The Markov Expert for Finding Episodes in Time Series

Jimming Cheng
Harvard University
jvcheng@fas.harvard.edu

Michael Mitzenmacher
Harvard University
michaelm@eecs.harvard.edu

Abstract

We describe a domain-independent, unsupervised algorithm for refined segmentation of time series data into meaningful episodes, focusing on the problem of text segmentation. The VOTING EXPERTS algorithm of Cohen et al. [1] achieves results with fairly low rates of error. The MARKOV EXPERT is a new approach that improves the performance of VOTING EXPERTS by further refining those results with votes from an additional expert. The new expert applies a Markov-based segmentation method inspired by the approach of Teahan et al. [2], using the results of VOTING EXPERTS' frequency and entropy experts as a sample corpus from which to draw prefix/suffix frequency data. In contrast with the setting of Teahan et al., in this setting the sample corpus is small and somewhat inaccurate, but despite its errors, it is directly similar to the intended output in terms of non-space characters. The result is a high quality domain-independent segmentation algorithm that performs substantially better than VOTING EXPERTS.

1 Introduction

Finding episodes in time series is a central challenge to many problems, including text, audio, and video segmentation. We define a time series to be a sequential series of occurrences, and an episode is a grouping of occurrences within the time series that carries some sort of meaning. For example, in speech processing, input is given as a continuous, unmarked stream of phonemes in which certain groups of consecutive phonemes correspond to words. Finding these word boundaries can ease processing by subsequent components. Another potential application of extracting episodes in time series relates to robot learning. A robot that records its own actions in conjunction with feedback from the environment might want to know exactly which sequence of actions caused certain environmental responses. Correctly segmenting the exact sequence of actions ensures that the robot performs all necessary actions and does not perform unneeded actions the next time the associated response is desired.

Although episodes are present in many different kinds of time series data, they often have common statistical cues. Thus, it is possible to construct a domain-independent algorithm that performs well in many different settings without extra training. The VOTING EXPERTS algorithm of Cohen et al. [1] is one such approach that yields high-quality results. In this paper, we design an algorithm that improves the already good performance of VOTING EXPERTS by refining its results with a Markov-based inference method while maintaining domain independence. We introduce our algorithm, the MARKOV EXPERT, which captures many of the benefits of Markov-based prefix/suffix matching without requiring an extensive external corpus. The MARKOV EXPERT's performance is compared to VOTING EXPERTS using experiments and metrics similar to those of Cohen et al.

2 Text Segmentation

Text segmentation is a natural domain for episode discovery. It is simple to evaluate because a successful segmentation can be measured easily both quantitatively (hit rate, error rate, etc.) and qualitatively (how readable is the resulting text?). These metrics can be used to objectively compare our method with other techniques. We operate on a simplified alphabet consisting of only lowercase alphabetic letters and the space character. Unsegmented input is prepared by converting all letters to lowercase and removing all non-alphabetic characters, including spaces. The challenge is to re-insert the removed spaces in the correct locations. For a randomly selected passage from James Joyce's *Portrait of the Artist as a Young Man*, the unsegmented input is:

> ...perhapsawildrosemightbelikethosecoloursandherememberedthesongaboutthewildroseblossomsonthelittlegreenplacebutyoucouldnothaveagreenrosebutperhapssomewhereintheworldyoucouldthebellrangandthentheclassesbegantofileoutoftheroomsandalongthecorridorstowardstherefectoryhesatlookingatthetwoprintsofbutteronhisplatebutcouldnoteatthedampbreadthetableclothwasdampandlimpbuthedrankoffthehotweakteawhichtheclumsysculliongirtwithawhiteapronpouredintohiscuphewonderedwhetherthescullionsapronwasdamptooorwhetherallwhitethingswerecoldanddamp...

Correctly segmented, this would be:

> ...perhaps a wild rose might be like those colours and he remembered the song about the wild rose blossoms on the little green place but you could not have a green rose but perhaps somewhere in the world you could the bell rang and then the classes began to file out of the rooms and along the corridors towards the refectory he sat looking at the two prints of butter on his plate but could not eat the damp bread the tablecloth was damp and limp but he drank off the hot weak tea which the clumsy scullion girt with a white apron poured into his cup he wondered whether the scullions apron was damp too or whether all white things were cold and damp...

2.1 Previous Domain-dependent Solutions

Most previous solutions for text segmentation are domain-dependent. Dictionary based methods such as [3] incorporate knowledge of the language's vocabulary into the segmentation process. The resulting segmentation strives to maximize the number of dictionary words represented and discourages segmentations that do not agree with the dictionary.

Teahan et al.'s PPM compression-based segmentation approach models space insertion as a Markov process [2]. It operates by gathering statistics on a very large (1 million words) correctly segmented corpus. Segmentations are decided using character sequence probabilities and conditional character entropies collected from the corpus.

Both of the above approaches achieve extremely high rates of success. However, their dependence on large bodies of outside data such as dictionaries and corpuses make them difficult or impossible to apply to other domains. It is easy to find an English dictionary or a large body of English text, but suppose we want to find the melodic theme in a symphony or traffic patterns on a network. In such cases, a dictionary or corpus would be much harder to come by.

2.2 Previous Domain-independent Solutions

Successful domain-independent solutions for text segmentation are fewer in number and quality. Nevill-Manning et al.'s SEQUITUR algorithm [4] infers hierarchical structure in sequences, by constructing a hierarchical grammar from the input, with rules subsuming other rules at each level in the grammar tree. This method has proven to be useful in finding not only word

boundaries, but also prefix/suffix boundaries within words and sentence structure across words. Since SEQUITUR forms grammar rules by observing repetitions of sequences in the input, it is input-dependent only and does not rely on any other prior knowledge of the domain. Indeed, SEQUITUR has demonstrated proficiency in analyzing other languages such as French and German, as well as completely different domains such as musical scores. Unfortunately, Nevill-Manning et al. did not provide a detailed experimental assessment of SEQUITUR's effectiveness for the segmentation problem as they focused on other applications of SEQUITUR.

The VOTING EXPERTS algorithm [1] is designed specifically for domain-independent episode discovery. It does this by employing different "experts" to analyze the frequency and conditional entropies of sequences within the input. This statistical data is then used to make decisions about segmentation. Cohen et al. compared VOTING EXPERTS with their own implementation of SEQUITUR in the domain of text segmentation. VOTING EXPERTS exceeded SEQUITUR in all of the performance measures they supplied. Because of its high performance, and because its design allows the addition of more "experts" to contribute to segmentation decisions, VOTING EXPERTS is the starting point for our improved algorithm.

3 Voting Experts

VOTING EXPERTS works by gathering votes on decisions to segment at every location in the input time series from multiple "experts". Each expert is specialized with a strategy to make segmentation decisions. Each location accumulates votes from all of the employed experts, and experts may cast multiple votes for a single location, depending on how strongly the expert feels about segmenting at that location. When all votes have been cast, the input is segmented at locations where the number of total votes exceeds a user-defined threshold.

Cohen et al. define two specific experts in their study: the frequency expert and the entropy expert. The frequency expert takes advantage of the fact that recurring sequences are likely to be episodes. In English, the characters t-h-e are likely to occur many times in a large body of text. Thus, the frequency expert should favor segmenting before and after this sequence. However, t-h-e should not be segmented in certain cases, such as when it appears within the word "brother". Thus, the frequency expert should take into account the context of the occurrence. It accomplishes this by studying characters within a fixed-sized window that slides across the input. The expert considers segmenting at each location within the window, starting after the first character in the window and ending before the last character in the window. A frequency score is recorded at each location. Once scores are calculated for each location in the window, a vote is cast for the single location with the highest frequency score.

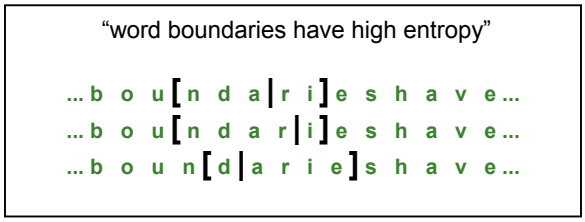

Figure 1: Sliding window of the frequency expert, looking for word boundaries.

The frequency score for a location is calculated by taking the sum of the frequency of the sequence preceding the proposed split and the frequency of the sequence following the split. For example, in the first line of Figure 1, a split is considered between "nda" and "ri" in the window.

The frequency score for this split is the number of times "nda" appears in the input, plus the number of times "ri" appears in the input.

Before the scores can be used, they must be normalized; we are interested in how unusual a count is compared to counts for other sequences of the same length. Following Cohen et al. [1], we standardize all scores to units expressing the *number of standard deviations from the mean score for sequences of that length.*

$$standardize(score(x)) = \frac{score(x) - \mu_{|x|}}{\sigma_{|x|}}$$

where $\mu_{|x|}$ = mean score among all sequences of length |x|
$\sigma_{|x|}$ = standard deviation among all sequences of length |x|

Cohen et al. also observed that conditional character entropy values can help decide word boundaries. In particular, character entropy peaks at the start of the word, just after the preceding word boundary. Within the word, entropy continually decreases. Intuitively, an upcoming character is less random within a word because that character is likely to be tied to the preceding characters. On the other hand, if the upcoming character occurs after a word boundary, its randomness is high because it is part of a different word than the preceding characters, and thus is free to assume more different values.

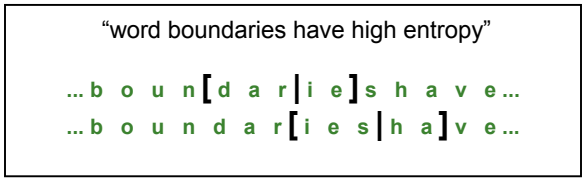

Figure 2: Sliding window for the entropy expert.

For example, the first line of Figure 2 considers the character entropy of "i" in the context of the preceding "dar". Since there are few English words ending in "dar", it is likely a within-word sequence. This means there are few options for the letter following it. "i" is one possibility ("boundaries"). "y" is another ("boundary"). Thus the conditional entropy of "i" is low.

In the second line of Figure 2, the context "ies" is a common English word ending. Since the next letter is the start of a new word, and that new word could be anything, the conditional entropy of the following letter "h" is high.

After entropy scores are calculated, they are standardized according to the same formula used for the frequency expert. Then, the entropy expert uses the same sliding windows method to iterate through the input and cast votes. In experiments on text [1], VOTING EXPERTS worked best when window sizes were 7 characters and the threshold was 4 votes.

Cohen et al. provided many experimental results in their study. We note that our experiments are based on our own implementation of VOTING EXPERTS, and the same code base is used as the foundation for the MARKOV EXPERTS extension to be discussed in the next section.

4 The Markov Expert

Our contribution, the MARKOV EXPERT, is an additional expert that participates with the entropy and frequency experts in the same process of voting for segmentation locations. However,

unlike the other experts it depends not only on the original input but also on preliminary results from the other experts. The MARKOV EXPERT waits for the entropy and frequency experts to vote first, and calculates a first-round segmentation based on those votes. The MARKOV EXPERT uses this first-round segmentation as a data corpus and analyzes suffix/prefix distributions within it. These gathered statistics are used to model space insertion as a Markov process. These voting locations are recombined with the original entropy and frequency expert votes to obtain a final segmentation. Thus, the MARKOV EXPERT's strategy is to use the partially correct results from the first two experts to generate further refinements.

Teahan et al.'s work segmenting Chinese text [2] was the original motivation for the MARKOV EXPERT's strategy. Their algorithm works on relatively short input, only a few sentences at a time. It requires a very large, correctly segmented training corpus for statistical reference. A dynamic programming approach efficiently considers every possible combination of space insertions in the input. Spaces are inserted in the locations that make the resulting segmented string smallest under PPM-compression with respect to the corpus. Thus, space insertion is modeled as a Markov process drawing from the supplied corpus. When executed on English text input with a 1 million word corpus, this algorithm yielded results with both a precision and a recall of 99.52%.

Clearly, the Markov modeling approach can return excellent results. We would like to design the MARKOV EXPERT so that it captures these benefits and contributes them in conjunction with the proposals by the frequency and entropy experts. Unfortunately, the MARKOV EXPERT cannot follow exactly the same strategy because it does not have the luxury of a large domain-specific corpus.

4.1 Adapting a Markov-base approach without an external corpus

We adapt the approach by having our MARKOV EXPERT compensate for lack of a large, correctly segmented corpus by instead using a smaller, imperfect one. An imperfect corpus can be obtained from a first-round segmentation of the input using the original VOTING EXPERTS. VOTING EXPERTS outputs segmentations with precisions around 77% and recall rates around 75%. These segmentations express a large percentage of correct space insertions, despite a smaller but considerable portion of false positives and missed segmentations. Also, these segmentations originate directly from the input. Hence the corpus and input contain the exact same sequence of non-space characters, and the resulting high level of corpus-input correlation should boost the effectiveness of a Markov-based analysis.

Teahan et al. demonstrated that the performance of their approach is correlated with the corpus size. Since the MARKOV EXPERT's input-derived corpus is imperfect and small (around 50,000 characters in our experiments), we do not expect to match Teahan et al.'s performance using their perfect, 5-7 million-character corpus. However, we will demonstrate that in settings for which external domain information is not available, this small and imprecise corpus is still useful and powerful enough to allow the MARKOV EXPERT to substantially improve the performance of VOTING EXPERTS.

4.2 Deciding Votes with Sliding Windows

In contrast with Teahan et al., the MARKOV EXPERT's objective is not to directly output a best-case segmentation. Rather, it seeks to offer its opinion for each individual space insertion and contribute to a collective segmentation decision with the other involved experts. This can be accomplished more efficiently with a sliding window method like that used by the frequency and entropy experts, rather than by using dynamic programming. The dynamic programming

approach gives an optimal global solution, but does not express local differences in confidence within that solution. If the MARKOV EXPERT used this approach, it would only cast either one or zero votes (yay or nay) for each space insertion decision. The sliding window method allows much richer expression. For each space insertion, the number of possible votes ranges from zero to the window size (at least 7 in most cases). This permits MARKOV EXPERT to impart finer grained influence on the final voting outcome.

Another consideration is that the sliding window method is less computationally expensive. The complexity is $O(nw)$, where n is the input length and w is the window size. The optimal method has complexity $O(n^2)$ when implemented using dynamic programming. This difference is significant because our inputs are large (50,000 characters) compared to Teahan et al. (1 or 2 sentences).

4.3 Segmentation Utility Function

As we consider a segmentation within a window, we are supposing that the sequence before the cut is the suffix of some word, and the sequence after the cut is the prefix of some other word. For example, in Figure 2, segmenting after the "s" implies that the preceding word should end in "-ies" and the next word should start with "ha-". The quality of a particular segmentation should be positively correlated to the probability that these suffixes and prefixes actually occur at word boundaries in the corpus, and negatively correlated to the probability that the current segmentation divides a word that appears intact in the corpus.

The frequencies of suffixes and prefixes can be easily determined using standard suffix and prefix trees. We denote these frequencies as:

$$fr_{\text{suffix}}(c,w) = frequency(\text{"}s_{c+1,w}\#\text{"})$$
$$fr_{\text{prefix}}(c,w) = frequency(\text{"}\#s_{1,c}\text{"})$$

In the above equations, $s_{i,j}$ denotes the substring starting with the i^{th} and ending with the j^{th} symbol in the current window. The position of the proposed cut is c. The symbol # represents a space; in more general contexts than text, it would represent an episode boundary.

To find the number of times that a particular cut through the current sequence would divide a word in the corpus, we consider every substring within the window that spans the cut. If c is the position of the proposed cut, then the number of times that the current cut conflicts with the corpus is given by:

$$fr_{\text{bad\_cut}}(c,w) = \sum_{i=2}^{c}\sum_{j=c+1}^{w-1} frequency(\text{"}\#s_{i,j}\#\text{"}) + \sum_{i=2}^{w} frequency(\text{"}\#s_{i,w}\text{"}) + \sum_{j=c+1}^{w-1} frequency(\text{"}s_{1,j}\#\text{"}) + frequency(\text{"}s_{1,w}\text{"})$$

This accounts for all words bounded or partially bounded within the current window, as well as words with boundaries that fall outside of the window. The original frequencies for the suffix and prefix are each discounted by $fr_{\text{bad\_cut}}$. That is, each occurrence of the sequence within a word cancels out an occurrence of the sequence at a word boundary. The discounted figures are then standardized according to the frequency means and standard deviations (μ_{suffix}, σ_{suffix}, μ_{prefix}, σ_{prefix}) for all sequences equal in length. The final utility function expressing the fitness of the cut c in window w is given by:

$$fitness(c,w) = \frac{fr_{suffix}(c,w) - fr_{bad\_cut}(c,w) - \mu_{suffix}}{\sigma_{suffix}} + \frac{fr_{prefix}(c,w) - fr_{bad\_cut}(c,w) - \mu_{prefix}}{\sigma_{prefix}}$$

4.4 Algorithm Parameters

Since it uses the output from frequency and entropy experts, the MARKOV EXPERT naturally shares their single optimal window size. However, experiments show that better results can be obtained by averaging the votes for a range of window sizes rather than just using a single value. This is also true for the frequency and entropy experts, but it is very costly for them to accommodate larger window sizes, because the number of sequences to investigate grows exponentially with the window size. The MARKOV EXPERT does not have this difficulty because the number of sequences to investigate is related only to the number of episodes inferred by the previous experts. The optimal window size was seven across all tested domains. The MARKOV EXPERT used the average value of four trials with window sizes of 6, 7, 8, and 9.

Another parameter to consider is the minimum suffix/prefix length. When analyzing the suffix and prefix resulting from a cut within a window, the MARKOV EXPERT should ignore any suffix or prefix shorter than this minimum value. For text inputs, imposing a minimum length of two characters gave the best results.

The final tally of votes will contain votes from the frequency expert, the entropy expert and the MARKOV EXPERT. Results improved when the contributions from each expert were weighted differently. Our experiments revealed that the in the optimal case, the MARKOV EXPERT's votes should be weighted twice as much as the frequency and entropy experts.

5 Results and Analysis

Our experiments focus on comparisons between our MARKOV EXPERT and the original VOTING EXPERTS, as Cohen et al. already showed that their approach was much more effective for this problem than other algorithms, including the SEQUITUR algorithm [4]. Our results show that the MARKOV EXPERT is consistently superior.

5.1 Experimental Domains

To demonstrate domain independence, we used samples from a diverse selection of human languages and from an experiment with a mobile robot:

- George Orwell's *1984* in English
- A collection of song lyrics in Romanized Japanese (Romaji)
- Johann Goethe's *Faust* in German
- Jules Verne's *20,000 Leagues Under the Sea* in French
- Cletto Arrighi's *Nana a Milano* in Italian
- Verner von Heidenstam's *Folkundatradet* in Swedish
- 30 minutes of controller data from a wandering mobile robot

Also, to show consistent performance within a single domain, we segmented six additional different samples of English text:

- James Joyce's *Portrait of the Artist as a Young Man*

- Mark Twain's *The Adventures of Huckleberry Finn*
- Jonathan Swift's *Gulliver's Travels*
- Nathaniel Hawthorne's *The Scarlet Letter*
- Charles Dickens's *A Tale of Two Cities*
- F. Scott Fitzgerald's *This Side of Paradise*

Several inputs are similar to those used in [1]; the Orwell text and the robot data are identical. To simplify comparison, each text is first reduced to all lowercase alphabetic characters, with all punctuation converted to spaces and redundant spaces removed. The resulting segmented text has exactly one space between each pair of neighboring words, and each word contains only alphabetic characters. This is used as the gold standard segmentation. The input to the algorithms is a copy of the gold standard with all spaces deleted.

The robot controller data from [1] is included to demonstrate effectiveness in a non-language domain. The pattern of activation for binary controllers on a Pioneer 2 mobile robot equipped with sonar and a video camera was recorded ten times per second for 30 minutes as it wandered around a room, interacting with objects it came across. The robot has five high level "behavioral" controllers and eight low level controllers (Figure 3). Each high level state triggers a sequence of low level states. The goal is to find the pattern of high level controller activation using only the low level data. Each token in the robot time series represents one of 65 internal low level states of the robot. (There are actually $2^8 = 256$ different combinations, but only 65 occur in the data.) An episode is the consecutive series of low-level states that occur during a particular high-level behavior.

High level "behavioral" controllers	Low level controllers
FLEEING	MOVE-FORWARD
WANDERING	TURN
AVOIDING	COLLISION-AVOIDANCE
ORBITING-OBJECT	VIEW-INTERESTING-OBJECT
APPROACHING-OBJECT	RELOCATE-INTERESTING-OBJECT
	SEEK-INTERESTING-OBJECT
	CENTER-CHASIS-ON-OBJECT
	CENTER-CAMERA-ON-OBJECT

Figure 3: Robot controllers

In contrast with text, the robot episodes are very noisy with poor structure. Because robot state remained constant for unpredictable durations and was measured multiple times per second, episodes mainly consist of series of repeating symbols and vary widely in length. Other episodes represent instantaneous controller activations that happen between persistent states. These episodes consist of only one symbol, providing poor context and undefined structure.

Text inputs are truncated to the first 50,000 characters for most experiments. The robot time series contains 17,798 tokens. Each time series is segmented twice, once using the original VOTING EXPERTS algorithm, and once with the addition of the MARKOV EXPERT.

5.2 Evaluation Metrics

We use the same metrics as [1] to evaluate our algorithms. The success of a segmentation is evaluated numerically using two main classes of metrics: F-measure and word capture. F-measure treats the segmentation as a search problem and relates the precision and recall by which episode boundaries are found:

Precision: Percentage of boundaries in the correct segmentation that were inferred successfully by the experts.

Recall: Percentage of attempted boundary insertions which were correct. False Positive Rate is 100% – Recall.

F-measure: $$\frac{2 \times precision \times recall}{precision + recall}$$

Precision and recall are generally inversely related: a high precision comes at the expense of a low recall, and a high recall comes at the expense of low precision. For example, inserting a boundary at every location in the input ensures 100% precision but very low recall. In VOTING EXPERTS, this tradeoff can be controlled by tuning the threshold number of votes for inserting a boundary. If the threshold is low, more boundaries will be inserted, boosting precision but also resulting in more false positives (and a lower recall rate). If the threshold is low, there may be less false positives (higher recall rate) but precision may suffer. In order to have a result measure that is not skewed by these tradeoffs, we define the F-measure to be the harmonic mean of precision and recall. F-measure is a fairly reliable measure of success in segmenting text.

Another way to visualize the success of each algorithm is in terms of the number of actual words captured by the inferred boundaries, and how completely they are captured. These results are obtained by comparing the inferred word boundaries with the correct word boundaries. To discuss word capture, we use the *exact, dangling* and *lost* measures from [1] and also introduce three new measures: *broken, captured* and *partially captured.*

Exact: The word is perfectly captured, bounded on both sides with no false positive boundaries within.

Dangling: The word is bounded only on one side, with no false positives within

Broken: Anything that is not Exact or Dangling

Captured: The word is bounded on both sides, but may have false positive boundaries within (Captured is a superset of Exact).

Partially Captured: The word is bounded only on one side, but may have false positives within (Partially Captured is a superset of Dangling).

Lost: None of the word's boundaries have been found.

5.3 Segmentation Results

The MARKOV EXPERT improved the performance of VOTING EXPERTS in both F-measure and word capture. In the trials shown in Figure 4, voting thresholds were adjusted to obtain the best possible F-measures for each algorithm on each input. The MARKOV EXPERT increased F-measures by an average of 6.8% over the seven different domains with a standard deviation of 1.5%. The number of episode captures increased by an average of 23.2%, exact captures increased by 24.6%, and lost episodes decreased by 35.8%.

	F-measure	Precision %	Recall %	Exact %	Captured %	Lost %
Orwell (English)						
VOTING EXPERTS	75.3	74.8	75.8	44.4	54.7	5.1
MARKOV EXPERT	79.8	80.9	78.7	54.2	65.5	3.6
Song lyrics (Romaji)						
VOTING EXPERTS	64.2	65.3	63.1	29.6	41.3	10.8
MARKOV EXPERT	67.6	72.7	63.1	37.2	52.3	5.0
Goethe (German)						
VOTING EXPERTS	69.9	74.2	66.0	41.4	54.6	6.1
MARKOV EXPERT	75.2	81.5	69.7	50.7	66.3	3.3
Verne (French)						
VOTING EXPERTS	68.2	71.3	65.4	37.0	50.3	7.7
MARKOV EXPERT	72.9	75.7	70.3	43.5	57.4	6.0
Arrighi (Italian)						
VOTING EXPERTS	70.0	71.7	68.3	38.4	50.8	7.3
MARKOV EXPERT	76.2	81.3	71.7	50.3	66.1	3.6
Heidenstam (Swedish)						
VOTING EXPERTS	73.1	74.0	72.2	43.0	54.0	6.0
MARKOV EXPERT	79.1	81.7	76.7	55.5	67.3	3.9
Robot Data						
VOTING EXPERTS	67.5	59.9	77.3	33.2	37.0	17.3
MARKOV EXPERT	70.8	65.8	76.6	41.3	46.6	14.9

Figure 4: Best possible F-measures with corresponding Precisions, Recall Rates and word captures

The MARKOV EXPERT was especially effective in segmenting English text. Performance was consistent across seven inputs, maintaining an average 6.5% improvement in F-measure with a standard deviation of only 0.7%. Captures and exact captures increased by 33.2% and 29.5% while lost episodes decreased by 50.3%.

	F-measure	Precision %	Recall %	Exact %	Captured %	Lost %
Orwell						
VOTING EXPERTS	75.3	74.8	75.8	44.4	54.7	5.1
MARKOV EXPERT	79.8	80.9	78.7	54.2	65.5	3.6
Joyce						
VOTING EXPERTS	76.8	73.2	80.9	44.9	51.7	5.4
MARKOV EXPERT	82.1	85.5	78.9	59.9	72.9	2.0
Twain						
VOTING EXPERTS	76.5	71.4	82.4	43.8	49.3	6.5
MARKOV EXPERT	82.5	83.5	81.6	59.6	69.3	2.4
Swift						
VOTING EXPERTS	74.9	72.8	77.0	42.7	51.7	6.1
MARKOV EXPERT	79.3	84.4	74.7	55.8	71.2	2.4
Hawthorne						
VOTING EXPERTS	71.5	72.8	70.3	40.0	51.8	6.1
MARKOV EXPERT	75.6	81.2	70.8	49.9	65.8	3.5
Dickens						
VOTING EXPERTS	74.1	73.1	75.2	43.0	52.4	6.3
MARKOV EXPERT	79.3	83.0	76.0	56.1	69.2	3.1
Fitzgerald						
VOTING EXPERTS	72.2	72.5	72.0	40.7	51.6	6.6
MARKOV EXPERT	76.7	81.9	72.1	52.4	67.6	3.8

Figure 5: Best possible F-measures with corresponding Precisions, Recall Rates and word captures for English inputs

Figure 6 shows differences in precision and false positive rates (F.P. Rate = 1 – recall rate) when the other measure is held constant. Here, we use the VOTING EXPERTS precision and false positive rates from the above trials. The MARKOV EXPERT's threshold is adjusted to (approximately) match one of these measures at a time. While making the same proportion of errors, the MARKOV EXPERT increases precision by 13.3% on average. Conversely, the false positive rate decreases by 29.4% when precision is held constant. That is, in order to infer a given amount of boundaries, the MARKOV EXPERT makes about 29.4% fewer mistakes than the original VOTING EXPERTS.

	Precision %, when F.P. Rate is held constant		F.P. Rate %, When Precision is held constant	
	VOTING EXPERTS	MARKOV EXPERT	VOTING EXPERTS	MARKOV EXPERT
Orwell (English)	74.8	83.8	24.2	17.4
Song lyrics (Romaji)	65.3	72.7	36.9	32.4
Goethe (German)	74.2	84.6	34.0	25.9
Verne (French)	71.3	81.9	34.6	22.3
Arrighi (Italian)	71.7	83.9	31.7	20.4
Heidenstam (Swedish)	74.0	84.8	27.8	13.5
Robot Data	59.9	65.8	22.7	18.4

Figure 6: Precisions and F.P. Rates with opposite measure held constant

Figure 7 shows average performance over the seven different domains in terms of word capture with the false positive rate held constant. This shows that the MARKOV EXPERT captured far more and lost fewer words while making roughly the same number of inferences.

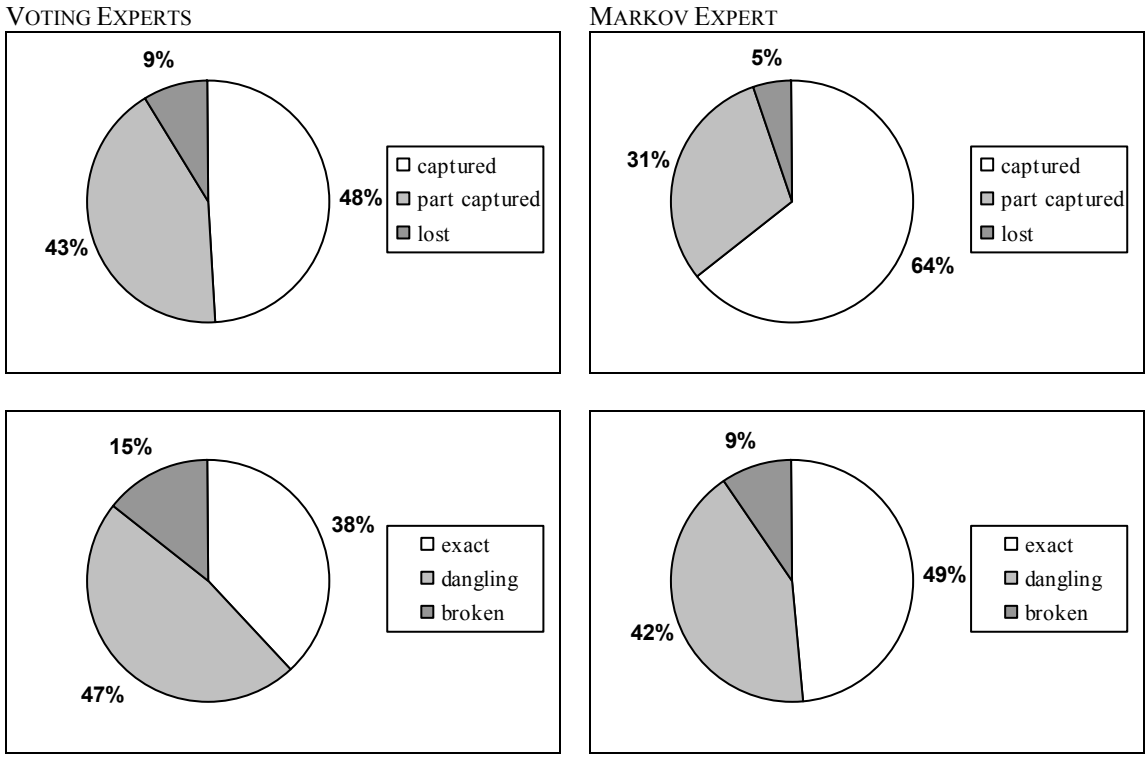

Figure 7: Average word capture over seven different domains, holding F.P. Rate constant

For the text inputs, a final way to gauge performance is to subjectively and visually evaluate the readability of the segmented output. The following are segmented output from a passage in the first 50,000 characters of James Joyce's *Portrait of the Artist as a Young Man*.

... perhaps a wild rose might be like those colours and he remembered the song about the wild rose blossoms on the little green place but you could not have a green rose but perhaps somewhere in the world you could the bell rang and then the classes began to file out of the rooms and along the corridors towards the refectory he sat looking at the two prints of butter on his plate but could not eat the damp bread the tablecloth was damp and limp but he drank off the hot weak tea which the clumsy scullion girt with a white apron poured into his cup he wondered whether the scullions apron was damp too or whether all white things were cold and damp...

Figure 8: Correctly segmented output

... perhaps a wildrose might be like th ose colour s and here member ed the song about the wildrose blo ssomsonthe little green place but you could not havea green rose but p erhaps some where inthe world you could the bell rang andthe n the class es beganto file outof the room s and along the co rridor s to wards the re fect ory hesat look ing atthe two print sof butter onhis plate but could not eatthe damp bread the table clo th was damp and li mpbuthe drank off the hot weak tea which the clumsyscu llion girtwith a white apronpoured into his cuphewond ered whe therthe scullion sapron was damp tooorwhe the r allwhite th ing s were coldand damp ...

Figure 9: Original VOTING EXPERTS output

... perhaps a wild rose might be like those colour s and here member ed the song about the wildrose blossoms on the little green place but you could no t have a green rose but perhaps some where in the world you could the bell rang and the n the class es began to file outof the rooms and along the co rridors to wards the re fect ory hesat look ing at the two pr intsof butter on his plate but could not eat the damp bread the table clo th was damp and limpbu the drank off the hot weak tea which the clumsysculli on girt wi th a white apron poured into his cup hewondered whe the r the scullion sapron was damp tooorw he the r all white thing s were cold and damp ...

Figure 10: MARKOV EXPERT output

The MARKOV EXPERT makes several corrections in cases where the original VOTING EXPERTS failed. For example, the words "rose", "those" and "blossoms" are segmented incorrectly by the original VOTING EXPERTS but are fixed by the MARKOV EXPERT. At the same time, the MARKOV EXPERT retains almost all of the words that VOTING EXPERTS captured correctly.

5.4 Influence of Input Length

Teahan et al. showed that the performance of their method improved with the size of the training corpus. For the MARKOV EXPERT, since the training corpus is formed from the input, we expect performance to increase with respect to input length. The following experiments monitor the F-measure of both the original VOTING EXPERTS and the MARKOV EXPERT as the input length changes. The James Joyce text was used for these trials.

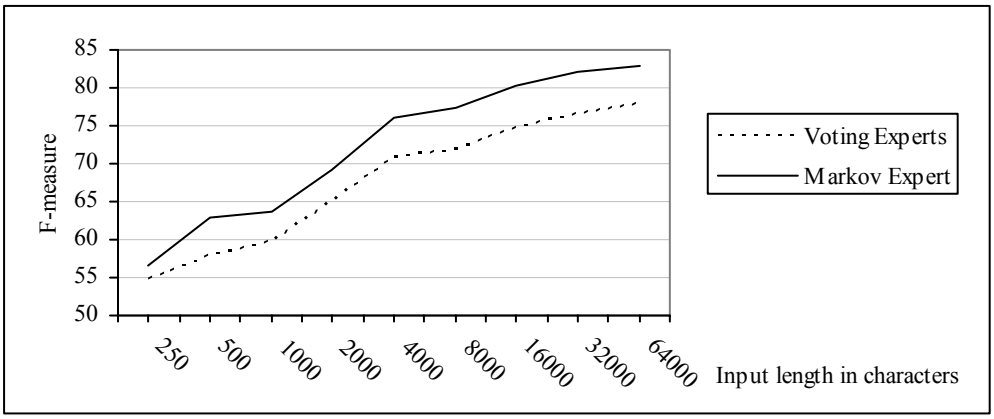

Figure 11: Influence of input length on F-measure

6 Applications

In addition to text segmentation and robot learning, finding meaningful episodes in time series is useful in a large number of practical applications.

In speech processing, input is likely to be presented as a raw, unsegmented sequence of phonemes since natural speaking does not exhibit pauses between words. Using our method, raw speech input could be segmented into words for easier processing by subsequent components.

Image processing and computer vision can also benefit from episode discovery. Pixels in an image can be treated as a time series, and clusters of pixels extracted as episodes might correspond to objects and object boundaries in the image.

Similarly, network traffic and economic behavior patterns could each be modeled as time series and thus benefit from our approach.

7 Conclusions and Future Work

The MARKOV EXPERT has proved to be an effective augmentation to the VOTING EXPERTS algorithm of Cohen et al. The MARKOV EXPERT takes advantage of the good performance from the original VOTING EXPERTS, and harnesses this output to reassess the input using a Markov model and further boost performance. The MARKOV EXPERT, like the original VOTING EXPERTS, does not perform well on smaller inputs. This is the major limiting factor to making the MARKOV EXPERT truly portable across domains. However, there are many domains that do provide large enough inputs to realize the MARKOV EXPERT's benefits.

One direction for future work is to evaluate the MARKOV EXPERT on additional domains. The different application areas discussed above may generate different flavors of time series. It would be interesting to explore how our method behaves across different kinds of time series.

Tuning parameters such as window sizes, voting thresholds and relative weighting of experts force the need for cursory human intervention. In some cases, where precision and recall rates cannot be readily calculated, it may be hard to determine the correct parameters in the first place. Another area of future work is to study automatic tuning of such parameters.

Because the MARKOV EXPERT requires large input, there may be delays in responding to a user when the input stream is slow. It would be useful to explore incremental processing of input so that users can see approximate results as they wait for all of the input to be processed.

Finally, experiments in this study focused on feeding the MARKOV EXPERT corpuses that maximized F-measures. Although F-measure is a good gauge of success for the final output, the MARKOV EXPERT might possibly work better with corpuses that are higher in precision or recall rate and suboptimal in F-measure.

8 Acknowledgments

This work was supported in part by NSF grants CCR-9983832 and CCR-0121154.

9 References

[1] P. Cohen, B. Heeringa, and N. Adams. An Unsupervised Algorithm for Segmenting Categorical Timeseries into Episodes. *Proceedings of the 2002 International Conference on Data Mining,* p. 99-106.

[2] W.J. Teahan, Y. Wen, R.J. McNab, and I.H. Witten. A Compression-based Algorithm for Chinese Word Segmentation. *Computational Linguistics,* 26: 375-393, 2000.

[3] K. Cheng, G.H. Young, and K. Wong. A Study on Word-based and Integral-bit Chinese Text Compression Algorithms. *Journal of the American Society for Information Science,* 50(3):218-228, 1999.

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

[5] I.H. Witten, Z.Bray, M. Mahoui and W.J. Teahan. Text mining: a new frontier for lossless compression. *Proceedings of the Data Compression Conference*, 198-207, 1999.

[6] W.J. Teahan and J.G. Cleary. The entropy of English using PPM-based models" *Proceedings of the Data Compression Conference*, 53-62, 1996.