An Unsupervised Algorithm for Finding Episode Boundaries

Tracking Number: 488

Paul Cohen
Department of Computer Science
University of Massachusetts, Amherst
Amherst, MA 01003
cohen@cs.umass.edu

Brent Heeringa
Department of Computer Science
University of Massachusetts, Amherst
Amherst, MA 01003
heeringa@cs.umass.edu

Niall Adams
Department of Mathematics
Imperial College
London, UK
n.adams@ic.ac.uk

ABSTRACT

This paper describes an unsupervised algorithm for segmenting categorical time series into episodes. The Voting-Experts algorithm first collects statistics about the frequency and boundary entropy of ngrams, then passes a window over the series and has two "expert methods" decide where in the window boundaries should be drawn. The algorithm segments text into words successfully in four languages. We claim that Voting-Experts finds meaningful episodes in categorical time series because it exploits two statistical characteristics of meaningful episodes.

1. INTRODUCTION

Though we live in a continuous world, we have the impression that experience comprises episodes: writing a paragraph, having lunch, going for a walk, and so on. Episodes have hierarchical structure; for instance, writing a paragraph involves thinking of what to say, saying it, editing it; and these are themselves episodes. Do these examples of episodes have anything in common? Is there a domain-independent, formal notion of episode sufficient, say, for a robot to segment continuous experience into episodes? One can distinguish three ways to identify episode boundaries: First, they may be *marked*, as spaces mark word boundaries and promoters mark coding regions in DNA. Second, episodes may be *recognized*. For instance, we recognize nine words in the sequence "itwasabrightcolddayinapriland". Third we might *infer* episode boundaries given the statistical structure of a series. For example, "juxbtbcsjhiudpmeebzjobqsjmboe" is formally (statistically) identical with "itwasabrightcolddayinapriland" — one is obtained from the other by replacing each letter with the adjacent one in the alphabet — however, the latter is easily segmented by recognition whereas the former requires inference.

This paper presents a domain-independent, formal theory of episodes and an algorithm for inferring episode boundaries. In our theory of the development of robot minds [1], the ability to segment experiences into episodes is both more general and developmentally prior to a large memory of episode types. In previous papers we demonstrated how a robot could cluster episodes to find types (e.g., [7, 9, 8]) after *we* marked episode boundaries by hand. This paper provides evidence that a robot can find episode boundaries for itself, without supervision, given *only* the sequence to be segmented.

The generality of our method is perhaps its most important feature, because a general algorithm can bootstrap a memory of specific episode types afforded by and appropriate to the environment. As evidence of generality, we show how one algorithm can divide sequences of letters and ideograms into words (in several languages) and sequences of controller data into robotic episodes. As the former case provides more accessible examples, we use it to illustrate our theory and algorithm in the following sections.

2. THE EPISODE BOUNDARY PROBLEM

Suppose we remove all the spaces and punctuation from a text, can an algorithm figure out where the word boundaries should go? Here is the result of running Voting-Experts on the first 500 characters of George Orwell's *1984*. The ⋆ symbols are induced boundaries:

Itwas ⋆ a ⋆ bright ⋆ cold ⋆ day ⋆ in ⋆ April ⋆ andthe ⋆ clockswere ⋆ st ⋆ ri ⋆ king ⋆ thi ⋆ rteen ⋆ Winston ⋆ Smith ⋆ his ⋆ chin ⋆ nuzzl ⋆ edinto ⋆ his ⋆ brea ⋆ st ⋆ in ⋆ aneffort ⋆ to ⋆ escape ⋆ the ⋆ vilewind ⋆ slipped ⋆ quickly ⋆ through ⋆ the ⋆ glass ⋆ door ⋆ sof ⋆ Victory ⋆ Mansions ⋆ though ⋆ not ⋆ quickly ⋆ en ⋆ ought ⋆ oprevent ⋆ aswirl ⋆ ofgrit ⋆ tydust ⋆ from ⋆ ent ⋆ er ⋆ inga ⋆ long ⋆ with ⋆ himThe ⋆ hall ⋆ ways ⋆ meltof ⋆ boiled ⋆ cabbage ⋆ and ⋆ old ⋆ ragmatsA ⋆ tone ⋆ endof ⋆ it ⋆ acoloured ⋆ poster ⋆ too ⋆ large ⋆ for ⋆ indoor ⋆ dis ⋆ play ⋆ hadbeen ⋆ tack ⋆ ed ⋆ tothe ⋆ wall ⋆ It ⋆ depicted ⋆ simplya ⋆ n ⋆ enormous ⋆ face ⋆ more ⋆ than ⋆ ametre ⋆ widethe ⋆ faceof ⋆ aman ⋆ of ⋆ about ⋆ fortyfive ⋆ witha ⋆ heavy ⋆ black ⋆ moustache ⋆ and ⋆ rugged ⋆ ly ⋆ handsome ⋆ featur

The segmentation is imperfect: Words are run together (Itwas, aneffort) and broken apart (st ⋆ ri ⋆ king). Very occasionally, words are split between segments ("to" in en ⋆ ought ⋆ oprevent). Still, the segmentation is surprisingly good when one considers that it is based on nothing more than statistical features of subsequences of letters — not words, as no word boundaries are available — in Orwell's text.

How can an algorithm identify subsequences that are *meaningful* in a domain — words and robot activities — lacking any knowledge about the domain; and particularly, lacking positive and negative training instances of meaningful subsequences? VOTING-EXPERTS must somehow detect *domain-independent* indicators of the boundaries of meaningful subsequences. In fact, this is a good description of what it does. It implements a weak theory of domain-independent features of meaningful units. The first of these features is that entropy remains low inside meaningful units and increases at their boundaries; the second is that high-frequency subsequences are more apt to be meaningful than low-frequency ones.

3. CHARACTERISTICS OF EPISODES

The features of episodes that we have implemented in the Voting Experts algorithm are called *boundary entropy* and *frequency*:

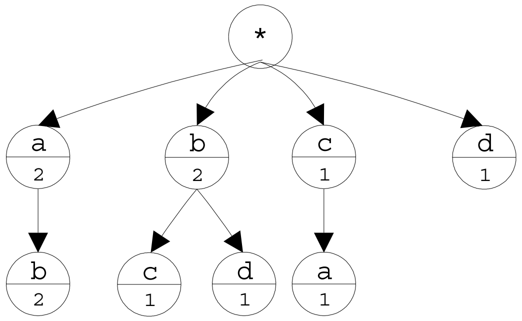
Boundary entropy. Every unique subsequence is characterized by the distribution of subsequences that follow it; for example, the subsequence "en" in this sentence repeats seven times and is followed by tokens c (4 times), t, s and ", a distribution of symbols with an entropy value (1.66, as it happens). In general, every subsequence S has a boundary entropy, which is the entropy of the distribution of subsequences of length m that follow it. If S is an episode, then the boundary entropies of subsequences of S will have an interesting profile: They will start relatively high, then sometimes drop, then peak at the last element of S. The reasons for this are, first, that the predictability of elements within an episode increases as the episode extends over time; and, second, that the element that immediately follows an episode is relatively uncertain. Said differently, within episodes, we know roughly what will happen, but at episode boundaries we become uncertain.

Frequency. Episodes, recall, are meaningful sequences. They are patterns in a domain that we call out as special, important, valuable, worth committing to memory, worth naming, etc. One reason to consider a pattern meaningful is that one can use it for something, like prediction. (Predictiveness is another characteristic of episodes nicely summarized by entropy.) Rare patterns are less useful than common ones simply because they arise infrequently, so all human and animal learning places a premium on frequency. In general, episodes are common patterns, but not all common patterns are episodes, as we saw earlier.

4. THE VOTING EXPERTS ALGORITHM

VOTING-EXPERTS includes experts that attend to boundary entropy and frequency, and is easily extensible to include experts that attend to other characteristics of episodes. The algorithm simply moves a window across a time series and asks, for each location in the window, whether to "cut" the series at that location. Each expert casts a vote. Each location takes n steps to traverse a window of size n, and is seen by the experts in n different contexts, and may accrue up to n votes from each expert. Given the results of voting, it is a simple matter to cut the series at locations with high vote counts. Here are the steps of the algorithm:

Build an ngram tree of depth $n+1$. Nodes at level i of an ngram tree represent ngrams of length i. The children of a node are the extensions of the ngram represented by the node. For example, $a\ b\ c\ a\ b\ d$ produces the following ngram tree of depth 2:



Every ngram of length 2 or less in the sequence $a\ b\ c\ a\ b\ d$ is represented by a node in this tree. The numbers in the lower half of the nodes represent the frequencies of the subsequences. For example, the subsequence ab occurs twice, and every occurrence of a is followed by b.

For the first 10,000 characters in Orwell's text, an ngram tree of depth 7 includes 33774 nodes, of which 9109 are leaf nodes. That is, there are over nine thousand unique subsequences of length 7 in this sample of text, although the average frequency of these subsequences is 1.1—most occur exactly once. The average frequencies of subsequences of length 1 to 7 are 384.4, 23.1, 3.9, 1.8, 1.3, 1.2, and 1.1.

Calculate boundary entropy. The boundary entropy of an ngram is the entropy of the distribution of tokens that can extend the ngram. The entropy of a distribution for a discrete random variable X is

$$-\sum_{x \in X} p(x) \log p(x)$$

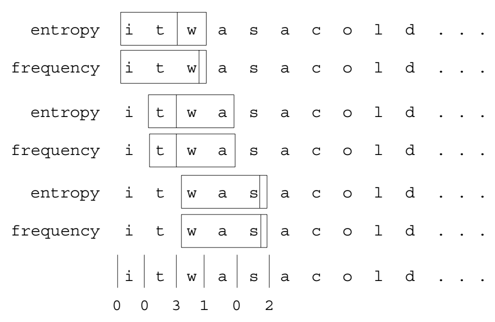
Boundary entropy is easily calculated from the ngram tree. For example, the node a in the tree above has entropy equal to zero because it has only one child, ab, whereas the entropy of node b is 1.0 because it has two equiprobable children, bc and bd. Clearly, only the first n levels of the ngram tree of depth $n+1$ can have node entropy scores.

Standardize frequencies and boundary entropies. In most domains, there is a systematic relationship between the length and frequency of patterns; in general, short patterns are more common than long ones (e.g., on average, for subsets of 10,000 characters from Orwell's text, 64 of the 100 most frequent patterns are of length 2; 23 are of length

3, and so on). Our algorithm will compare the frequencies and boundary entropies of ngrams of different lengths, but in all cases we will be comparing how *unusual* these frequencies and entropies are, relative to other ngrams of the same length. To illustrate, consider the words "a" and "an". In the first 10000 characters of Orwell's text, "a" occurs 743 times, "an" 124 times, but "a" occurs only a little more frequently than other one-letter ngrams, whereas "an" occurs much more often than other two-letter ngrams. In this sense, "a" is ordinary, "an" is unusual. Although "a" is much more common than "an" it is much less unusual relative to other ngrams of the same length. To capture this notion, we standardize the frequencies and boundary entropies of the ngrams. (To standardize a value in a sample, subtract the sample mean from the value and divide by the sample standard deviation. This has the effect of expressing the value as the number of standard deviations it is away from the sample mean.) Standardized, the frequency of "a" is 1.1, whereas the frequency of "an" is 20.4. In other words, the frequency of "an" is 20.4 standard deviations above the mean frequency for sequences of the same length. We standardize boundary entropies in the same way, and for the same reason.

Score potential segment boundaries. In a sequence of length k there are $k-1$ places to draw boundaries between segments, and, thus, there are 2^{k-1} ways to divide the sequence into segments. Our algorithm is greedy in the sense that it considers just $k-1$, not 2^{k-1}, ways to divide the sequence. It considers each possible boundary in order, starting at the beginning of the sequence. The algorithm passes a window of length n over the sequence, halting at each possible boundary. All of the locations within the window are considered, and each garners zero or one vote from each expert. Because we have two experts, for boundary-entropy and frequency, respectively, each possible boundary may accrue a maximum of $2n$ votes. This is illustrated below. A window of length 3 is passed along the sequence `itwasacold`.

Initially, the window covers `itw`. The entropy and frequency experts each decide where they could best insert a boundary within the window (more on this, below). The entropy expert favors the boundary between `t` and `w`, while the frequency expert favors the boundary between `w` and whatever comes next. Then the window moves one location to the right and the process repeats. This time, both experts decide to place the boundary between `t` and `w`. The window moves again and both experts decide to place the boundary after `s`, the last token in the window. Note that each potential boundary location (e.g., between `t` and `w`) is seen n times for a window of size n, but it is considered in a slightly different context each time the window moves. The first time the experts consider the boundary between `w` and `a`, they are looking at the window `itw`, and the last time, they are looking at `was`.



entropy i t | w a s a c o l d . . .
frequency i t w | a s a c o l d . . .
entropy i t | w a s a c o l d . . .
frequency i t | w a s a c o l d . . .
entropy i t w a s | a c o l d . . .
frequency i t w a s | a c o l d . . .
| i | t | w | a | s | a c o l d . . .
0 0 3 1 0 2

In this way, each boundary gets up to $2n$ votes, or $n = 3$ votes from each of two experts. The `wa` boundary gets one vote, the `tw` boundary, three votes, and the `sa` boundary, two votes.

The experts use slightly different methods to evaluate boundaries and assign votes. Consider the window `itw` from the viewpoint of the boundary entropy expert. Each location in the window bounds an ngram to the left of the location; the ngrams are `i`, `it`, and `itw`, respectively. Each ngram has a standardized boundary entropy. The boundary entropy expert votes for the location that produces the ngram with the highest standardized boundary entropy. As it happens, for the ngram tree produced from Orwell's text, the standardized boundary entropies for `i`, `it`, and `itw` are 0.2, 1.39 and 0.02, so the boundary entropy expert opts to put a boundary after the ngram `it`.

The frequency expert places a boundary so as to maximize the sum of the standardized frequencies of the ngrams to the left and the right of the boundary. Consider the window `itw` again. If the boundary is placed after `i`, then (for Orwell's text) the standardized frequencies of `i` and `tw` sum to 1.73; if the boundary is placed after `it`, then the standardized frequencies of `it` and `w` sum to 2.9; finally, if it is placed after `itw`, the algorithm has only the standardized frequency of `itw` to work with; it is 4.0. Thus, the frequency expert opts to put a boundary after `itw`.

Segment the sequence. Each potential boundary in a sequence accrues votes, as described above, and now we must evaluate the boundaries in terms of the votes and decide where to segment the sequence. Our method is a familiar "zero crossing" rule: If a potential boundary has a locally maximum number of votes, split the sequence at that boundary. In the example above, this rule causes the sequence `itwasacold` to be split after `it` and `was`. We confess to one embellishment on the rule: The number of votes for a boundary must exceed an absolute threshold, as well as be a local maximum. We found that the algorithm splits too often without this qualification.

Let us review the design of the experts and the segmentation rule, to see how they test the characteristics of episodes described earlier. The boundary entropy expert assigns votes to locations where the boundary entropy peaks, locally, implementing the idea that entropy increases at episode boundaries. The frequency expert tries to find a "maximum likelihood tiling" of the sequence, a placement of boundaries that makes the ngrams to the left and right of the boundary as likely as possible. When both experts vote for a boundary, and especially when they vote repeatedly for the same boundary, it is likely to get a locally-maximum number of votes, and the algorithm is apt to split the sequence at that location.

5. EVALUATION

In these experiments, induced boundaries stand in six relationships to episodes.

1. The boundaries coincide with the beginning and end of the episode;
2. The episode falls entirely within the boundaries and begins or ends at one boundary.
3. The episode falls entirely within the boundaries but neither the beginning nor the end of the episode correspond to a boundary.
4. One or more boundaries splits an episode, but the beginning and end of the episode coincide with boundaries.
5. Like case 4, in that boundaries split an episode, but only one end of the episode coincides with a boundary.
6. The episode is split by one or more boundaries and neither end of the episode coincides with a boundary.

These relationships are illustrated graphically in Figure 1, following the convention that horizontal lines denote actual episode, and vertical lines denote induced boundaries. The cases can be divided into three groups. In cases 1 and 4, boundaries correspond to both ends of the episode; in cases 2 and 5, they correspond to one end of the episode; and in cases 3 and 6, they correspond to neither end. We call these cases *exact*, *dangling*, and *lost* to evoke the idea of episodes located exactly, dangling from a single boundary, or lost in the region between boundaries.

We use both hit and false-positive rates to measure the accuracy of our episode finding algorithms. To better explain the trade-offs between hits and false-positives we employ the

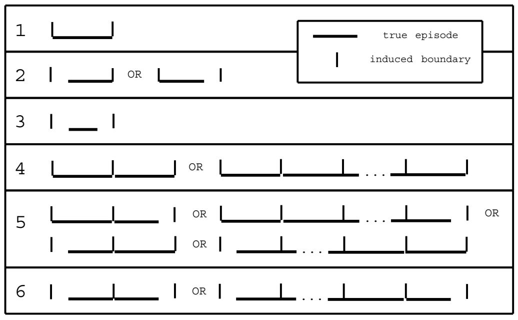


Figure 1: A graphical depiction of the relationships between boundaries and episodes. Horizontal lines denote true episodes; their ends the correct boundaries. Vertical lines denote induced episode boundaries.

F-measure. This standard comparison metric is formally defined as

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

where Recall is the hit-rate and Precision is the ratio of correct hits to proposed hits. Note that difference in proposed and correct hits yields the number of false positives. Higher F-measures indicate better performance.

In many of these experiments, we compare the results of VOTING-EXPERTS with another unsupervised algorithm, SEQUITUR, that also claims to find structure in categorical time series.

SEQUITUR is a compression-based algorithm that builds a context-free grammar from a string of discrete tokens [6, 5]. It has successfully identified structure in both text and music. The rules used to build the grammar are easily expressed by two constraints.

1. No pair of adjacent symbols appears more than once in the grammar.
2. Every production is used more than once.

SEQUITUR works by appending a new token to the right-hand-side of the sentential production and then iteratively applying the above constraints. The algorithm runs in linear time in the size input and is often used to infer hierarchical structure in univariate timeseries. This structure is denoted by the rules of the induced grammar. Expanding the productions reveals boundary information. For example giving SEQUITUR the sequence of tokens

s e e s p o t s p i t s e e s p o t s i t

results in the grammar

$$\begin{array}{lll} S & \rightarrow & A\ B\ C\ A\ s\ C \\ A & \rightarrow & s\ e\ e\ B\ o\ t \\ B & \rightarrow & s\ p \\ C & \rightarrow & i\ t \end{array}$$

If we view each non-terminal on the right-hand-side of the sentential production S as an episode, then the following boundaries are induced:

s e e s p o t ⋆ s p ⋆ i t ⋆ s e e s p o t ⋆ s ⋆ i t

We refer to these as the level 1 episode boundaries because, by definition, they mark longer, macro-level boundaries. By expanding each non-terminal in S one level and applying the same criteria for episodes results in the level 2 boundaries:

s e e ⋆ s p ⋆ o t ⋆ s p ⋆ i t ⋆ s e e ⋆ s p ⋆ o t ⋆ s ⋆ i t

Note that level 2 boundaries contain the level 1 boundaries and that in general, level n boundaries contain level $n-1$ boundaries.

5.1 Word Boundaries

We removed spaces and punctuation from text and assessed how well Voting-Experts and Sequitur could induce word boundaries on four languages.

5.1.1 *English*

We ran Voting-Experts and Sequitur on the first 50,000 characters of Orwell's *1984*. The detailed results are given in Table 1. Voting-Experts performed best when the window length was 7 and the threshold was 4. The algorithm induced 12153 boundaries, for a mean episode length of 4.11. The mean word length in the text was 4.49. The algorithm induced boundaries at 80% of the true word boundaries (the hit rate) missing 20% of the word boundaries. 27% of the induced boundaries did not correspond to word boundaries (the false positive rate). Exact cases, described above, constitute 62.6% of all cases; that is, 62.6% of the words were bounded at both ends by induced boundaries. Dangling and lost cases constitute 33.9% and 3.3% of all cases, respectively. Said differently, only 3.3% of all words in the text got lost between episode boundaries. These tend to be short words, in fact, 59% of the lost words have length 3 or shorter and 85% have length 5 or shorter. In contrast, all 89% the words for which the algorithm found exact boundaries are of length 3 or longer.

Sequitur performed best when expanding only to the level 1 boundaries. That is, it achieved its highest F-measure by not further expanding any non-terminals off the sentential production. Expanding to further levels leads to a substantial increase in the false positive rate and hence the overall decrease in F-measure. Sequitur, at level 5, identified 78% of the word boundaries correctly, 20% dangling and only 2% missed. This happens because Sequitur is inducing more boundaries. However, the increase in proposed boundaries results in a non-proportional increase of the ratio between hits and false-positives. This is clear by the decreasing F-measure. In fact, at level 5, the false-positive rate of 68% is near the 78% maximum false positive rate achieved by inducing a boundary at every location. The same behavior occurs to a smaller extent in Voting-Experts when the splitting threshold is decreased. For example, with a window length of 4 and a threshold of 2, Voting-Experts finds 74% of the word boundaries exactly. Still, the F-measure for this example is higher than the best achieved by Sequitur. In general, Sequitur found likely patterns, but these patterns did not always correspond to word boundaries.

It is easy to ensure that all word boundaries are found, and no word is lost: Induce a boundary between each letter. However, this strategy would induce a mean episode length of 1.0, much shorter than the mean word length. The false-positive count would equal the total number of non-boundaries in the text and the false-positive rate would converge to the ratio of non-boundaries over total locations. In contrast, Voting-Experts finds roughly the same number of episodes as there are words in the text, and loses very few words between boundaries. This success is evident in the high F-measure (.76) achieved by Voting-Experts.

Further evidence of Voting-Experts ability to find meaningful word boundaries is given in Figures 2 and 3. In Figure 2 we graph the percentage of exact word matches as a function of word length. For example, Sequitur exactly matches 30% of words having length 15 while Voting-Experts matches 70%. The curves converge at word length 17 because only two words in our corpus have length 17 and both algorithms find only one of them. The curves roughly mimic each other except in the word length interval from 2 to 4. In this period, Voting-Experts accelerates over Sequitur because it finds disproportionately more exact matches than Sequitur. This phenomenon is even easier to see in Figure 3. Here cumulative percentage of exact word matches is plotted as a function of word lengths and the distribution of word lengths is given behind the curves. The slope of Voting-Experts is steeper than Sequitur in the interval from 2 to 4 revealing the success it has on the most frequent word lengths. Furthermore, words with length 2, 3, and 4 comprise over 57% of the Orwell corpus, so at places where accuracy is perhaps most important, Voting-Experts performs well.

The effects of window length and splitting threshold on F-measure are shown in Figure 4. The two horizontal axes represent window length and splitting threshold while the vertical axis represents F-measure. We see window length has little effect on the success of Voting-Experts as it extends past length 6. This is because long subsequences are uncommon, so if we build an ngram tree of depth 30, then the standardized frequencies and boundary entropies at nodes deeper in the tree become miniscule.

The appropriate control conditions for this experiment were run and yielded the expected results: Voting-Experts performs marginally less well when it is required to segment text it has not seen. For example, if the first 10,000 characters of Orwell's text are used to build the ngram tree, and then the algorithm is required to segment the next 10,000 characters, there is a very slight decrement in performance. The algorithm performs very poorly given texts of random words, that is, subsequences of random letters.

5.1.2 *Chinese, German and Roma-ji*

As a test of the generality of Voting-Experts, we ran it on corpora of Roma-ji and Chinese texts as well as a segment of Franz Kafka's *The Castle* in the original German. Roma-ji is a transliteration of Japanese into roman characters. The corpus was a set of Anime lyrics comprising 19163 characters. The Chinese text comes from Guo Jim's Mandarin Chinese PH corpus. The PH corpus is taken from stories in newspaper texts and is coded in the standard GB-scheme. In GB code each Chinese character is a pair of ASCII values. We mapped each unique pair of ASCII characters to a unique token. For comparison purposes we selected the first 19163 characters of Kafka's text and the same number of characters from *1984* and the PH corpus. As always, we stripped away spaces and punctuation, and the algorithm induced word boundaries. The window length was 6. The

Orwell	F-measure	Hit Rate	False Positive Rate	Exact Rate	Dangling Rate	Lost Rate
SEQUITUR						
Level 1	.58	.58	.43	.30	.56	.14
Level 2	.55	.78	.58	.61	.35	.04
Level 3	.50	.86	.65	.74	.24	.02
Level 4	.48	.88	.67	.77	.21	.02
Level 5	.47	.88	.68	.78	.20	.02
VOTING-EXPERTS						
Depth 7, Threshold 4	.76	.80	.27	.63	.34	.03
Depth 10, Threshold 5	.75	.70	.18	.47	.45	.08
Depth 10, Threshold 8	.65	.51	.09	.23	.55	.22
Depth 4, Threshold 2	.64	.86	.48	.74	.24	.02

Table 1: Results of running SEQUITUR and VOTING-EXPERTS on George Orwell's *1984*.

VOTING-EXPERTS	F-measure	Hit Rate	False Positive Rate	Exact Rate	Dangling Rate	Lost Rate
German	.75	.79	.31	.61	.25	.04
English	.71	.76	.33	.58	.38	.04
Roma-ji	.65	.64	.34	.37	.53	.10
Chinese	.57	.42	.07	.13	.57	.30

Table 2: Results of running VOTING-EXPERTS on Franz Kafka's *The Castle*, Orwell's 1984, a subset of the Chinese PH corpus of newspaper stories, and a set of Roma-ji Anime lyrics.

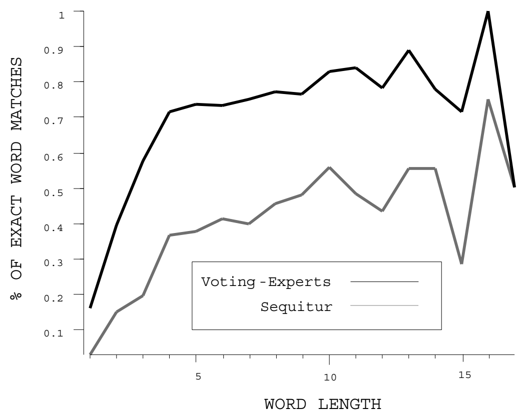


Figure 2: A comparison of exact match-rate on a per-word basis between SEQUITUR and VOTING-EXPERTS.

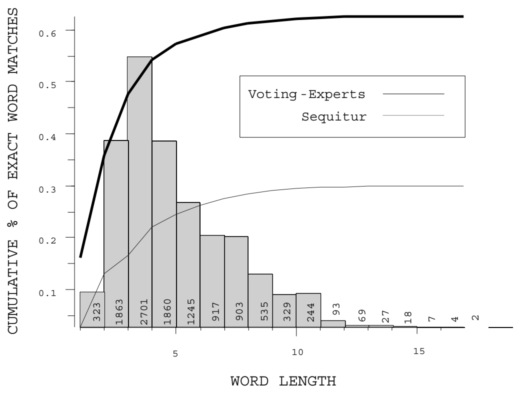


Figure 3: A comparison of cumulative exact match-rate over word length for SEQUITUR and VOTING-EXPERTS. The background histogram depicts the distribution of word lengths in the Orwell corpus.

results are given in Table 2.

Clearly the algorithm is not biased to do well on English. In particular, it performs very well on Kafka's text, losing only 4% of the words and identifying 61% exactly. The algorithm performs less well with the Roma-ji text; it identifies fewer boundaries accurately (i.e., places 34% of its boundaries within words) and identifies fewer words exactly. VOTING-EXPERTS performed worst on Chinese corpus. Only 42% of the boundaries were identified although the false positive rate is an extremely low 7%. The explanation for these results has to do with the lengths of words in the corpora. We know that the algorithm loses disproportionately many short words. Words of length 2 make up 39% of the Chinese corpus, 32% of the Roma-ji corpus, 17% of the Orwell corpus, and 10% of the Kafka corpus, so it is not surprising that the algorithm performs worst on the Chinese corpus and best on the Kafka corpus.

If we incorporate the knowledge that Chinese words are rather short in length by decreasing the splitting threshold, we can increase the F-measure of VOTING-EXPERTS to 77% on the PH corpus. In general, knowledge of the mean episode length can help improve the boundary detection of VOTING-EXPERTS.

5.2 Robot Episodes

We ran VOTING-EXPERTS and SEQUITUR on a multivariate timeseries of robot controller data comprising 17788 time steps and 65 unique states. Each state was mapped to a unique identifier, and these tokens were given to the algorithm as input. The timeseries data was collected with a Pioneer 2 mobile robot, equipped with sonar and a Sony pan-tilt-zoom camera. The robot wandered around a room-

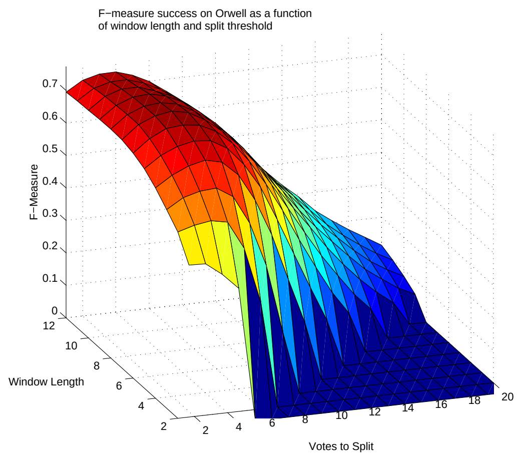


Figure 4: F-measure as a function of window length and the votes-to-split threshold. Note that window lengths of k can only have thresholds up to $2k$.

size playpen for 30 minutes looking for interesting objects. Upon finding an object, the robot orbited around it for a few minutes. The multivariate timeseries consisted of eight binary variables representing different controllers in our agent architecture. Each variable is 1 when its corresponding controller is active and 0 when its inactive.

- MOVE-FORWARD
- TURN
- COLLISION-AVOIDANCE
- VIEW-INTERESTING-OBJECT
- RELOCATE-INTERSTING-OBJECT
- SEEK-INTERESTING-OBJECT
- CENTER-CHASIS-ON-OBJECT
- CENTER-CAMERA-ON-OBJECT

This timeseries can be broken up into five different observable robot behaviors. Each behavior represents a qualitatively different episode in the timeseries. We denote these episodes as

- FLEEING
- WANDERING
- AVOIDING
- ORBITING-OBJECT
- APPROACHING-OBJECT

Table 3 summarizes the results of running VOTING-EXPERTS and SEQUITUR on the robot controller data. The definition of hit-rate and false-positive rate is slightly different here. Because the controller data can be noisy at the episode boundaries, we allow *hits* a window of length 1 in either temporal direction. For example, if we induce a boundary at location 10, but the actual boundary is at location 9, we still count it as a hit. We also enforce a rule that actual boundaries can only count once toward induced boundaries. For example, if we induce a boundary at 8 and count it as a hit toward the actual boundary 9, the induced boundary at 10 can no longer count toward 9.

The mean episode length in the robot controller data is 7.13. This length is somewhat smaller than expected because the robot often gets caught up in the corners of its playpen for periods of time and performs a series of wandering, avoiding, and fleeing behaviors to escape. The total number of true episodes was 2491. VOTING-EXPERTS induced 3038 episodes with a hit rate of 66% and a false-positive rate of 46% for a combined F-measure of 59%. Like on Orwell, VOTING-EXPERTS consistently outperforms SEQUITUR on the F-measure. SEQUITUR does best when expanding to the level 1 boundaries. The transition from level 1 to level 2 produces a sharp increase in the false-positive rate with a corresponding increase in hit rate, however the F-measure decreases slightly. At level 5, SEQUITUR loses only 8% of the episodes but its false-positive rate is 78%, which is near the maximum possible rate of 86%.

6. RELATED WORK

Many methods have been developed for segmenting time series. Of these, many deal with continuous time series, and so are not directly applicable to the problem we are considering here. Some methods for categorical series are based on compression (e.g., [5, 6, 10]), but as we just saw with SEQUITUR, compression finds common, not necessarily meaningful, subsequences. Some methods are trained to find instances of patterns or templates (e.g., [4, 2]), but we wanted an unsupervised method. There is some work on segmentation in the natural language and information retrieval literature, for instance, techniques for segmenting Chinese, which has no word boundaries in its orthography. The method in [11] is similar to ours, though it requires supervised training on very large corpora. The parsing based on mutual information statistics approach in [3] is similar to our notion of boundary entropy. We know of no related research on characteristics of meaningful subsequences, that is, statistical markers of boundaries of meaning-carrying subsequences.

7. CONCLUSION

In order for an agent to generalize its experiences, it must divide them into meaningful units. The VOTING-EXPERTS algorithm uses statistical properties of categorical time series to segment them into episodes without supervision or prior training. Although the algorithm does not use explicit knowledge of words or robot behaviors, it detects episodes in these domains. The algorithm successfully segments texts into words in four languages. With less success, VOTING-EXPERTS segments robot controller data into activities. In the future we will examine how other, domain-independent experts can help improve performance. The idea that meaningful subsequences differ from meaningless ones in some formal characteristics—that syntactic criteria might help us identify semantic units—has practical as well as philosophical implications.

Robot Data	F-measure	Hit Rate	False Positive Rate	Exact Rate	Dangling Rate	Lost Rate
SEQUITUR						
Level 1	.55	.57	.47	.17	.37	.46
Level 2	.51	.77	.62	.34	.37	.29
Level 3	.32	.88	.71	.48	.33	.19
Level 4	.38	.94	.76	.56	.32	.12
Level 5	.36	.97	.78	.63	.29	.08
VOTING-EXPERTS						
Depth 7, Threshold 4	.59	.66	.46	.20	.39	.41
Depth 9, Threshold 6	.59	.60	.41	.18	.38	.44
Depth 5, Threshold 2	.56	.80	.56	.27	.42	.31

Table 3: Results of running SEQUITUR **and** VOTING-EXPERTS **on 30 minutes of robot controller data.**

8. ACKNOWLEDGMENTS

This research is supported by DARPA under contract numbers DARPA/USASMDCDASG60-99-C-0074 and DARPA/AFRLF30602-01-2-0580. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of DARPA or the U.S. Government.

We are grateful to Ms. Sara Nishi for collecting the corpus of Anime lyrics.

9. REFERENCES

[1] P. Cohen, T. Oates, N. Adams, and C. Beal. Robot baby 2001. In *Proceedings of the Twelve Conference on Algorithmic Learning Theory*. Springer, 2001. Forthcoming.

[2] M. N. Garofalakis, R. Rastogi, and K. Shim. SPIRIT: Sequential pattern mining with regular expression constraints. In *The VLDB Journal*, pages 223–234, 1999.

[3] D. Magerman and M. Marcus. Parsing a natural language using mutual information statistics. In *Proceedings, Eighth National Conference on Artificial Intelligence (AAAI 90)*, pages 984–989, 1990.

[4] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289, 1997.

[5] C. G. Nevill-Manning and I. H. Witten. Compression and explanation using hierarchical grammars. *The Computer Journal*, 40(2/3):103–116, 1997.

[6] C. G. Nevill-Manning and I. H. Witten. Identifying hierarchical structure in sequences: A linear-time algorithm. *Journal of Artificial Intelligence Research*, 7(1):67–82, 1997.

[7] T. Oates, M. Schmill, and P. Cohen. Identifying qualitatively different outcomes of actions: Gaining autonomy through learning. In *Proceedings Fourth International Conference on Autonomous Agents, pp 110-111*. ACM, 2000.

[8] M. Ramoni, P. Sebastiani, and P. Cohen. Multivariate clustering by dynamics. In *Proceedings of the Seventeenth National Conference on AI*, pages 633–638. AAAI Press/The MIT Press, 2000.

[9] P. Sebastiani, M. Ramoni, and P. Cohen. Sequence learning via bayesian clustering by dynamics. In R. Sun and C. L. Giles, editors, *Sequence Learning: Paradigms, Algorithms and Applications*. Springer-Verlag: LNAI 1828, 2001.

[10] W. J. Teahan, Y. Wen, R. J. McNab, and I. H. Witten. A compression-based algorithm for chinese word segmentation. *Computational Linguistics*, 26(3):375–393, 2000.

[11] G. M. Weiss and H. Hirsh. Learning to predict rare events in event sequences. In *Knowledge Discovery and Data Mining*, pages 359–363, 1998.