOTING EXPERTS, this parameter is manipulated for exactly this reason, to slowly back-off from very high precision while increasing recall. An illustration of the impact of the threshold parameter on VOTING EXPERTS, for a given window size, is shown in Figure 7.

Recall that the zero crossing rule simply requires that a given cut point have a locally maximal number of votes, meaning that it must have a greater number of votes than both neighboring cut positions. This constraint often has the positive effect of preventing over-segmentation. However, the zero crossing rule also prevents any single-character words

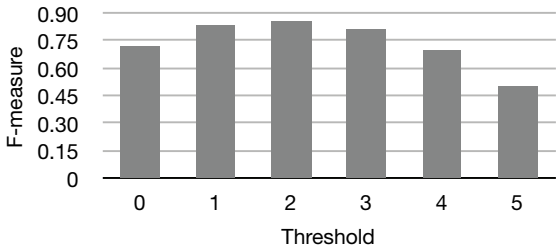

Figure 7: Effects of the threshold parameter on overall performance. Results are from the BR87 corpus with a window size of 4.

from being found. For English, this sacrifice may be acceptable, but for languages like Chinese, where many words consist of a single character, the zero crossing rule will place a low ceiling on performance. Thus, we can consider the presence of the zero crossing rule to be a boolean parameter to VOTING EXPERTS.

Minimum Description Length (MDL) provides an unsupervised way to set these parameters indirectly by selecting among the segmentations each combination of parameters would generate. The *Description Length* for a given hypothesis and data set refers to the number of bits needed to represent both the hypothesis and the data given that hypothesis. The *Minimum* Description Length, then, simply refers to the principle of selecting the hypothesis that minimizes description length. In this context, the data is a corpus (sequence of symbols), and the hypotheses are proposed segmentations of that corpus, each corresponding to a different combination of parameter settings. Thus, we choose the vector of parameter settings that generates the hypothesized segmentation which has the minimum description length.

The method for computing MDL employed here is based on that of Argamon et al. [2004]. The intuition driving this formulation is that any segmentation, by specifying boundaries, also implicitly specifies a lexicon of words. The corpus can then be encoded as a list of words in this lexicon. The basic description length formula is shown below in Equation (3), where L is the lexicon and Data is the corpus. The CODE function represents the minimal number of bits required to encode its argument. Thus, $\text{CODE}(\text{Data}|L)$ is the minimum number of bits required to encode the corpus using the inferred lexicon.

$$\text{CODE}(\text{Data}|L) + \text{CODE}(L) \tag{3}$$

The formula for the information cost of the lexicon is:

$$\text{CODE}(L) = b \sum_{w \in L} \text{length}(w) \tag{4}$$

where b is the number of bits required to encode a single character in the language's alphabet, and $\text{length}(w)$ is the length of word w in characters.

The corpus is treated as a list of N words, of the form $\text{Data} = w_3 w_1 w_5 w_2 w_2 w_3 \ldots$, where each w_j represents an instance of the jth word of the lexicon L. The information cost of the corpus given the lexicon, $\text{CODE}(\text{Data}|L)$, is:

$$-\sum_{j=1}^{|L|} C(w_j)(\log(C(w_j)) - \log(N)) \tag{5}$$

where $C(w_j)$ represents the number of instances of word w_j the corpus contains. See Argamon et al. [2004] for a detailed derivation of these equations.

With the MDL calculation now fully specified, we can estimate the best vector of parameter values for the window size, threshold, and zero crossing rule, $\mathbf{p} = \langle w, \theta, z \rangle$, by solving:

$$\underset{\mathbf{p}}{\text{argmin}} \big(\text{CODE}(\text{Data}|L_{\mathbf{p}}) + \text{CODE}(L_{\mathbf{p}})\big) \tag{6}$$

where $L_{\mathbf{p}}$ is the lexicon implicitly generated by VOTING EXPERTS with the parameters set to $\mathbf{p}$.

7.1 MDL Results

To evaluate the ability of MDL to select effective parameters for VOTING EXPERTS, we generating a set of candidate segmentations by testing a wide range of parameter settings, and compared the boundary F-measure of the segmentation chosen by MDL to the highest boundary F-measure among the candidates. For the parameter settings, we considered every window size between 2 and 9 (inclusive), every threshold value between 0 and the current window size (inclusive), and both with and without the zero crossing rule. This yields a total of 104 unique parameter vectors. We tested VOTING EXPERTS with each of these parameter vectors on several testing corpora, generating 104 segmentations for each corpus. The segmentation with the minimal description length is called the *MDL* segmentation. Each of the segmentations has an associated F-measure, and we will refer to the one with the highest F-measure as the *best* segmentation. By comparing the F-measure of the MDL-selected segmentation with that of the best segmentation, we can determine what percentage of the best performance can be recovered by using MDL to choose between the segmentations. The results of this comparison are shown below in Figure 8.

Corpus	Mean BF	MDL BF	Best BF	Percent of Best
Orwell	0.6555	0.7807	0.7816	99.88%
Twain	0.6710	0.7845	0.8161	96.13%
Chinese GW	0.5112	0.8646	0.8688	99.52%
Nietzsche	0.6683	0.7940	0.7940	100.00%
Caesar	0.6262	0.7721	0.7776	99.29%
BR87	0.6881	0.7969	0.8734	91.24%
Bloom73	0.6970	0.8527	0.8722	97.76%
Gray	0.7728	0.9075	0.9277	97.82%

Figure 8: Quality of the MDL-selected segmentation.

All of the corpora introduced in Section 5 (Orwell's *1984*, Chinese Gigaword, BR87) were included in the this evaluation. Three additional literary texts were added, Mark Twain's *The Adventures of Huckleberry Finn*, Friedrich Nietzsche's *Also Sprach Zarathustra* in the original German, Julius Caesar's *Comentarii de Bello Gallico* in the original Latin, as well as an additional corpus from CHILDES,

Bloom73 [Bloom, 1973], and a collection of stories for young children by William S. Gray. For one of these corpora, *Also Sprach Zarathustra*, MDL chose the best possible segmentation. Overall, MDL was always able to recover more than 90% of the quality of the best segmentation.

8 Discussion

The results presented in Section 5 show that BOOTSTRAP VOTING EXPERTS improves performance over VOTING EXPERTS in several word segmentation tasks. The results for Orwell's *1984* indicate that BOOTSTRAP VOTING EXPERTS can outperform other algorithms in the VOTING EXPERTS family. Performance on BR87 is particularly significant because a number of recent algorithms have been evaluated against this very same corpus, and BOOTSTRAP VOTING EXPERTS outperforms all of them, by a substantial margin. The results for the Chinese Gigaword corpus, while yielding the smallest gain from bootstrapping, are interesting given the current interest in Chinese word segmentation. Comparison with results other algorithms have obtained for Chinese corpora is difficult because few have been evaluated using the same metric (boundary F-measure or word F-measure). Tanaka-Ishii and Jin [2006] have done so, and achieved scores of over 0.8, but their evaluation used a *phonemic* encoding, which presents a very different segmentation problem (one that is actually more similar to English).

Though not designed with the incremental paradigm in mind, the results of Section 6 show that BOOTSTRAP VOTING EXPERTS can be used effectively within it, besting at least one prominent segmentation algorithm designed for it. With a large enough corpus, both VOTING EXPERTS algorithms converge to a level of performance that is similar to their respective performance in the usual batch scenario. Contrary to prior published results, we suggest that performance of VOTING EXPERTS and MBDP-1 will generally be competitive in this paradigm (though whenVOTING EXPERTS operates in batch mode, it will generally be the higher performer). It is important to note that BOOTSTRAP VOTING EXPERTS is not without drawbacks, the two most obvious being the requirement of an additional parameter, and the extra computational expenses incurred by segmenting the corpus multiple times and maintaining an additional trie.

As an unsupervised algorithm, the sensitivity of VOTING EXPERTS to parameter settings results in a dilemma: Either fix a set of universal parameters and tolerate compromised performance on many tasks, or hand-set parameters for each domain. Not content with these choices, we examined a third possibility: Estimate the parameters in an unsupervised manner. The method of parameter estimation investigated here, a form of MDL, provides a good first step, and generally results in segmentation quality that is close to the best possible for VOTING EXPERTS.

References

[Argamon *et al.*, 2004] Shlomo Argamon, Navot Akiva, Amihood Amir, and Oren Kapah. Efficient unsupervised word segmentation using minimum description length. In *Coling 2004*, 2004.

[Bloom, 1973] Lois Bloom. *One Word at a Time*. Mouton, Paris, 1973.

[Brent and Tao, 2001] Michael Brent and Xiaopeng Tao. Chinese text segmentation with MBDP-1: making the most of training corpora. In *ACL 2001*, 2001.

[Brent, 1999] Michael Brent. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105, 1999.

[Cheng and Mitzenmacher, 2005] Jimming Cheng and Michael Mitzenmacher. The markov expert for finding episodes in time series. In *Proceedings of the Data Compression Conference*, 2005.

[Cohen and Adams, 2001] Paul R. Cohen and Niall Adams. An algorithm for segmenting categorical time series into meaningful episodes. In *Proceedings of the Fourth Symposium on Intelligent Data Analysis*, 2001.

[Cohen *et al.*, 2006] Paul Cohen, Niall Adams, and Brent Heeringa. Voting experts: An unsupervised algorithm for segmenting sequences. *Journal of Intelligent Data Analysis*, 2006.

[Fleck, 2008] Margaret M. Fleck. Lexicalized phonotactic word segmentation. In *ACL 2008 Proceedings*, 2008.

[Goldwater *et al.*, 2008] Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. A bayesian framework for word segmentation. Submitted, 2008.

[Huang, 2007] Chu-Ren Huang. Tagged chinese gigaword. Catalog LDC2007T03, Linguistic Data Consortium, Philadelphia, 2007.

[MacWhinney and Snow, 1985] Brian MacWhinney and Catherine Snow. The child language data exchange system. *Journal of Child Language*, 12:271–192, 1985.

[Miller and Stoytchev, 2008] Matthew Miller and Alexander Stoytchev. Hierarchical voting experts: An unsupervised algorithm for hierarchical sequence segmentation. In *Proceedings of the 7th IEEE International Conference on Development and Learning (ICDL)*, 2008.

[Nevill-Manning and Witten, 1997] Craig G. Nevill-Manning and Ian H. Witten. Identifying hierarchical structure in sequences: A linear-time algorithm. *Journal of Artificial Intelligence Research*, 7:67–82, 1997.

[Ratner, 1987] Nan Bernstein Ratner. The phonology of parent child speech. In K. Nelson and A. van Kleek, editors, *Children's Language*, volume 6. Erlbaum, Hillsdale, NJ, 1987.

[Tanaka-Ishii and Jin, 2006] Kumiko Tanaka-Ishii and Zhihui Jin. From phoneme to morpheme: Another verification using a corpus. In *ICCPOL 2006*, pages 234–244, 2006.

[Venkataraman, 2001] Anand Venkataraman. A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27:351–372, 2001.