Mining Temporal Patterns of Movement for Video Content Classification

Michael Fleischman
Cognitive Machines Group
The Media Laboratory
Massachusetts Institute of Technology
mbf@mit.edu

Phillip Decamp
Cognitive Machines Group
The Media Laboratory
Massachusetts Institute of Technology
decamp@media.mit.edu

Deb Roy
Cognitive Machines Group
The Media Laboratory
Massachusetts Institute of Technology
dkroy@media.mit.edu

ABSTRACT

Scalable approaches to video content classification are limited by an inability to automatically generate representations of events that encode abstract temporal structure. This paper presents a method in which temporal information is captured by representing events using a lexicon of hierarchical patterns of movement that are mined from large corpora of unannotated video data. These patterns are then used as features for a discriminative model of event classification that exploits tree kernels in a Support Vector Machine. Evaluations show the method learns informative patterns on a 1450-hour video corpus of natural human activities recorded in the home.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning –*knowledge acquisition.*

General Terms

Algorithms, Experimentation.

Keywords

Temporal Data Mining, Video Content Classification, Video Event Recognition, Tree Kernel, Support Vector Machine.

1. INTRODUCTION

Just as the rise of the internet saw an explosion in available text corpora, the falling prices of digital video cameras and storage media have set the stage for a similar proliferation of personal video recordings. Our ability to search through and index these new resources is dependant upon techniques that can automatically classify the content of events in video. Although simple events can often be modeled based on image features extracted from key frames, many complex events require more structured models for accurate classification. Researchers have made some progress in this task by modeling the temporal structure inherent to complex tasks (e.g., [6], [8], [9]). While such work generally relies on hand built models, scalable approaches to video content classification will only be fully realized by automatically generating representations that encode the temporal structure of events. In this paper, we present a novel approach to learning such representations in which a lexicon of hierarchical patterns of human movements is mined from unannotated video data and then used to train a discriminative model of event classification.

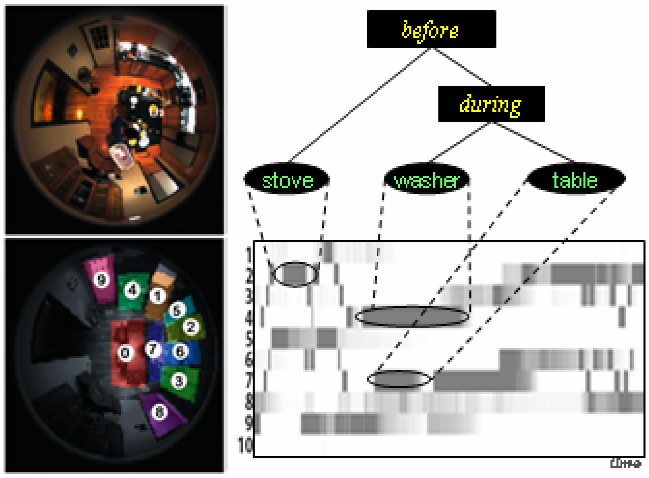

Figure 1. A static overhead view of the kitchen as recorded in the Human Speechome Project database (above left). By tracing regions of interest over objects, such as the stove and table, motion detection can be localized to specific areas of a room (below left). This allows for sequences of video data to be represented as multi-variate time series (below right). By thresholding over these series, movement events are extracted and hierarchical temporal relations between them can be discovered (above right).

A great deal of work in event classification has focused on representing the temporal structure of complex events. Much of this work, however, has focused on modeling the dynamics between low level features in video data and does not, in general, address the more abstract temporal relations that make up complex events. Such relations are extremely useful, though,

MIR '06, October 26–27, 2006, Santa Barbara, CA, USA.

because complex events are often composed of sub-events that occur in varied temporal and hierarchical relations to each other. For example, Figure 2 shows how the complex event *making coffee* can be decomposed into the sub-events: *grinding beans* and *filling the pot*, which can occur simultaneously, and *turning on the machine,* which must occur after the others have completed.

Recently a number of researchers have developed techniques to model such abstract temporal information by hand using, for example, probabilistic context free grammars [9] and temporal logics [6]. However, it is unclear how well such formalisms may be automatically learned from large real-world video corpora. For example, while [5] shows that models based on temporal logic can be learned from data, his approach only handles a limited set of relations (i.e. *before* and *at the same time*) and is only tested on small sets of staged events (e.g. putting an object on a table using carefully controlled movements). Also, although algorithms for learning probabilistic context free grammars may scale to large corpora (e.g., [10]), such grammars can only encode strictly sequential relations (e.g. *before* and *after*, but not *during* or *overlap*) and thus may not be appropriate for classifying many types of complex events.

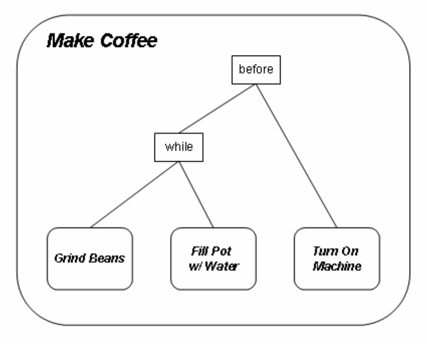

Figure 2. A graphical representation of the event *making coffee* shows how complex events can be composed of sub-events in various hierarchical temporal relationships.

In this paper, we present a novel methodology to facilitate learning temporal structure in event classification. Complex events are modeled using a lexicon of hierarchical patterns of movement, which are mined from a large corpus of unannotated video data. These patterns act as features for a tree kernel [3] based Support Vector Machine (SVM) that is trained on a small set of manually annotated events. Evaluations on a large corpus of real-world video data show the performance of the system to be above a baseline Hidden Markov Model. These results confirm the effectiveness of the pattern mining algorithm, and further, suggest a new approach to event classification that achieves high accuracy with minimal human effort (i.e. small human annotated data sets combined with a larger amount of unannotated data).

The remainder of the paper is organized as follows: first, we describe our corpus of real-world video data, collected as part of the Human Speechome Project [13], and discuss methods for data pre-processing and automatic detection of low level movement. We then detail how the hierarchical patterns of movement are mined from this unannotated video data. Next, we describe how these learned patterns are used as features in a discriminative model of event classification. Finally, evaluations of the acquired patterns and the overall system are presented.

2. THE HUMAN SPEECHOME PROJECT

The Human Speechome Project (HSP) is an effort to observe and computationally model the longitudinal course of language development for a single child at an unprecedented scale [13]. In pursuing this goal, the home of a newborn infant has been instrumented so that each room in the house contains a microphone and video camera that record audio and video data. Now in the sixth month of a three year study, approximately 24,000 hours of audio and video data have been collected, averaging about 300 gigabytes of data collected per day. The video data used throughout this work represents a subset of this HSP corpus; namely, about 1450 hours of video (ceiling mounted omnidirectional camera, 1 megapixel resolution, 14 frames per second) collected from the kitchen during the first two months of the project.

This data represents an unedited and highly complex domain of real-world human activity: the patterns of life for a family at home. Events are unscripted, often complex, sometimes multi-agent, and rarely easy to find. These factors make it a challenging test bed for work in video content classification, but relevant for applications in security, smart homes, and many other domains.

Although such real-world data presents many challenges, this corpus offers certain advantages for event classification as well. In particular, the omnidirectional cameras offer a full, top-down view of each room from a fixed and known position. Such static positioning simplifies the task of motion detection, allowing for high precision using relatively simple techniques. In this work we employ an algorithm from Computer Vision research [11] in which a pixel is considered to have motion if the luminosity of that pixel changes by some threshold. Thresholds are continually updated based on each individual pixel's mean and variance in order to allow for spurious intensity changes not due to observed movements (e.g., lighting changes due to shifting cloud cover). This algorithm allows for online motion detection of the video data that is used for both storage compression (see [13]) as well as data analysis.

Another aspect of the HSP data which we exploit in our analysis is the fact that home environments are populated with many objects (particularly furniture and large appliances) that are stationary and are rarely moved to new locations. These objects can be traced out as regions of interest that have special semantic interpretations when motion is detected in their vicinity. Figure 1 shows regions of interest traced over a static image coming from the camera in the kitchen of the HSP environment. Here, Region 3 is traced over the refrigerator. Thus, any motion detected within Region 3 can be interpreted as motion near the refrigerator. By tracing out many such regions (e.g. the sink, table, dishwasher) we can convert the video data into a compact representation that captures the level of motion in each of a number of hand-specified regions of interest. Figure 1 shows a graph of this representation with time extending on the x-axis, each region of interest represented as a different row on the y-axis, and the level of motion in that region represented by the region's brightness (the darker the region, the more motion). These multi-variate time series representations not only provide a convenient way to view large amounts of video statically (see [13]), but by drastically reducing the dimensionality of video (i.e. from 960 pixels x 960 pixels per frame to 10 regions per frame) they provide a compact basis for mining hierarchical patterns of movement.

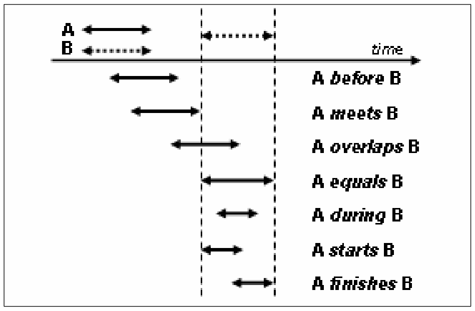

Figure 3. Any two events must be in one of seven asymmetric temporal relations described by Allen (1984).

3. MINING PATTERNS OF MOVEMENT

In this work, events are modeled using a lexicon of hierarchical patterns of movement that capture aspects of the underlying temporal structure of a complex event. In this section we describe how these hierarchical patterns are mined from the large unannotated HSP corpus. As described in section 2, the video data from HSP can be represented as a multi-variate time series of movements within regions of interest that have been manually-defined over semantically relevant areas of a room. We equate the problem of learning hierarchical patterns of movement with that of finding significant temporal relationships between movements in these regions of interest.

Although some temporal relationship must exist between movements in any two regions of interest, some are more interesting than others. For example, although it may be the case that while someone was using the sink (i.e. movement in Region 1, Figure 1) someone else in the kitchen opened the refrigerator door (i.e. movement in Region 3, Figure 1), the temporal relationship between these two events is probably not going to be particularly relevant for modeling the structure of a complex event. On the other hand, the fact that someone is moving near the sink (i.e. Region 1, Figure 1) just before there is movement near the dishwasher (i.e. Region 4, Figure 1) may be an important pattern to detect because it often occurs when someone is washing the dishes.

Ideally, in generating a lexicon of hierarchical patterns, we would seek to separate those temporal relationships in the data that are useful for event classification from those that are not. However, since we do not know a priori which event types the lexicon will be used to represent, we make the simplifying assumption that the only patterns worth finding are those that occur significantly in the data. We now describe the steps necessary to discover such patterns.

The first step in finding significant patterns of movement is to threshold the multi-variate time series such that movement either is or is not occurring at any particular region at time t. Given this threshold, we can view each region as being a state indicator, and the continued occurrence of an active state may be treated as a low level movement event (e.g. Movement above the threshold from time t1 to time t2 in region 3 is considered a "refrigerator movement" with duration (t2-t1); see Figure 1).

```
LEARN-PATTERNS(matrix data)
  significant Events ← ∅
  counts ← ∅
  foreach timeslice t in data
      events ← FIND-COMPOSITE-EVENTS(t)
      foreach event F in events
      increment $counts_f$
          if F passes threshold
      add to significant Events
  return significant Events

FIND-COMPOSITE-EVENTS(vector t)
  candidateCompositeEvents ← ∅
  justFinishedEvents ← list of events ending at t
  stillActiveEvents ← list of events still open at t
  foreach event F in justFinishedEvents
      //find present relations
      FIND-RELATIONS(F, justFinishedEvents)
      //find future relations
      FIND-RELATIONS(F, stillActiveEvents)
      //find past relations
      FIND-RELATIONS(F, STM)
  updateSTM()
  return candidateCompositeEvents

FIND-RELATIONS (event F, list eventSet)
  foreach event G in eventSet
  compositeEvent ← temporal relation btw F and G
  if compositeEvent is reliable
      push onto significant Events
  else
      push onto candidateCompositeEvents
```

Figure 4. Pseudo-code for mining hierarchical patterns of movement from large unannotated video corpora.

Given such movement events, we can categorize the temporal relations between them using the set of temporal relations outlined by Allen [1]. Allen suggests 7 symmetric temporal relations that may be used to classify relations between pairs of time periods as shown in Figure 3 (i.e. *meets, equals, during, before, starts, finishes*, and *overlap*). Further, we can speak of a relation as being hierarchical when one or more of the events related to each other are themselves relations between sub events (e.g. [A *meets* [B *before* C]] or [[A *before* B] *equals* [C *during* D]]). These hierarchical events have an order, that corresponds to the depth of that event (e.g. 1st order: [A *meets* B]; 2nd order: [[A *meets* B] *before* C]; 3rd order: [[[A *meets* B] *before* C] *during* D]; etc…).

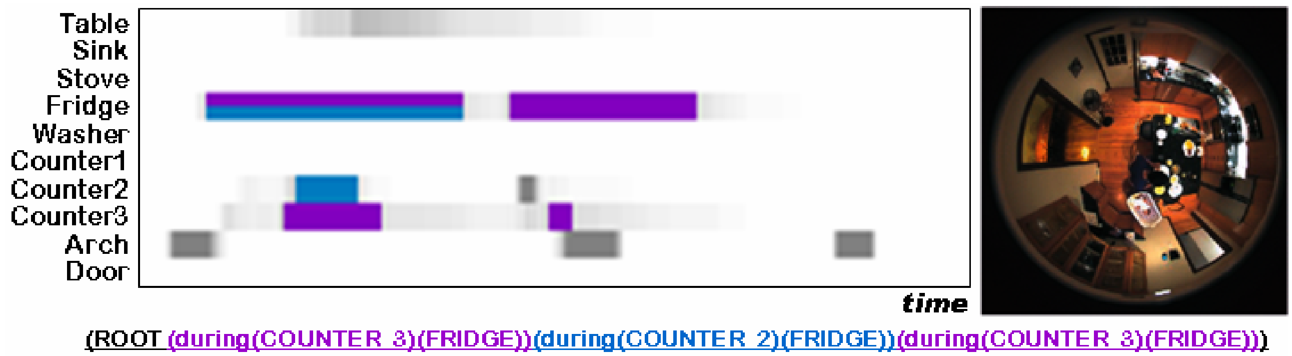

Figure 5. Significant patterns are matched in multi-variate time series representations of events. These patterns are collected together under a dummy root node for use with a tree kernel based Support Vector Machine. Single patterns can be instantiated multiple times in an event and individual events can participate in multiple patterns.

We equate the task of discovering patterns of movement with that of identifying hierarchical movement events that occur significantly in the large set of unannotated video data. To discover such events, two algorithms are combined: one originally developed in the domain of weather analysis [7], the other from robotics [2]. While both seek to find hierarchical patterns in unannotated data, Hoppner's algorithm is preferable in its manner of detecting related events, while Cohen's algorithm has a more stringent thresholding method. The algorithm presented here seeks to exploit useful aspects of both approaches.

We present pseudo-code for our algorithm in Figure 4. The algorithm processes the multi-variate time series frame by frame, at each point checking if any low level movement events have just ended (i.e., the state indicator for a region has gone to zero). If this has happened, that low level movement event is compared with the events in three different sets: 1) the set of events that also have just ended; 2) the set of events that are still ongoing; and 3) the set of events that have recently ended. We define a time limited Short Term Memory (STM) in which these recently completed events are stored. The size of this STM acts as a windowing length such that only events within this window can be compared.[1]

For each pair of events (i.e., the newly ended event and the event selected from one of the three sets), the temporal relation that exists between them is calculated. [2] This relation can now be treated as a new event that is composed of the two sub-events. The number of occurrences for each composed event is maintained. If the composite event has already been found significant (e.g., in a previous iteration), it is added to the set of events that just ended, and is itself composed with the other events as described above. By recursively adding events that were previously found significant, the system is able to discover hierarchical patterns of movement.

Once the algorithm has examined all the frames in the dataset it cycles through each observed composite event and checks if that event is significant.[3] Similar to [2], we use the phi statistic to measure the significance of an event. For each composite event, we create a 2-by-2 contingency table that describes how often different sub-events of the composite were observed in that temporal relation. For example, in the contingency table shown in Table 1, A represents how often a table event occurred during a washer event, B represents how often a table event occurred during any other type of event, C represents how often a non-table event occurred during a washer event, and D represents how often any non-table event occurred during any non-washer event.

Table 1. Contingency table used to calculate significance of event during(table,washer)

during	washer	¬ washer
table	A	B
¬ table	C	D

Phi can now be calculated using equation 1, in which χ^2 is the chi square statistic calculated from the contingency table and N is the table's total. The phi statistic provides a measure between 0 and 1 of the strength of the association between the sub-events in a composite event and can be tested for statistical significance as with a Pearson r. In order for a composite event observed by our system to be considered significant, its phi must both be greater than some value *rho* as, as well as, significant above a threshold *alpha.*

$$\phi = \sqrt{\frac{\chi^2}{N}} \qquad \textbf{(1)}$$

In the experiments presented here, the algorithm was set such that each iteration produced significant patterns of increasingly higher orders (e.g., the 1st iteration produced 1st order patterns, the 2nd produced 2nd order patterns, etc.). After all iterations are completed, the output of the program is a set of significant

[1] This differs from STM in [2] which is limited by size, not time, and thus allows comparisons between events that occur arbitrarily far apart in time.

[2] Because of noise in the data, relations are based on soft intervals. E.g., A meets B iff (end of A)-α < (start of B) < (end of A)+α.

[3] Like [2], the algorithm can learn events online by checking for significance after each frame.

hierarchical patterns of movement discovered from the unannotated data.

Having described how a lexicon of hierarchical patterns is discovered, we now present a method for using the lexicon to represent events for classification.

4. CLASSIFYING EVENTS

As is the case in many domains, successful classification of events depends upon the choice of representation for the data. Here we describe a method in which video events are represented using the lexicon of hierarchical patterns of movement described above. We treat video content classification as a discriminative problem and train a classifier using a small training set of labeled events. In a discriminative classification framework, each event in the training set is represented as a compact vector of features. The hierarchical patterns of movement that are observed in video events are used as features for representing those events.

For each event in the training set, all movement events that occur within it are examined (e.g., [fridge-movement *while* [sink-movement *before* counter-movement]]), and if that event is in the lexicon of significant patterns learned above, it is used as a feature.

This examination is easily achieved using the same framework employed to discover significant patterns. The algorithm described above (Figure 4) was modified such that, instead of counting *all* observed movements, it only counts those that were previously learned to be significant. In this way, the learned lexicon acts as a filter for removing unreliable features that may be noisy and uninformative. Further, it allows for a massive reduction in the feature space; for, by not filtering the movement events, feature vectors can suffer an exponential explosion in size (see Section 5 for more details).

One of the benefits of using the hierarchical patterns of movement as features is that the structure inherently captures a great deal of valuable information. For example, even though a given pattern (e.g., [fridge-movement *while* [sink-movement *before* counter-movement]]) may not be useful for classification in itself, it may be the case that a sub-element of that pattern (e.g., [sink-movement *before* counter-movement]) is informative for classification. Fortunately, researchers have developed methods to capture this type of structural information using tree kernels [3], a dynamic programming technique that affords an efficient means of comparing hierarchically structured representations of data.

Just as the phrase structure of sentences can be represented using syntactic trees, we represent video events using hierarchical patterns of movement. For each event in the training set, all significant observed patterns of movement that occur within the event are parsed out and joined together under one dummy root node. In this way, each event can be re-described as one tree feature (see Figure 5), which is used to train a Support Vector Machine using a tree kernel. By treating event classification in this way, we are able to exploit a large dataset of unannotated data to generate features for a small dataset of annotated data. In the following section, we evaluate the performance of this framework.

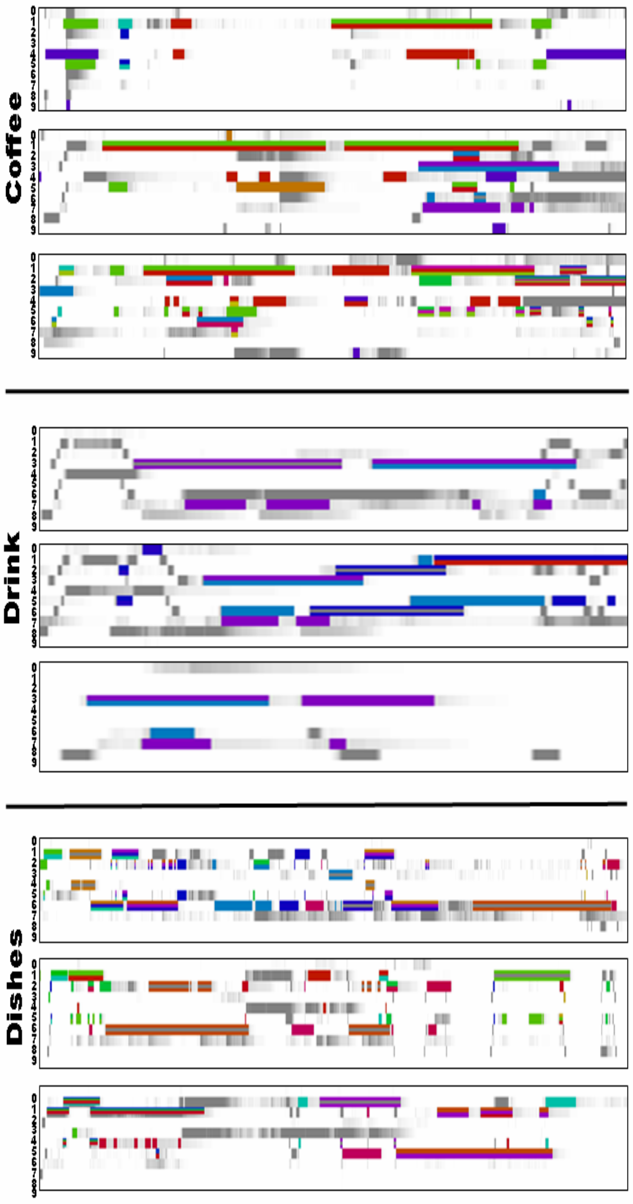

Figure 6. Significant patterns of movement are displayed on examples of different event types. Each significant pattern is given a unique color, showing the different distributions of patterns for each event class. Time scales are not equivalent between event exemplars.

5. EVALUATIONS

5.1 Data

We evaluate our approach using a subset of the HSP data: the first two months (approximately 1450 hours) of video data collected by the kitchen camera. Hierarchical patterns of motion were learned from this unlabeled corpus. The algorithm was run for four iterations, such that each iteration produced patterns of

Table 2. Example outputs of the pattern mining algorithm are shown for four hierarchical orders of pattern along with their phi statistic and the frequency with which they were observed in the unannotated data.

Pattern	*phi*	Freq
1st Order Patterns		
equals(TABLE)(COUNTER_TABLE):	0.081	9147
equals(COUNTER_TABLE)(WASHER):	0.064	8339
2nd Order Patterns		
equals(before(SINK)(COUNTER_FRIDGE)) (before(TABLE)(COUNTER_SINK)):	0.055	53
starts(equals(TABLE)(COUNTER_TABLE)) (FRIDGE):	0.059	1443
3rd Order Patterns		
equals(overlap(before(COUNTER_SINK)(FRIDGE)) (before(DOOR)(WASHER))) (before(STOVE)(COUNT_TABLE):	0.050	14
equals(overlap(before(COUNTER_SINK)(FRIDGE)) (before(COUNTER_FRIDGE) (SINK))) (finishes (before(DOOR) (WASHER))(COUNTER_TABLE)):	0.198	20
4th Order Patterns		
meet(overlap(overlap(during(COUNTER_SINK)(STOVE)) (during (COUNTER_SINK)(SINK))) (during(COUNTER_FRIDGE) (STOVE)))(during(COUNTER_FRIDGE) (STOVE)):	0.057	120
meet(before(during(COUNTER_FRIDGE)(STOVE)) (during (STOVE)(during(STOVE)(COUNTER_FRIDGE)))) (starts(STOVE)(COUNTER_SINK)):	0.063	150

successively greater order (initial results did not indicate improvement beyond 4th order). The learning algorithm has 3 parameters: a short term memory (STM) size, an *alpha* to control the level of significance, and a *rho* to control the strength of the association between sub events. In these experiments the parameters were set as follows: STM=300 frames (~50 sec), alpha =0.95, rho=0.05. Further, there are other parameters used to threshold the activity ribbons into binary low level events. These include a motion threshold (set to .15) and a low pass filter window (set to 3). Both of these sets of parameters were optimized on a small held out validation set. The total number of unique patterns of activity discovered by the algorithm using these settings is as follows: 38 1st order patterns, 115 2nd order patterns, 245 3rd order patterns, and 418 4th order patterns. Table 2 shows example patterns of each order.

In order to evaluate these patterns' usefulness for event classification, a small set of training instances (approximately 130 minutes) was hand labeled. This set consisted of three event classes in the following distribution: 13 examples of *making coffee,* 9 examples of *putting away the dishes*, and 13 examples of *getting a drink*. Figure 6 shows multi-variate time series graphs for a subset of these events in which all significant patterns have been uniquely colored. All classification tests were done using leave one out cross validation. Held out validation sets were used to optimize the parameters of a tree kernel SVM implementation from [12].

5.2 Experiments and Results

Figure 7 shows a comparison of the event classification performance for the system trained using the learned 1st order patterns versus the learned 2nd order patterns. These results are compared against two baselines: 1) the results of a classifier that always chooses the most frequent class; and 2) the results of a simple 1st order Hidden Markov Model (HMM) trained on the multi-variate time series data of motion in each region of interest (i.e., the same time series used to generate the hierarchical patterns upon which the SVM is trained). The HMM baseline system employed a mixture model of two Gaussians and had the number of states set using cross-validation. The figure shows that the 2nd order pattern system outperforms both baselines as well as the 1st order system. While this result is significant for the simple frequent baseline (p>.05), because of small sample sizes, no other differences are significant.

Table 3 and 4 show the confusion matrices for the system trained on 2nd order patterns and the HMM baseline. For both systems, confusion is greatest for the *putting away the dishes* events. This confusion is not surprising as such events match a great many significant patterns, many of which are also seen with the other two event types. This can be observed in Figure 6 as well, where the coloring of *making coffee* events (mostly red and green) differs from the coloring of *getting drink* events (mostly purple and blue), while *putting away dishes* events share colors of both.

Figure 8 shows a comparison between the system trained on varying orders of patterns, from 1st order to 4th order, against the system trained in the same manner (i.e. using hierarchical patterns of movement and tree kernel SVM) but without using the lexicon of significant patterns as a filter. Again, results of choosing the most frequent class are presented as a baseline. Results are not available for the unfiltered system using patterns of 3rd order and above because their feature representations were too large for the SVM implementation used. Although not the case for strictly significant patterns, there is an exponential increase in the number of observed patterns of motion as the order of patterns examined increases. Thus, while only 77 2nd order patterns were used in the filtered system, 9057 2nd order patterns were used in the unfiltered system. The data sparsity that follows from this elucidates the consistent advantage of the filtered over the unfiltered system shown in Figure 8.

Finally, Figure 9 shows the performance of a hybrid system in which the hypothesized class from the HMM model (i.e. either *making coffee*, *putting away dishes*, or *getting drink*) is used as a feature, along with the hierarchical pattern features, in the tree kernel SVM.[4] Both the SVM with pattern only features and the

[4] The SVM implementation supports combining Tree Kernels with standard Radial Basis Kernels (see [12]).

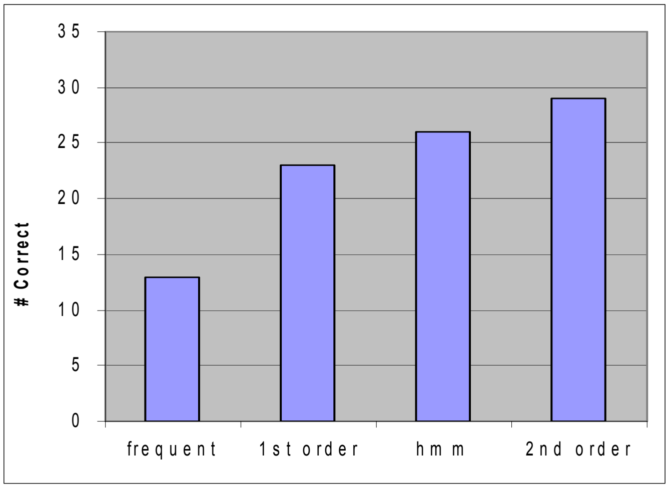

Figure 7. Comparison between event classification systems using significant patterns (1st and 2nd order) as features for a tree kernel based Support Vector Machine and two baselines: always choosing the most frequent class, and a simple Hidden Markov Model (hmm) implementation.

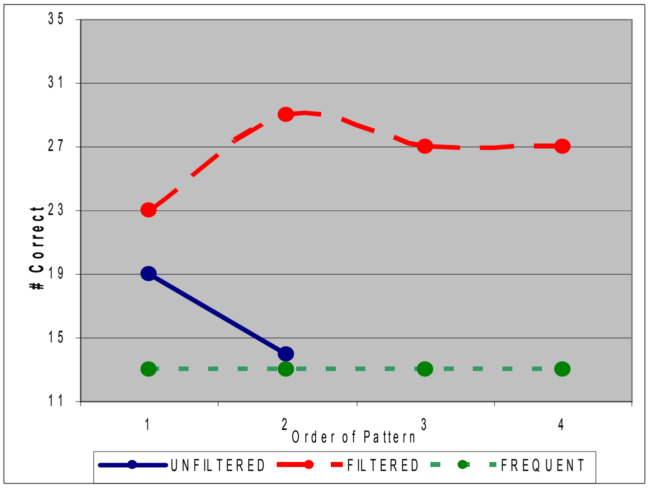

Figure 8. The system using only significant patterns of movement is compared t. a system using all observed patterns as features. Performance is given as a function of the order of patterns used and most frequent baseline is repeated.

HMM by itself are presented for comparison. Results are displayed for multiple orders of patterns. The figure shows that, although performance of the hybrid system is only minimally better than using the pattern features alone, there is consistent improvement across all orders of pattern.

DISCUSSION

The results of these evaluations show that the lexicon of hierarchical patterns of movement mined using the algorithm presented in section 3 is useful and informative for event classification. Figure 7, demonstrates that using 2nd order patterns as features for a discriminative model of event classification outperforms both a simple most-frequent baseline as well as a more traditional HMM approach. Further, Figure 8 shows that this result does not hold when all possible 2nd order patterns are used, but rather, is dependent upon using only those patterns in the lexicon learned by the method described in section 3. These findings validate our methodology for learning this lexicon and confirm the value of the temporal information encoded in the hierarchical patterns of movement.

The usefulness of these patterns for event classification stems from two distinct types of information that they encode: fine grained temporal relations and global information about events. First, by abstracting to the level of events as opposed to lower level observations of motion, the patterns allow for the encoding of more fine-grained temporal relations than traditional HMM approaches. Although both methods can capture relations such as *before,* only by representing both the starting and endpoints of motion can temporal relations such as *overlap* and *during* be represented (see [1]).

Additionally, hierarchical patterns of movement have the ability to capture global information about an event. Unlike HMMs that only encode local transitions between states, hierarchical patterns can capture temporal relations between movements that are not locally apparent. For example, the 2nd order pattern [sink-movement *before* [counter-movement *before* stove-movement] represents a relationship that exists between movements at the sink and the stove, a long distance relationship that is not expressible in dynamic models such as HMMs. The usefulness of this global information is evidenced by Figure 8, which shows the large difference in performance between the system using 1st order patterns, which do not encode global information, and the systems using higher order patterns, which do.

Table 3. Confusion matrix for Support Vector Machine trained on 2nd order significant patterns

[hyp->]	Coffee	dishes	drink	*Recall*
Coffee	13	0	0	*1*
Dishes	3	5	1	*0.55*
Drink	0	2	11	*0.84*
Prec.	*0.81*	*0.71*	*0.91*	

Table 4. Confusion matrix for Hidden Markov Model baseline

[hyp->]	Coffee	dishes	drink	*Recall*
Coffee	10	3	0	*0.76*
Dishes	2	6	1	*0.66*
Drink	2	1	10	*0.76*
Prec.	*0.71*	*0.6*	*0.9*	

Although capable of encoding the temporal and global characteristic of events, other useful sources of information are not captured by hierarchical patterns of movement. Unlike the HMM, which explicitly models the amount of motion in each region, the hierarchical patterns abstract away such amounts into strictly binary values (i.e. either there is movement in a region, or there is not). This movement information may be quite useful for classification, however, as evidenced by Figure 7 which shows greater performance for the HMM baseline compared to the system using only 1st order patterns (neither of which encode any global information). This suggests that a useful direction for future work is to examine methodologies for integrating the benefits of these two information sources.

Figure 9 shows results of a preliminary step toward achieving this. Although the method is extremely simple (i.e. using the output of the HMM as a feature in the tree kernel SVM) and the results are not yet significant, the hybrid system shows a consistent improvement in performance using each order of hierarchical pattern. Although not definitive, these results suggest that a fruitful area of future research will lie in finding more sophisticated techniques for combining dynamic models of event classification with the global information in hierarchical patterns of movement.

Another area of future work that we are exploring is the effect of running the algorithm over low level events that are more sophisticated than simple region movements (e.g., object tracking). Importantly, changing the nature of the low level events that the algorithm examines does not necessitate any change in the algorithm itself, i.e., the algorithm is agnostic to the nature of the events it operates on. This flexibility suggests that, as the low level events become more informative, so too will the hierarchical patterns that are mined from them.

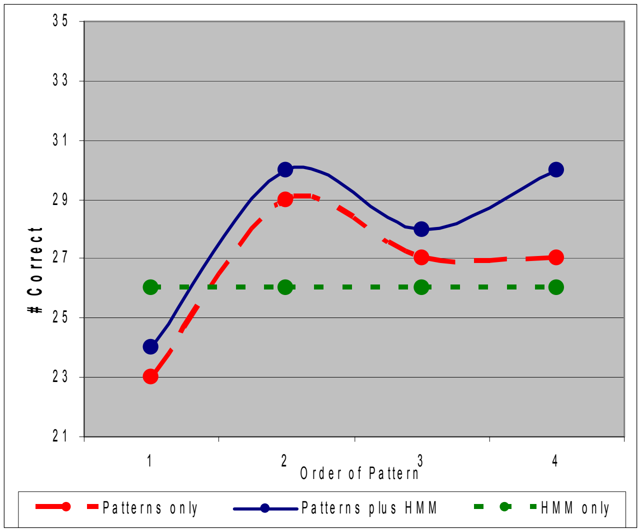

Figure 9. Comparison of Support Vector Machine (SVM) trained using 2nd order patterns against baseline Hidden Markov Model (HMM) and a hybrid model in which the output of the HMM is given as a feature to the SVM using pattern features.

6. CONCLUSION

We have presented a methodology for automatically learning a lexicon of hierarchical patterns of movement from unannotated video data. These hierarchical patterns encode fine-grained temporal relations and capture global information about events.

In order to validate our methodology, we present a discriminative approach to event classification in which the lexicon of hierarchical patterns is used to represent events for a tree kernel Support Vector Machine. Evaluations indicate that the patterns are informative and suggest that accurate event classification systems may be achieved by incorporating the local information encoded in dynamic models of event classification with the global information captured in automatically learned lexicons of hierarchical patterns of motion.

The utility of hierarchical patterns does not end with video event classification. Another area of our research examines how such patterns can be used to aid human search through video data. We are examining the effectiveness of incorporating the multi-variate time series representations into a user interface in order to provide users with a static view of large amounts of dynamic video data. By coloring these time series with mined patterns (as in Figure 6), users will be able to more easily hone in on significant periods of movement, facilitating users' ability to search through large amounts of video data.

In addition to data visualization, the methods described here may also facilitate the learning of language to describe video events. Recent work in the cognitive sciences has stressed the importance of hierarchical representations of events in models of situated word learning [4]. The approach described here provides a way to learn such hierarchical structures automatically, allowing for the event representations learned from the HSP video data to be mapped to the speech classified in the HSP audio data. The development of such models would advance work on the cognitive modeling of human language development as well as research on Natural Language Interfaces for searching video data.

7. ACKNOWLEDGMENTS

The authors would like to acknowledge Nikolas Mavridis for the idea to split the motion detection into regions of interest. We would also like to thank Rony Kubat for his help with the user interface to the video data used in the project. This work was partly funded by a National Defense Science and Engineering Grant.

8. REFERENCES

[1] Allen, J.F. A General Model of Action and Time. Artificial Intelligence 23, 2, July 1984.

[2] Cohen, Paul R., 2001. Fluent Learning: Elucidating the Structure of Episodes. In Proceedings of the Fourth Symposium on Intelligent Data Analysis, London, England.

[3] Collins, Michael and Duffy, Nigel. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In ACL02, 2002.

[4] Fleischman, M. B. and Roy, D. Why Verbs are Harder to Learn than Nouns: Initial Insights from a Computational Model of Intention Recognition in Situated Word Learning. 27th Annual Meeting of the Cognitive Science Society, Stresa, Italy. July 2005.

[5] Fern, A., Givan R., Siskind, J.: Specific-to-General Learning for Temporal Events with Application to Learning Event Definitions from Video. J. Artif. Intell. Res. (JAIR) 17: 379-449 (2002)

[6] Hongeng, S. and R. Nevatia, Multi-Agent Event Recognition, IEEE ICCV, pp. II: 84-91, July 2001

[7] Höppner, Frank: Discovery of Temporal Patterns. Learning Rules about the Qualitative Behaviour of Time Series. PKDD 2001: 192-203

[8] Intille, S. and A.F. Bobick, “Recognizing planned, multi-person action”, Computer Vision and Image Understanding 81, 414–445 (2001)

[9] IvanovY. , Bobick, A.: Recognition of Visual Activities and Interactions by Stochastic Parsing. IEEE Trans. Pattern Anal. Mach. Intell. 22(8): 852-872 (2000)

[10] Klein, D. and Manning, C. Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency. *Proceedings of the 42nd Annual Meeting of the ACL,* 2004.

[11] Manzanera, Antoine, Richefeu, J. C.: A Robust and Computationally Efficient Motion Detection Algorithm Based on Sigma-Delta Background Estimation. ICVGIP 2004: 46-51

[12] Moschitti, Alessandro. A study on Convolution Kernels for Shallow Semantic Parsing. In proceedings of the 42-th Conference on Association for Computational Linguistic (ACL-2004), Barcelona, Spain, 2004.

[13] Roy, D., Patel, R., DeCamp, P., Kubat, R., Fleischman, M., Roy, B., Mavridis, N., Tellex, S., Salata, A., Guinness, J., Levit, M., Gorniak, P. The Human Speechome Project. 28th Annual Meeting of the Cognitive Science Society, Vancouver, Canada. July 2006.