Learning Effects of Robot Actions using Temporal Associations

Paul R. Cohen, Charles Sutton, and Brendan Burns
Computer Science Building
University of Massachusetts
Amherst, MA 01003
{cohen,casutton,bburns}@cs.umass.edu

Abstract

Agents need to know the effects of their actions. Strong associations between actions and effects can be found by counting how often they co-occur. We present an algorithm that learns temporal patterns expressed as fluents*, propositions with temporal extent. The fluent-learning algorithm is hierarchical and unsupervised. It works by maintaining co-occurrence statistics on pairs of fluents. In experiments on a mobile robot, the fluent-learning algorithm found temporal associations that correspond to effects of the robot's actions.*

1. Introduction

Temporal associations are useful things. Embedded agents are particularly concerned with associations among their activities: for example, when one moves forward, things get closer; when a robot closes its gripper, it may find itself holding an object. Learning such associations without supervision is especially desirable, so humans do not need to decide which associations are important, and so agents can respond to changes in the environment simply by learning different patterns.

We express temporal patterns using *fluents* [7]. A fluent is a proposition with temporal extent. For example, "drinking-coffee" can be defined as a fluent that is true whenever I am drinking coffee. This fluent can be represented as a binary time series $\mathbf{x}$, where $\mathbf{x}_t$ is 1 if and only if I am drinking coffee at time t.

FLUENT-LEARNING [3] is an unsupervised algorithm that finds associations between fluents, that is, it finds patterns in multivariate time series of binary variables. We apply FLUENT-LEARNING to learn patterns in the observations and actions of a mobile robot. Many of the patterns correspond to effects of the robot's actions, for example, moving forward often leads to change in the front sonar.

The method is statistical: For a possible association between A and B, the algorithm maintains a contingency table that measures how much the presence of A affects the frequency of B. Since we use Allen's [1] interval calculus, the learned patterns are both hierarchical and propositional.

In the first section, we describe the Allen temporal relations and how we use contingency tables to judge whether a candidate pattern is statistically significant. Then, we describe the FLUENT-LEARNING algorithm that constructs these contingency tables. Third, we briefly describe how fluents correspond to operator models, that is, effects and initial conditions of actions. Fourth, we report fluents learned from the experiences of a mobile robot. Finally, we discuss related and future work.

2. Fluents and Temporal Relationships

Fluents have beginnings and ends. During the time in which a fluent holds, we say that it is *open*. Allen [1] gave a logic for relationships between the beginnings and ends of fluents. We use a nearly identical set of relationships:

SBEB(A,B)	A starts before B, ends before B; Allen's "overlap"
SWEB(A,B)	B starts with A, ends before A; Allen's "starts"
SAEW(A,B)	B starts after A, ends with A; Allen's "finishes"
SAEB(A,B)	B starts after A, ends before A; Allen's "during"
SWEW(A,B)	B starts with A, ends with A; Allen's "equal"
ES(A,B)	B starts after A ends; amalgamating Allen's "meets" and "before"

Although a proper subset of Allen's relations, this list is exhaustive: any two fluents are in exactly one of these relations.

A *composite fluent* is a temporal relationship between fluents, written $\rho(A, B)$, where A and B are fluents, and

ρ is one of the six relationships above. The relationship is itself a fluent that begins with its first constituent and ends with its second constituent.

For example, suppose the robot begins pushing an object and then stalls, diagrammed like this:

Three temporal relationships are displayed: SWEB(touch,push), SAEW(touch,stall) and ES(push, stall). But there are six other ways to describe this situation using composite fluents, e.g., SAEW(stall,SWEB(touch,push)) says, "the relationship between touch and push begins before and ends with their relationship with stall."

Let $\rho \in$[SBEB,SWEB,SAEW,SAEB,SWEW,ES], and let P be a proposition (e.g., MOVING-FORWARD). Composite fluents have the form:

$$F \leftarrow P \text{ or } CF$$
$$CF \leftarrow \rho(F, F)$$

That is, a fluent F may be a proposition or a temporal relationship between fluents. Propositions are also called *base fluents*, to distinguish them from composite fluents.

A *significant composite fluent* is a statistically significant temporal relationship between fluents. That is, we will accept ρ(A,B) as a pattern if and only if the constituent fluents are statistically associated, if they "go together." We can interpret a significant composite fluent ρ(A,B) as "When I see A, I tend to see a B such that ρ(A,B)."

For example, suppose we are considering the composite fluent ES(coffee,jitters), that is, jitters begin after the end of having coffee. Four frequencies are relevant:

	jitters	no-jitters
coffee	a	b
not-coffee	c	d

Here a is the number of "coffee" fluents that are followed by "jitters", b the number of "coffee" fluents that are not followed by "jitters", c the number of "not-coffee" fluents that are followed by "jitters", and d the number of "not-coffee" fluents that are not followed by "jitters." A "not-coffee" fluent is defined simply to be true at a time step if and only if "coffee" is false.

For the composite fluent to be significant, certainly a should be bigger than b, that is, I should get the jitters more often than not after drinking coffee. Suppose this is true, so $a = kb$ for some $k > 1$. If the relative frequency of jitters is no different after I drink, say, orange juice, or talk on the phone (e.g., if $c \approx kd$) then clearly there's no special relationship between coffee and jitters. Thus, to accept ES(coffee,jitters), I'd want $a = kb$ and $c = md$, and $k >> m$. In other words, whether we see coffee or not affects the distribution of when we see jitters.

To test the hypothesis that the start of the jitters fluent is independent of the end of the drinking coffee fluent, we use Cramer's statistic, which for a 2×2 contingency table is defined as

$$\phi_c = \sqrt{\frac{\chi^2}{N}},$$

where N is the table total. Cramer's statistic ranges between 0 and 1 and is interpreted like a correlation coefficient. If Cramer's statistic exceeds a predefined threshold ϕ_{crit}, we say that the fluent is significant.

This statistic also serves to test hypotheses about the other five temporal relationships between fluents. Consider a composite fluent like SBEB(brake,clutch): When I approach a stop light in my standard transmission car, I start to brake, then depress the clutch to stop the car stalling; later I release the brake to start accelerating, and then I release the clutch. To see whether this fluent—SBEB(brake,clutch)—is statistically significant, we use a contingency table to test whether "clutch" is more likely to start when "brake" is open than when any other fluent is open:

	clutch	non-clutch
brake	a	b
non-brake	c	d

where

- a is the number of times we see SBEB(brake,clutch),
- b is the number of times we see "brake" but not "clutch," i.e., SBEB(brake,B) for any B that is not "clutch"
- c is the number of times we see "clutch" but not "brake," i.e., SBEB(A,clutch) where A is not "brake"
- d is the number of times we see SBEB(A,B) where A is not "brake" and B is not "clutch"

Imagine some representative numbers here: Only rarely do I start something other than braking and then depress the clutch, so c is small. Only rarely do I start braking without depressing the clutch (otherwise the car would stall), so b is also small. Clearly, a is relatively large, and d bigger still, so the table has most of its frequencies on a diagonal, producing a significant statistic. This means that SBEB(brake,clutch) is a significant composite fluent.

There is a difference between the tables given for ES and SBEB: in the ES table, the rows are "coffee" and "not-coffee", while in the SBEB table, the rows are "brake" and "anything but brake". For example, suppose that fluents "morning news," "eating grapefruit," and "starting car" happen after I finish my coffee. These fluents count as three instances of "anything but coffee," for they are three fluents that are not coffee, but they are subsumed by only one

```
FLUENT-LEARNING(data)
  n ← table to record number of occurrences of each fluent,
        initially empty
  I ← table to record intervals during which each fluent
        is open, initially empty
  for t from 0 to length(data)
     fluents ← NEWLY-CLOSED-FLUENTS (t)
     for each f in fluents
        increment n_f
        push (start (f), end (f)) onto I_f
        if f is now significant
           add to list of significant fluents
  return list of significant fluents


NEWLY-CLOSED-FLUENTS(t)
  open ← list of base fluents that end at time t
  closed ← empty list
  for each fluent f in open
     for each fluent g that ends within the short-term memory
        ρ ← the temporal relation between f and g
        if ρ(f, g) is significant
           push ρ(f, g) onto open
        else push ρ(f, g) onto closed
     push f onto closed
  return closed
```

Figure 1. The FLUENT-LEARNING algorithm

"not-coffee" fluent, which persists until the next time I have coffee. The choice of which counting method to use is subtle, and we do not understand it fully. In the results here, ES and SAEB use the not-A counting method, while the other four relations use the anything-but method.

However we count, the hypothesis test allows us to find *rare patterns*: I might not drive my car very often, and so I rarely use the brake, but when I do, I always depress the clutch. Also, we avoid patterns that involve fluents that occur frequently in any context: for example, every time I open the refrigerator, I am still breathing.

We have described the fluent relations and how contingency tables are used to decide if fluents are significant. Next we describe how the contingency tables are constructed.

3. Fluent Learning Algorithm

The fluent learning algorithm incrementally constructs contingency tables for fluents. It processes a time series $\mathbf{x}$ of binary vectors. Each vector $\mathbf{x}_t$ has one boolean value for each base fluent f—for example, "moving-forward" or "gripper-closed"—that indicates whether f is true at time t.

The idea is to record the number of occurrences of every fluent that occurs in the input. Since there is an enormous number of possible patterns, we reduce the number of patterns ρ(A,B) that we consider in two ways.

First, we require that A and B occur within a *short-term memory*, that is, a sliding window over the time series which records when fluents start and end. A time step on which no fluent starts or ends does not occupy space in memory. For example, the fluent given in Section 2 requires a short-term memory of length 3: one time step when "touch" and "push" start, one when "push" ends and "stall" starts, and one when "stall" and "touch" end. The short-term memory is inspired by animal learning, because animals do not learn associations between events that occur far apart.

Second, we record information about a composite fluent ρ(A,B) only if A and B are either base fluents or are already significant. This heuristic assumes that significant fluents are likely to be important in the environment, and therefore fruitful for exploring further.

The fluent-learning algorithm is given formally in Figure 1. The algorithm records the intervals and counts of all fluents observed that meet the above two constraints. With this information, one can calculate the contingency tables described in the previous section. Note that when a base fluent ends, many composite fluents end as well; the procedure NEWLY-CLOSED-FLUENTS calculates them all by looking backward through the short-term memory.

FLUENT-LEARNING is unsupervised and multivariate. It is also incremental because new composite fluents become available for inclusion in other fluents as they become significant.

4. Fluents as Operator Models

Significant fluents can correspond to initial conditions and effects of an agent's actions. For example, a significant fluent ES(move-forward,F) could be interpreted as "The action move-forward tends to result in F."

Learning initial conditions is more difficult. If the initial condition is I, the action A, and the effect E, then a significant fluent $\rho_1(\rho_2$(I,A),E) might be interpreted to mean that in the presence of I, action A leads to E. But composite fluents can be significant when their constituents are not. In this example, if all actions may be attempted in all states, ρ_2(I,A) is unlikely to be significant, but the composite fluent $\rho_1(\rho_2$(I,A),E) is significant. Recall that FLUENT-LEARNING does not keep statistics on a composite fluent unless both of its constituents are already significant. Because of this, FLUENT-LEARNING will never learn the fluent $\rho_1(\rho_2$(I,A),E), for its constituent is not significant.

The problem is that planning operators such as $\rho_1(\rho_2$(I,A),E) may be significant when none of their pairwise constituents are; essentially, they are ternary relations between initial condition, action, and effect. This would not be a problem if FLUENT-LEARNING worked with three-way tables instead of bivariate relationships. One way to achieve the effect of learning three-way relations, without paying a prohibitive price in combinatorics, is to allow a fluent of the form ρ(B,A), where B is a base fluent and A an action, to participate in higher-level fluents. That is, in the **if** statement in NEWLY-CLOSED-FLUENTS (Figure 1), we would expand ρ(B,A) as if it were significant.

5. Experiments

We tested FLUENT-LEARNING on data generated using a Pioneer II robot in a walled playpen. A small styrofoam block was in front of the robot. The robot could perform four actions: move forward, move backward, open gripper and close gripper. At each time step its perceptual system output six propositions about the environment: four about its relative location measured using sonar—closer front, farther front, closer rear and farther rear—and two about the gripper—object in gripper, meaning that an object was between the gripper paddles, and holding object, meaning that the gripper was closed on an object.

In other experiments, we also included the negations of these base fluents, such as "not-holding-object," but in this experiment we did not, because composite fluents that contain negations tend to be harder for humans to interpret.

The data contained 2707 binary vectors of length 10: six elements recorded the sensory propositions, and four recorded whether each action had been initiated. The robot initiated actions on 604 of the time steps. Of the 1024 possible binary vectors, some combinations were impossible, for example moving forward and moving backward at the same time. In all, 36 unique states occurred in the data. We hypothesized that many significant fluents would correspond to the effects and initial conditions of the robot's actions.

At the confidence level $\phi_{crit} = 0.8$, the algorithm found 61 significant fluents. By hand, we classified these into three categories: good, marginal, and bad. Good fluents were those that in our judgment reflected dynamics of the environment, or the effects of robot actions, for example, ES(move-forward,closer-front), shown in Figure 2.

Some significant fluents indicate initial conditions of actions. For example, the fluent SAEB(SAEB(object-in-gripper,close-gripper),holding-object), displayed in Figure 3, could be interpreted, "During an object-in-gripper fluent that contains a close-gripper fluent, one tends to see a holding-object fluent." This pattern leaves the order of "close-gripper" and "holding-object" unspecified; we have diagrammed the relation in the order found in the data.

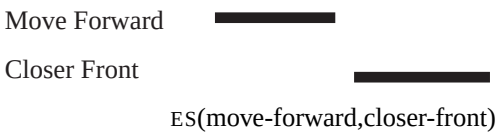

Figure 2. A significant fluent returned by the algorithm

Figure 3. A learned fluent that indicates an action's initial condition

The constituent fluent SAEB(object-in-gripper,close-gripper) was not significant; only about half the time that an object was between the gripper paddles did the robot happen to close the gripper.

Marginal fluents were generally those that combined two good fluents inappropriately. For example, consider the fluent in Figure 4. Certainly moving backward while an object is between the gripper paddles will lead to things being closer in the rear, but that is only because moving backward *always* leads to closer rear. Marginal fluents are thus not spurious; they reflect regularities in the environment, but they do not correspond to accurate operator models. An automatic pruning procedure could probably remove such a fluent, given that ES(move-backward,closer-rear) is already significant.

Bad fluents were those that seemed completely implausible. An example of a bad fluent the algorithm found is given in Figure 5. While the association between SAEB(object-in-gripper,move-backward) and closer-rear is significant, it does not correspond to any *causal* rule in the environment.

Figure 4. A marginal fluent returned by the algorithm

ES(SAEB(object-in-gripper,close-gripper),closer-rear)

Figure 5. A bad fluent returned by the algorithm

The relationship between the clauses is accidental.

Out of the 61 fluents the algorithm returned, we classified 15 as good, 25 marginal, and 21 bad. In other words, about two-thirds of the significant fluents reflect some regularity in the environment, but less than one-third correspond to accurate operator models. Of course, by raising the significance threshold ϕ_{crit}, we can reduce the number of marginal and bad fluents, at the expense of some good fluents. Apart from that, exploring different ways of counting fluents in the contingency tables, as discussed in the second section, may help to reduce the number of marginal and bad fluents. Because we use hypothesis testing, in general we expect that a small percentage of significant fluents are due simply to chance.

We have also run experiments to learn the effects of turning. In this case, the playpen contained two large trash cans that are detectable by sonar. The robot could perform two actions: turn left 45° and turn right 45°. Very few good fluents were found, mainly because sonar is difficult to correlate with turning. For example, after the robot turns left, anything could happen to the front sonar: a wall in front could become closer or farther, depending on the angle between wall and robot, which sonar cannot sense. We may attain better results using a vision system, tracking how objects move across the visual field, instead of sonar.

6. Related Work

The fluent-learning algorithm was first presented by Cohen[3]. There, the task was to partition robot experience into episodes; here, we focus on learning the outcomes of controllers. This application of fluent learning was inspired by previous work that used the MSDD algorithm to learn planning operators [8].

Some other recent work finds patterns expressed using Allen relations. One technique [5, 6] measures the strength of a pattern by its total duration, rather than by counts of open and close events. Associations between patterns are learned using rules whose antecedent and consequent are patterns. The consequent of a rule always temporally follows the antecedent.

Association rule learning, a technique from knowledge discovery, has been adapted to find temporal associations [12]. This algorithm works on a database of tuples whose elements are propositions with temporal extent. First associations are learned within the tuples by using standard knowledge discovery algorithms that ignore time, and then temporal relations are learned for each association.

In general, our approach differs from both of these in that, for a pattern ρ(A,B), we consider not only the frequency of B given A but also the background frequency of B without A. We would most likely treat a temporal rule A $\rightarrow$ B as a composite fluent ES(A,B).

Witkowski et al. [16] describe using temporal inference for robot control. Given a domain theory of an environment written in predicate logic and the event calculus, they use abductive reasoning for tasks like map-making and motion planning. The domain theory is not learned.

Fluent learning is based on the idea that random coincidences of events are rare, so the structure of a time series can be discovered by counting these coincidences. Thus, it accords with psychological literature on neonatal abilities to detect coincidences [15], and it has a strong statistical connection to causal induction algorithms [10], though we do not claim that the algorithm discovers causal patterns. In discovering patterns, it differs from techniques that elucidate only probabilistic structure, such as autoregressive models [4], HMMs [11, 2], and markov-chain methods such as MBCD [13]. Clustering by dynamics and time-warping also discover patterns [9, 14], but require the user to first identify episode boundaries in time series.

7. Conclusion

Fluent learning works for multivariate time series in which all the variables are binary. It does not attend to the durations of fluents, only the temporal relationships between open and close events. This is an advantage in domains where the same episode can take different amounts of time, and a disadvantage in domains where duration matters. The algorithm uses hypothesis testing; thus, we accept a fluent ρ(A,B) not because of the absolute number of times ρ(A,B) occurs, but because of how much A affects the observed distribution of B.

Because it is a statistical technique, fluent learning finds common patterns, not all patterns; it is easily biased to find more or fewer patterns by adjusting the threshold value of the statistic and varying the size of the short-term memory.

We applied the algorithm to data from a mobile robot to find effects and initial conditions of its actions. In future work, we hope to use these as planning operators for robot control. Also, it would be intriguing for an algorithm to construct experiments to determine whether a significant fluent in fact reflects a causal relation. FLUENT-LEARNING

does not learn causal models—causal induction is computationally much more challenging, and our approach to learning causal rules is not to induce what we can from observational data but rather to use fluents as hypotheses about causal rules to be learned by the robot in true experiments.

References

[1] J. F. Allen. An interval based representation of temporal knowledge. In *IJCAI81*, pages 221–226, San Mateo, CA, 1981. Morgan Kaufmann Publishers, Inc.

[2] E. Charniak. *Statistical Language Learning*. MIT Press, 1993.

[3] P. Cohen. Fluent learning: Elucidating the structure of episodes. In *Proceedings of Fourth Symposium on Intelligent Data Analysis*, pages 268–277. Springer, 2001.

[4] J. D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

[5] F. Höppner. Discovery of temporal patterns – learning rules about the qualitative behaviour of time series. In *Proc. of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Springer, 2001.

[6] F. Höppner and F. Klawonn. Finding informative rules in interval sequences. In *Proceedings of Fourth Symposium on Intelligent Data Analysis*. Springer, 2001.

[7] J. McCarthy. Situations, actions and causal laws. Stanford Artificial Intelligence Project: Memo 2, also, http://wwwformal.stanford.edu/jmc/mcchay69/mcchay69.htm, 1963.

[8] T. Oates and P. R. Cohen. Learning planning operators with conditional and probabilistic effects. In *Proceedings of the AAAI Spring Symposium on Planning with Incomplete Information for Robot Problems*, pages 86–94, 1996.

[9] T. Oates, M. D. Schmill, and P. R. Cohen. A method for clustering the experiences of a mobile robot that accords with human judgements. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 846–851. AAAI Press/The MIT Press, 2000.

[10] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

[11] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.

[12] C. P. Rainsford and J. F. Roddick. Adding temporal semantics to association rules. In *Proc. of the 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 504–509. Springer, 1999.

[13] 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.

[14] D. Sankoff and J. B. Kruskal, editors. *Time Warps, String Edits, and Macromolecules: Theory and Practice of Sequence Comparisons*. Addison-Wesley Publishing Company, Reading, MA, 1983.

[15] E. S. Spelke. The development of intermodal perception. In P. Salapatek and L. Cohen, editors, *The Handbook of Infant Perception*. Academic Press, 1987.

[16] M. Witkowski, D. Randell, and M. Shanahan. Deriving fluents from sensor data from mobile robots. In *Papers from the 2001 AAAI Fall Symposium Series*, pages 44–51, 2001.