11 Learning and Causal / Temporal Reasoning

The passive learning model introduced in Chapter 4 was designed to enable us to find formal descriptions of properties. At first blush, we might also wish to use it to *predict* some aspect of future states, if we use data that is labeled with such outcomes for past observations. In particular, we may wish to produce a model of the world that an agent can use to guide its behavior. Crucially, we suppose here that the actions of that agent can influence the future states of the world. In this chapter, in Section 11.1, we describe a simple framework for capturing the causal and temporal structure of such environments that change over time, and in Section 11.2, we consider a basic version of the problem of learning how the environment changes over time. Subequently, we discuss some important issues that arise with such learning problems. First, in Section 11.3, we find that when our observations of changes in the environment are insufficiently fine-grained, the learning problem often becomes intractable. Second, in Section 11.4, we consider situations in which *acting* on these predictions influences the future states of the environment. In this *causal* learning problem, that we call *introspective forecasting*, we generally no longer are trying to predict future observations that are distributed identically to our training data, invalidating the fundamental assumption of the PAC learning model. In this case, we need to extend the problem, to anticipate the impact of the prediction on the future states of the environment. We show how algorithms for the passive PAC learning model can be adapted to this setting, and discuss the inherent limitations such algorithms face.

11.1 Reasoning about actions and change.

Our exposition so far has mostly focused on static settings where structure exists within each isolated state of the world. Yet, a substantial portion of the knowledge agents need in real-life settings is causal in nature: it specifies *when* certain conditions bring about particular changes and *how* these changes propagate through the rest of the world. In turn, this necessitates the development of frameworks for reasoning about action and change, able to encode this causal knowledge to support prediction, explanation, and planning. In contrast to the static settings considered earlier, temporal reasoning treats the world as a sequence of states linked by causal dependencies; actions may produce direct and indirect effects, while many properties simply persist unless there is a reason

to change. While we have briefly touched upon a central aspect of temporal reasoning, namely the frame problem, in Chapter 10, here we offer a fuller formalization.

We introduced temporal domains as the setting of classical planning problems in Section 5.6, and we recall the key terminology here. A *state* is a binary vector $\vec{x} \in \{0,1\}^n$ over a fixed list of propositional attributes (also called *fluents*) $X_1, \ldots, X_n$, with $\vec{x}_i$ denoting the truth value of fluent X_i in $\vec{x}$. Fluents represent properties of the environment that an agent may sense and that may change over time as actions are executed. In the temporal setting, a *world* is not a single state but a sequence of states; for concreteness, we define a world to be a mapping $w : \{0,1,\ldots\} \to \{0,1\}^n$, where $w(t)$ is the state at (discrete) time t.

Analogously to how individual states can be constrained to reflect structure that exists between the fluents of a state, the sequence of states in a temporal world can similarly be constrained to reflect structure that exists between the fluents across states.

We encode these temporal constraints as follows.

Definition 11.1 (Causal implications and domains) A *causal implication* (of order k) is a statement of the form "$S \rightsquigarrow L$", where S is a set of (k) literals read as a conjunction, and L is a literal. A causal implication is *monotone* if no literal in $S \cup \{L\}$ is negated. A *domain* c is a finite set of causal implications. A literal L is *triggered* in a domain c by a state $\vec{x}$ if there is a causal implication "$S \rightsquigarrow L$" $\in c$ such that $\vec{x}$ satisfies S.

Recalling the compact frame axiom from Chapter 10, one could introduce other types of constraints in a domain, including static constraints, defeaters of causal implications, or priority relations between conflicting causal implications. We opt not to do so, both to avoid the reintroduction of the full conflict-resolution mechanisms of argumentation and to maintain our focus on the nuances of the temporal setting. Accordingly, we do not explicitly represent the frame axiom in domains, but instead let the reasoning semantics address the frame problem.

Definition 11.2 (Worlds obeying causal implications) A world w *obeys* a domain c if, for every literal L and time $t \geq 0$, $w(t+1)$ satisfies L if and only if either (i) L is triggered in c by $w(t)$, or (ii) $w(t)$ satisfies L and $\neg L$ is not triggered in c by $w(t)$.

We silently assume that the domains and worlds under consideration in this chapter never give rise to a state in which both a literal L and its complement $\neg L$ are triggered.

Example 11.1: [Faulty Wiring] Consider a scenario described in terms of current flowing (C), the wiring being okay (W), and the fuse working (F). Let c be a domain with the causal implications "$\{C, \neg W\} \rightsquigarrow \neg F$" and "$\{\neg F\} \rightsquigarrow \neg C$".
An initial state $w(0)$ that satisfies $\{C, \neg W, F\}$ triggers $\neg F$, resulting in a state $w(1)$ that satisfies $\{C, \neg W, \neg F\}$. This new state triggers $\overline{C}$ (and $\neg F$), resulting in a state $w(2)$ that satisfies $\{\neg C, \neg W, \neg F\}$. By persistence, all other states $w(t)$, $t > 2$ are equal to $w(2)$. The world w obeys c.

Although causal implications describe changes associated with action executions, actions are left implicit in the formalism. In the Faulty Wiring example, the initial state

$w(0)$ may be viewed as the direct outcome of flipping a switch in a pre-initial state satisfying $\{\neg C, \neg W, F\}$, thereby causing current to begin flowing. More generally, $w(0)$ incorporates the direct effects of one or more concurrently executed actions, while the domain c governs the subsequent evolution of the world. The transitions from $w(0)$ onwards therefore represent the indirect effects, or ramifications, arising from those direct effects. When these post-action dynamics depend on the set A of concurrently executed actions, each such set may be associated with a distinct domain c_A.

The Faulty Wiring example also illustrates how causal implications capture cascading ramifications of actions across multiple time-points. The initial state triggers the failure of the fuse, so $\neg F$ holds in $w(1)$. This newly established condition triggers the cessation of the current, so $\neg C$ holds in $w(2)$. Together with the implicit action described above, the scenario represents a causal chain in which flipping the switch directly, but only temporarily, causes current to flow, the resulting current through faulty wiring causes the fuse to fail, and the failed fuse subsequently causes the current to stop.

More generally, domains can encode computations that unfold over time.

Example 11.2: [Computations as Rules] Consider a feed-forward monotone Boolean circuit whose input values are represented by certain fluents in $w(0)$. Additional fluents may represent gate outputs, with causal implications encoding the operation of each gate: an AND gate with input fluents X_1, X_2 and output fluent Y may be represented by "$\{X_1, X_2\} \rightsquigarrow Y$", while an OR gate may be represented by "$\{X_1\} \rightsquigarrow Y$" and "$\{X_2\} \rightsquigarrow Y$", assuming that Y is initially false. Arranging gates in layers allows the values computed in one state to trigger the computation of the next layer in the following state.

Thus, a temporal world may represent not only a physical chain of ramifications, but also the successive stages of a symbolic computation.

11.2 Learning causes in the presence of inertia.

The discussion so far has assumed that the domain c is known, allowing the temporal evolution of a world to be determined from a given initial state. We now fix an unknown target domain c and consider whether its induced evolution can instead be predicted from experience. The learner's goal is not necessarily to recover the exact causal implications in c, but to identify a hypothesis h that predicts the successor of a previously unseen initial state. If h is required to be a domain from the same representation class as c, the learning task is proper; otherwise, it is improper.

As in the static learning setting of earlier chapters, the learner receives independently drawn examples according to some fixed but otherwise arbitrary distribution D over initial states. Here, each example is a pair of consecutive states $\langle w(0), w(1)\rangle$ from

a world that obeys c, where $w(0)$ is drawn according to D. During training, both states are available, whereas during testing the learner is given only $w(0)$ and predicts $w(1)$.

Definition 11.3 Let $\mathcal{C}_n$ and $\mathcal{H}_n$ be classes of representations of domains on the same n fluents. We say that $\mathcal{C}_n$ is *PAC-learnable from consecutive states* using $\mathcal{H}_n$ in time $t(n, \frac{1}{\epsilon}, \frac{1}{\delta})$ and sample complexity $m(n, \frac{1}{\epsilon}, \frac{1}{\delta})$ if there is an algorithm such that for any world w obeying some $c \in \mathcal{C}_n$, when m examples of the form $\langle w(0), w(1)\rangle$ where each $w(0)$ is drawn independently according to D, given these examples and the parameters ϵ and δ, with probability at least $1 - \delta$ the algorithm outputs a representation $h \in \mathcal{H}_n$ such that

$$\Pr_{w(0)\in D}[h(w(0)) = w(1))] \geq 1 - \epsilon.$$

If $\mathcal{C}_n = \mathcal{H}_n$ we say the learner is *proper*, and otherwise we say it is *improper*. If $\mathcal{C}_n$ is PAC-learnable from consecutive states in polynomial time and sample complexity by any class $\mathcal{H}_n$ for which evaluation can be computed in polynomial time, then we simply say that $\mathcal{C}_n$ is *efficiently PAC-learnable from consecutive states*.

Partial observability, as studied in earlier chapters, extends directly to the temporal setting considered here, by allowing either state in an observed pair to be masked. Since this introduces no complication specific to temporality, we do not develop it further.

It is easy to see that the temporal learning setting is reducible to the static learning setting of earlier chapters. Indeed, for each fluent X_i, let $f_i^c : \{0,1\}^n \to \{0,1\}$ map an initial state $w(0)$ to the value of X_i in its successor state $w(1)$ for a world w that obeys c. Learning to predict the successor state can therefore be reduced to learning the n Boolean functions $f_1^c, \ldots, f_n^c$, using the initial state as input and the corresponding successor-state fluent as the label. In particular, if each coordinate function belongs to a PAC-learnable class, then the complete successor state can be learned by applying the corresponding learning algorithm separately to each coordinate, with appropriately chosen error and confidence parameters. We leave the formal reduction as an exercise (Exercise 11.3).

The coordinate functions reflect both causal change and inertia. For each fluent X_i, let $P_i^c(\vec{x})$ indicate that X_i is triggered (via some causal implication) in c by $\vec{x}$, and let $N_i^c(\vec{x})$ indicate that $\neg X_i$ is triggered (via any causal implication) in c by $\vec{x}$. It follows from the semantics that $f_i^c(\vec{x}) = P_i^c(\vec{x}) \vee \big(x_i \wedge \neg N_i^c(\vec{x})\big)$. The first disjunct captures a positive value brought about by a causal implication, while the second captures persistence of a fluent's truth unless its complement is triggered. When $N_i^c(\vec{x})$ holds, and hence $P_i^c(\vec{x})$ does not hold under our consistency assumption, both disjuncts are false and the fluent becomes false in the successor state.

The coordinate-wise reduction establishes learnability whenever the induced successor functions belong to learnable classes, but it does not generally produce a domain hypothesis. We next give a direct learning result that preserves the causal representation and hence establishes proper learnability for a particular class of domains: For an integer $k \geq 0$, let $\mathcal{C}_k$ denote the class of domains over $X_1, \ldots, X_n$ in which every causal implication has order at most k. We refer to $\mathcal{C}_k$ as the class of *order-k domains*.

Theorem 11.4 (Learning order-k domains) *For every constant k, the class C_k is properly PAC-learnable from consecutive-state examples.*

Proof Let $\mathcal{L}_k$ be the set of all causal implications "$S \rightsquigarrow L$" over $X_1, \ldots, X_n$ whose order is at most k. Its size is upper-bounded by

$$|\mathcal{L}_k| \leq 2n \sum_{j=0}^{k} \binom{2n}{j} = n^{O(k+1)}.$$

Thus, $\mathcal{L}_k$ can be enumerated in polynomial time when k is constant.

Input: Training sample $T = \{\langle w_r(0), w_r(1)\rangle\}_{r=1}^{m}$
Output: A domain h.
begin
  Initialize h to contain all of $\mathcal{L}_k$.
  for $r = 1, \ldots, m$ **do** **foreach** *literal L false in $w_r(1)$* **do**
    forall "$S \rightsquigarrow L$" $\in h$ **do**
      if *S satisfied in $w_r(0)$* **then** Remove "$S \rightsquigarrow L$" from h.
    end
  end

  return h.
end

Algorithm 24: Elimination Algorithm for learning order-k domains

Given a training sample

$$T = \{\langle w_r(0), w_r(1)\rangle\}_{r=1}^{m},$$

where every w_r obeys the target domain $c \in C_k$, consider the variant of the Elimination Algorithm given in Algorithm 24

No implication belonging to c is removed. Indeed, suppose that "$S \rightsquigarrow L$" $\in c$ and that $w_r(0)$ satisfies S. Then L is triggered in c by $w_r(0)$, so the semantics of the target domain implies that $w_r(1)$ satisfies L. The condition for removing the implication therefore never holds. Consequently, $c \subseteq h$.

We show that h predicts every training successor exactly. Fix a pair $\langle w(0), w(1)\rangle \in T$. First observe that no literal that is false in $w(1)$ is triggered in h by $w(0)$. Otherwise, some surviving implication "$S \rightsquigarrow L$" $\in h$ would have its antecedent satisfied by $w(0)$ while its conclusion L is not satisfied by $w(1)$, contradicting the elimination rule. In particular, h cannot trigger both a literal and its complement on this training state.

Now let L be any literal satisfied by $w(1)$. If $w(0)$ does not satisfy L, then the change to L cannot be due to inertia. Hence, by the semantics of the target domain, L is triggered in c by $w(0)$. Since $c \subseteq h$, it is also triggered in h. If $w(0)$ already satisfies L, then $\overline{L}$ is false in $w(1)$ and therefore, by the preceding observation, is not triggered in h by $w(0)$. Thus, L persists under the semantics of h.

It follows that every literal satisfied by $w(1)$ is also satisfied by the successor predicted by h. Since states assign exactly one truth value to every fluent, and no complementary literals are jointly triggered on this training state, the predicted successor is exactly $w(1)$. Hence, h is consistent with every example in T.

Every possible output of the algorithm is a subset of $\mathcal{L}_k$, so there are at most $2^{|\mathcal{L}_k|}$ possible hypotheses. To accommodate the possibility that a learned domain may trigger both a literal and its complement on an unseen state, we count such a conflict as an incorrect successor prediction. Thus, we say that the *loss* of h on an initial state is 1 if h either triggers complementary literals or predicts a successor different from that induced by c, and is 0 otherwise. The target domain c has zero loss on all worlds w that obey it, and the preceding argument shows that the algorithm returns a hypothesis with zero loss on T.

The Occam's Razor bound (Theorem 4.2) therefore implies that

$$m \geq \frac{1}{\epsilon}\left(|\mathcal{L}_k| \ln 2 + \ln \frac{1}{\delta}\right)$$

examples suffice to ensure, with probability at least $1 - \delta$, that the probability under D that h makes an incorrect successor prediction is at most ϵ.

Finally, enumerating the implications in $\mathcal{L}_k$ and testing each of them against the sample takes polynomial time for constant k. The returned hypothesis is itself an order-k domain, and hence the learner is proper. □

The restriction to constant order in Theorem 11.4 is substantive. Even restricting attention to monotone causal implications does not make domains of unrestricted order efficiently learnable, unless polynomial-size DNF formulas are efficiently learnable.

Proposition 11.5 (Hardness of monotone domains) *If monotone domains of unrestricted order are efficiently PAC-learnable from consecutive-state examples, then polynomial-size DNF formulas are efficiently PAC-learnable.*

We leave the proof of Proposition 11.5 as an exercise (Exercise 11.4).

11.3 The effect of hidden temporal states.

Orthogonally to the masking of fluent values within states, temporal settings admit a second form of incomplete information: some of the intermediate states in a world may remain unobserved. Instead of observing consecutive states $\langle w(0), w(1)\rangle$, the learner may observe $w(0)$ and a later state $w(\tau)$, for some $\tau > 1$, possibly because an action's ramifications unfold more quickly than the agent can sense the changing environment.

We model this setting by fixing a delay $\tau > 1$ and providing the learner with independently drawn pairs $\langle w(0), w(\tau)\rangle$, where $w(0)$ is drawn according to a distribution D and w obeys the unknown target domain c. Only the two endpoints are revealed; the intermediate states $w(1), \ldots, w(\tau - 1)$ remain hidden. For each fluent X_i, let $f_i^{c,\tau} : \{0,1\}^n \to \{0,1\}$ map the initial state $w(0)$ to the value of X_i in $w(\tau)$. Unlike

the one-step function f_i^c, the function $f_i^{c,\tau}$ reflects the composition of τ successive transitions, and may depend on causal chains mediated by several unobserved states.

In the Faulty Wiring example (Example 11.1), an agent might observe the state $w(0)$ in which current is flowing and then observe $w(2)$, in which the fuse has failed and the current has stopped. The intermediate state $w(1)$, in which the fuse has failed but current continues to (momentarily) flow, remains hidden. The observed change from $w(0)$ to $w(2)$ consequently reflects the composition of several successive transitions induced by the domain. This composition can increase the representation complexity, and the computational difficulty of learning, even if every single-step transition is simple.

The circuit encoding from Section 11.1 illustrates the source of this added complexity. A layered circuit can be evaluated across several successive states, with the fluents computed at one time-point serving as inputs to gates evaluated at the next. If only $w(0)$ and $w(\tau)$ are observed, the values of the intermediate gate fluents remain hidden. The learner therefore observes only the input–output behavior of the entire circuit, rather than the simpler local computations represented by its individual layers.

The preceding observation can be formalized even for order-two domains. The construction is similar to Theorem 3.29, but produces causal implications rather than their clausal encodings, and uses two distinct fluents to represent the two possible values of each circuit wire. These fluents should not be confused with complementary literals.

Theorem 11.6 (Circuit simulation by order-two domains) *Let G be a Boolean circuit of size s and depth d, over* AND, OR, *and* NOT *gates of fan-in at most two. There exist, constructible in time polynomial in s,*

1. *an order-two domain $c_G \in C_2$ over $O(s)$ fluents, containing $O(s)$ causal implications, all but one of which are monotone;*
2. *an encoding ι of circuit inputs as initial states; and*
3. *an encoding o of circuit outputs as stable states,*

such that, for every input $\vec{x}$ to G, the world w that obeys c_G and satisfies $w(0) = \iota(\vec{x})$ has $w(t) = o\big(G(\vec{x})\big)$, for every $t \geq d + 1$. In particular, the final state depends only on the output of G and not on the values of its inputs or its internal gates.

Proof For every wire Z of G (including input, internal, and output wires), introduce two distinct fluents Z^+ and Z^- to represent, respectively, that the value carried by Z is 1 or 0. Introduce also a fluent Y that will record the output of the circuit.

For an input $\vec{x}$, define $\iota(\vec{x})$ so that, for each input wire X_i, exactly one of X_i^+ and X_i^- is true: X_i^+ when X_i carries the value 1, and X_i^- when X_i carries the value 0. All fluents associated with non-input wires, as well as Y, are false in $\iota(\vec{x})$.

We next encode the gates of G. For an AND gate $Z = A \wedge B$, include in c_G

$$\text{“}\{A^+, B^+\} \rightsquigarrow Z^+\text{”}, \qquad \text{“}\{A^-\} \rightsquigarrow Z^-\text{”}, \qquad \text{“}\{B^-\} \rightsquigarrow Z^-\text{”}.$$

For an OR gate $Z = A \vee B$, include in c_G

$$\text{“}\{A^+\} \rightsquigarrow Z^+\text{”}, \qquad \text{“}\{B^+\} \rightsquigarrow Z^+\text{”}, \qquad \text{“}\{A^-, B^-\} \rightsquigarrow Z^-\text{”}.$$

For a NOT gate $Z = \neg A$, include in c_G

$$\text{“}\{A^+\} \rightsquigarrow Z^-\text{”}, \qquad \text{“}\{A^-\} \rightsquigarrow Z^+\text{”}.$$

Let O be the output wire of G. To record a positive circuit output, include in c_G

$$\text{“}\{\neg O^-, O^+\} \rightsquigarrow Y\text{”}.$$

Finally, once either value of the output wire has been computed, we force every wire fluent to become true. For every wire fluent $R \in \{Z^+, Z^- : Z \text{ is a wire of } G\}$, include

$$\text{“}\{O^+\} \rightsquigarrow R\text{”} \qquad \text{and} \qquad \text{“}\{O^-\} \rightsquigarrow R\text{”}.$$

Consider the evolution from $\iota(\vec{x})$ before either output rail has become true. During this period none of the cleanup implications is triggered. It follows by induction that for every wire Z at circuit depth j, the fluent representing the correct value of Z becomes (and remains) true by time j, while the fluent representing the opposite value remains false. Consequently, unless cleanup has already occurred, exactly one of O^+ and O^- becomes true no later than time d, and it represents $G(\vec{x})$. In the following transition, the cleanup implications make every wire fluent true.

If $G(\vec{x}) = 1$, then immediately before cleanup O^+ is true and O^- is false. Hence, $\text{“}\{\neg O^-, O^+\} \rightsquigarrow Y\text{”}$ is triggered, and Y becomes true. If $G(\vec{x}) = 0$, then O^- is true, so this implication is not triggered, and Y remains false by inertia. Thus, by time $d+1$ the world has reached the state $o(G(\vec{x}))$, in which every wire fluent is true and $Y = G(\vec{x})$. It can be further verified that this state remains stable at all subsequent time-points. □

The implication of Theorem 11.6 is immediate. Whereas order-two domains are properly PAC-learnable when consecutive states are observed, the same domains can encode arbitrary Boolean circuit computations when their intermediate states are hidden. Thus, recalling Corollary 4.18, if we suppose that secure (public-key) cryptography is possible,[1] there is no efficient PAC learning algorithm for Boolean circuits. Then, in turn, hidden intermediate temporal states render the τ-step endpoint prediction task (the natural generalization of Definition 11.3 from $\tau = 1$ to some polynomially-bounded $\tau > 1$) computationally intractable.

Corollary 11.7 (Hardness from hidden temporal states) *If endpoint prediction from observations $\langle w(0), w(\tau)\rangle$ with hidden intermediate states is efficiently PAC-learnable for polynomial-size order-two domains and polynomially bounded τ, then polynomial-size Boolean circuits are efficiently PAC-learnable and hence secure public-key cryptography is impossible.*

11.4 Temporal predictions that lead to actions.

The learning settings considered, both in this and preceding chapters, treat prediction as external to the temporal evolution of the world (or more generally to the data labeling

[1] Actually, even secret-key encryption suffices, see the notes for Chapter 4.

process, in case no temporal dimension exists): a hypothesis predicts a future state or some feature of it, but making the prediction does not itself affect what subsequently happens. In many settings, however, especially where the future state is temporally distant, a prediction is announced or acted upon before its accuracy can be determined. Clearly, in the case of an agent using a learned model of the world to guide its behavior and avoid danger, voiding a prediction of an undesirable outcome is the whole point. But the same holds more generally even if we are simply casting the problem as predictive modeling. For example, a prediction of illness may lead to preventative treatment, while a prediction of scarcity may lead to stockpiling. The resulting actions may alter the very outcome being predicted. Predictions thus situated within the world then become part of the causal process, and their correctness must be evaluated with respect to the evolution that the prediction itself contributes to bringing about.

Example 11.3: [A self-defeating medical prediction] A doctor observes a patient in an initial state $w(0)$ and predicts whether the patient's eventual stable condition will be ill or well. Hospital policy prescribes treatment if and only if the doctor predicts ill. Thus, the same observed state $w(0)$ may give rise to different actions and subsequent evolutions, depending on the prediction that the doctor announces.
Let P, I, and T indicate, respectively, that an underlying disease process is active, that the patient is ill, and that treatment is being administered. Suppose that $w(0)$ satisfies $\{P, \neg I, \neg T\}$. The patient is not yet ill, but the active disease process will eventually cause illness unless treatment interrupts it. Let the domain be

$$c^* = \{\text{“}\{P\} \rightsquigarrow I\text{”}, \text{“}\{T\} \rightsquigarrow \neg P\text{”}, \text{“}\{\neg P\} \rightsquigarrow \neg I\text{”}\}\,.$$

If the doctor predicts ill, hospital policy administers treatment. The resulting evolution is governed by the revised domain $c_{\mathsf{ill}} = c^* \cup \{\text{“}\emptyset \rightsquigarrow T\text{”}\}$. Here, the additional causal implication represents the action taken in response to the announced prediction; treatment is not triggered merely by observing P in $w(0)$. The disease initially causes illness, but treatment subsequently stops the disease process and the patient eventually becomes well. The prediction ill is therefore false.
If the doctor instead predicts well, no treatment is administered, and the evolution is governed by domain $c_{\mathsf{well}} = c^*$. The disease process then causes illness and remains active, so the patient eventually becomes ill. The prediction well is also false.
Thus, neither prediction is correct: predicting illness causes its prevention, whereas predicting wellness causes treatment that would have prevented illness to be withheld.

The example can be generalized within the chapter's formalism by associating each possible prediction y with a domain c_y, reflecting that the post-prediction dynamics may depend on the prediction itself. These domains may consist of a common core of causal implications together with prediction-specific additions. We will not require this shared structure, however, and will instead allow the domains c_y to vary arbitrarily.

We formalize this setting as follows. Let $\mathcal{Y}$ be a known partition of the (future) state space $\{0,1\}^n$: its members are pairwise disjoint and their union is $\{0,1\}^n$. Each $y \in \mathcal{Y}$

represents a possible prediction, that the relevant future state will belong to y. Let C be a class of domains, and let $\mathbf{c} = (c_y)_{y \in \mathcal{Y}} \in C^{\mathcal{Y}}$ be the unknown target domain family, where c_y governs the evolution that follows when prediction y is announced.

A learner in this setting still receives independently drawn initial states according to some fixed but arbitrary distribution D. For an initial state $\vec{x}$ and an announced prediction $y \in \mathcal{Y}$, let $w_{\vec{x},y}$ denote the world that obeys c_y and satisfies $w_{\vec{x},y}(0) = \vec{x}$. A training observation therefore has the form $\langle \vec{x}, y, w_{\vec{x},y}(\tau) \rangle$, where $\vec{x}$ is drawn according to D, the prediction y is selected by the learner, and the resulting state at the prediction horizon is observed. Thus, unlike ordinary supervised learning, the learner partly determines which outcome is observed by choosing which prediction to announce.

During training, the learner may explore the members of $\mathcal{Y}$ independently of the forecasting rule that it will eventually return. At the end of training, the learner seeks to construct a hypothesis $h : \{0,1\}^n \to \mathcal{Y}$ such that on a previously unseen initial state $\vec{x}$, the learner announces $h(\vec{x})$, after which the world evolves according to $c_{h(\vec{x})}$. The prediction is correct precisely when $w_{\vec{x},h(\vec{x})}(\tau) \in h(\vec{x})$. Accordingly, define

$$\mathrm{err}_{D,\mathbf{c}}(h) \stackrel{\text{def}}{=} \Pr_{\vec{x} \in D} \left[w_{\vec{x},h(\vec{x})}(\tau) \notin h(\vec{x}) \right].$$

Because the resulting outcome depends on the prediction, the setting need not be realizable: for some initial states $\vec{x}$, there may be no prediction $y \in \mathcal{Y}$ such that $w_{\vec{x},y}(\tau) \in y$, as illustrated in Example 11.3. Define the response function $r_{\vec{x}} : \mathcal{Y} \to \mathcal{Y}$ by letting $r_{\vec{x}}(y)$ be the unique member of $\mathcal{Y}$ containing $w_{\vec{x},y}(\tau)$. A prediction y is correct on $\vec{x}$ precisely when it is a fixed point of $r_{\vec{x}}$. Let $\alpha_{D,\mathbf{c}} \stackrel{\text{def}}{=} \Pr_{\vec{x} \in D}[r_{\vec{x}} \text{ has no fixed point}]$. This quantity $\alpha_{D,\mathbf{c}}$ is exactly the *unavoidable forecasting error* in the following sense.

Proposition 11.8 (Unavoidable forecasting error) *For every forecasting hypothesis $h : \{0,1\}^n \to \mathcal{Y}$, $\mathrm{err}_{D,\mathbf{c}}(h) \geq \alpha_{D,\mathbf{c}}$. Moreover, when hypotheses are unrestricted, this bound is attainable.*

Proof If $r_{\vec{x}}$ has no fixed point, then $r_{\vec{x}}(h(\vec{x})) \neq h(\vec{x})$ for every possible output of h. It follows that $w_{\vec{x},h(\vec{x})}(\tau) \notin h(\vec{x})$, so every hypothesis is incorrect on every such initial state. Taking the probability over $\vec{x} \in D$ proves the lower bound. Conversely, fix an ordering of $\mathcal{Y}$ and define $h^*(\vec{x})$ to return the first fixed point of $r_{\vec{x}}$ whenever one exists, and an arbitrary prediction otherwise. This hypothesis is correct on every $\vec{x}$ admitting a fixed point and incorrect on every remaining state. Therefore, $\mathrm{err}_{D,\mathbf{c}}(h^*) = \alpha_{D,\mathbf{c}}$. □

The learner's objective is consequently to return, with probability at least $1 - \delta$, a hypothesis satisfying $\mathrm{err}_{D,\mathbf{c}}(h) \leq \alpha_{D,\mathbf{c}} + \epsilon$. In summary, we have the following learning problem.

Definition 11.9 (Introspective Forecasting) Let the set of possible *forecasts* $\mathcal{Y}$ be a partition of the possible states of a set of n fluents. Let C_n and $\mathcal{H}_n$ be classes of representations of domains on the same n fluents, where each $\mathbf{c} \in C_n$ contains a domain c_y for each prediction $y \in \mathcal{Y}$. *Learning introspective forecasting* with the set of forecasts $\mathcal{Y}$ for C_n using $\mathcal{H}_n$ at *horizon* τ with sample complexity $m(n, |\mathcal{Y}|, \frac{1}{\epsilon}, \frac{1}{\delta})$ is

the following task. For any world w obeying some $\mathbf{c} \in C_n$, given the parameters ϵ and δ and m examples of the form $\langle w(0), y, w(\tau)\rangle$ where each $w(0)$ is drawn independently according to D and $w(\tau) \in y$ for the unique $y \in \mathcal{Y}$, with probability at least $1-\delta$ output a representation $h \in \mathcal{H}_n$ such that

$$\mathrm{err}_{D,\mathbf{c}}(h) \leq \alpha_{D,\mathbf{c}} + \epsilon.$$

We next relate this learning problem to prediction of the underlying temporal evolution. Suppose that $\mathcal{Y}$ can be enumerated efficiently, that $|\mathcal{Y}|$ is polynomially bounded, and that the member of $\mathcal{Y}$ containing a given state can be identified efficiently. Then, we can perform efficient introspective forecasting given an efficient PAC-learning algorithm L such that for every domain $c \in C$, L learns the mapping from an initial state $\vec{x}$ to the state $w_{\vec{x}}(\tau)$ reached at time τ in the world $w_{\vec{x}}$ that obeys c, from examples of the form $\left\langle \vec{x}, w_{\vec{x}}(\tau)\right\rangle$.

Theorem 11.10 (Learning introspective forecasts) *Suppose that $\mathcal{Y}$ can be enumerated in time $t_{enum}(n)$ and that there are circuits of size $s_{\mathcal{Y}}(n)$ computable in time $O(s_{\mathcal{Y}}(n))$ that, given a representation of a state $\vec{x} \in \{0,1\}^n$, output a $O(\log|\mathcal{Y}|)$-bit representation of the unique $y \in \mathcal{Y}$ such that $\vec{x} \in y$. Suppose furthermore that there is a PAC-learning algorithm L for the class given by the states $w_{\vec{x}}(\tau)$ for worlds $w_{\vec{x}}$ with $w(0) = \vec{x}$ obeying $\{c_y : \mathbf{c} \in C_n, y \in \mathcal{Y}\}$ running in time $t_{\mathsf{L}}(n, \frac{1}{\epsilon}, \frac{1}{\delta})$ and with sample complexity $m_{\mathsf{L}}(n, \frac{1}{\epsilon}, \frac{1}{\delta})$, that produces as output representations of circuits of size $s_{\mathsf{L}}(n)$. Then, there is an algorithm for learning introspective forecasts that produces representations of circuits of size*

$$S(n) = O\big(|\mathcal{Y}|(s_{\mathsf{L}}(n) + s_{\mathcal{Y}}(n) + \log|\mathcal{Y}|)\big),$$

running in time

$$O\left(t_{enum}(n) + |\mathcal{Y}| t_{\mathsf{L}}\left(n, \frac{|\mathcal{Y}|}{\epsilon}, \frac{|\mathcal{Y}|}{\delta}\right) + S(n)\right),$$

and using $|\mathcal{Y}| \cdot m_{\mathsf{L}}\left(n, \frac{|\mathcal{Y}|}{\epsilon}, \frac{|\mathcal{Y}|}{\delta}\right)$ examples. In particular, for every $\epsilon, \delta > 0$, with probability at least $1-\delta$, the output circuit h satisfies

$$\mathrm{err}_{D,\mathbf{c}}(h) \leq \alpha_{D,\mathbf{c}} + \epsilon.$$

Proof For each $y \in \mathcal{Y}$, collect examples by independently drawing $\vec{x} \in D$, announcing y, and observing $w_{\vec{x},y}(\tau)$. Run L separately on the examples associated with each y, using error parameter $\epsilon/|\mathcal{Y}|$ and confidence parameter $\delta/|\mathcal{Y}|$.

By a union bound over these $|\mathcal{Y}|$ invocations, with probability at least $1-\delta$, the resulting hypotheses $\{g_y\}_{y\in\mathcal{Y}}$ satisfy

$$\Pr_{\vec{x}\sim D}\left[g_y(\vec{x}) \neq w_{\vec{x},y}(\tau)\right] \leq \frac{\epsilon}{|\mathcal{Y}|}$$

simultaneously for every $y \in \mathcal{Y}$.

For each initial state $\vec{x}$, define an approximate response function

$$\widehat{r}_{\vec{x}} : \mathcal{Y} \to \mathcal{Y}$$

by letting $\widehat{r}_{\vec{x}}(y)$ be the unique member of $\mathcal{Y}$ containing $g_y(\vec{x})$. Fix an ordering of $\mathcal{Y}$. The forecasting hypothesis h evaluates $\widehat{r}_{\vec{x}}(y)$ for every $y \in \mathcal{Y}$ and returns the first fixed point in the enumeration. If $\widehat{r}_{\vec{x}}$ has no fixed point, it returns an arbitrary member of $\mathcal{Y}$.

Let

$$E \stackrel{\text{def}}{=} \left\{\vec{x} : \text{there exists } y \in \mathcal{Y} \text{ such that } g_y(\vec{x}) \neq w_{\vec{x},y}(\tau)\right\}.$$

Another union bound gives

$$\Pr_{\vec{x}\in D}[E] \leq \sum_{y\in\mathcal{Y}} \Pr_{\vec{x}\in D}\left[g_y(\vec{x}) \neq w_{\vec{x},y}(\tau)\right] \leq |\mathcal{Y}|\frac{\epsilon}{|\mathcal{Y}|} = \epsilon.$$

Whenever $\vec{x} \notin E$, every prediction-conditioned endpoint is predicted correctly, and hence $\widehat{r}_{\vec{x}} = r_{\vec{x}}$. If $r_{\vec{x}}$ has a fixed point, the procedure therefore finds a genuine fixed point and makes a correct prediction. An error can occur only when $r_{\vec{x}}$ has no fixed point or when $\vec{x} \in E$. Consequently,

$$\mathrm{err}_{D,\mathbf{c}}(h) \leq \alpha_{D,\mathbf{c}} + \Pr_{\vec{x}\in D}[E] \leq \alpha_{D,\mathbf{c}} + \epsilon.$$

The procedure enumerates $\mathcal{Y}$ in time $t_{\text{enum}}(n)$ and makes $|\mathcal{Y}|$ calls to L, with parameters $\epsilon/|\mathcal{Y}|$ and $\delta/|\mathcal{Y}|$. It produces as output a representation h structured as follows. An input initial state $\vec{x}$ is passed to each of the $|\mathcal{Y}|$ respective hypotheses g_y of size $s_{\mathsf{L}}(n)$ each, each of which produces a prediction of a state that is fed to the circuits of size $s_{\mathcal{Y}}(n)$ that decode these to respective forecasts $\hat{y}(\vec{x})$, to produce a circuit computing $\widehat{r}_{\vec{x}}(y)$. The output of these circuits is tested for equality to the respective y, using a circuit of size $O(\log|\mathcal{Y}|)$. Observe that there is a circuit of size $O(\log|\mathcal{Y}|)$ that can take as input representations of $y^1, y^2 \in \mathcal{Y}$ and two Boolean flags, indicating whether the respective y^1 or y^2 are fixed points of the response function, and outputs y^1 if it is a fixed point and otherwise outputs y^2. A complete binary tree of such circuits, with each leaf receiving the result of the test for one $y \in \mathcal{Y}$, of total size $O(|\mathcal{Y}|\log|\mathcal{Y}|)$ thus computes the first fixed point, completing the construction of h of the claimed size $S(n)$. Since, apart from the time spent by the $|\mathcal{Y}|$ invocations of L to generate the circuits $\{g_y\}_{y\in\mathcal{Y}}$, this circuit can be generated in size $O(S(n))$, the claimed time bound follows. □

The theorem makes three concessions relative to a standard PAC-learning guarantee: the additive error $\alpha_{D,\mathbf{c}}$, running-time dependence on $|\mathcal{Y}|$, and the learner's ability to select and observe the announced prediction. Each reflects a genuine limitation.

First, the additive error is unavoidable when the response function has no fixed point, as we saw in Proposition 11.8.

Second, for deterministic procedures with black-box access to the response function, linear test-time search over $\mathcal{Y}$ (hence, circuit size $\Omega(|\mathcal{Y}|)$) is unavoidable.

Proposition 11.11 (Linear dependence on the prediction space) *Suppose that, for a fixed initial state, a forecasting procedure is given evaluation access to an otherwise arbitrary response function $r : \mathcal{Y} \to \mathcal{Y}$. Any deterministic procedure guaranteed to output a fixed point whenever one exists must make at least $|\mathcal{Y}| - 1$ evaluations in the worst case.*

Proof If $|\mathcal{Y}| = 1$, the bound is vacuous, so suppose $|\mathcal{Y}| \geq 2$. Fix a permutation $\sigma : \mathcal{Y} \to \mathcal{Y}$ having no fixed points, and answer every query y with $\sigma(y)$. Suppose that a procedure terminates after fewer than $|\mathcal{Y}| - 1$ queries and outputs z. At least two predictions remain unqueried. We may therefore choose an unqueried prediction $y^* \neq z$ and define $r(y^*) = y^*$, and $r(y) = \sigma(y)$ for every $y \neq y^*$. This response function agrees with every answer observed by the procedure but has y^* as its unique fixed point. Since the procedure outputs $z \neq y^*$, it fails. Hence at least $|\mathcal{Y}| - 1$ evaluations are required in the worst case. □

We remark that even though the *size* of the circuit h produced by Theorem 11.10 is $\Omega(|\mathcal{Y}| \log |\mathcal{Y}|)$, its *depth* is only that of one of the circuits output by L, plus a circuit converting these into forecasts from $\mathcal{Y}$, plus an additional $O(\log |\mathcal{Y}|)$. Forecasts thus may be computable efficiently in *parallel time*, in spite of Proposition 11.11.

Third, if predictions are selected externally and not given in the examples, the observed endpoints alone may be inadequate to learn introspective forecasters.

Proposition 11.12 (Hidden predictions can destroy identifiability) *Suppose that the announced prediction is selected uniformly from $\mathcal{Y}$ but is not revealed to the learner, and that a training observation contains only the initial and endpoint states. Then the resulting introspective forecast learning problem may be impossible, even with unlimited training data, even when $|\mathcal{Y}| = 3$, even when every domain has order zero, and even when $\alpha_{D,\mathbf{c}} = 0$.*

Proof Let D be concentrated on a single state $\vec{x}$, let $\mathcal{Y} = \{a, b, c\}$, and choose representative endpoint states $\vec{s}_a \in a$, $\vec{s}_b \in b$, and $\vec{s}_c \in c$. Consider the response functions $r^a(a) = a, r^a(b) = c, r^a(c) = b$, and $r^b(a) = c, r^b(b) = b, r^b(c) = a$. Their unique fixed points are a and b, respectively, so $\alpha_{D,\mathbf{c}^a} = \alpha_{D,\mathbf{c}^b} = 0$.

For any fixed $\tau \geq 1$, either response function can be realized by order-zero domains that force the endpoint under prediction y to be $\vec{s}_{r^a(y)}$ or $\vec{s}_{r^b(y)}$, respectively. Since both response functions are permutations of $\mathcal{Y}$, a uniformly selected but hidden prediction produces each of $\vec{s}_a, \vec{s}_b, \vec{s}_c$ with probability $1/3$ under either target family. The observable training distributions are therefore identical.

Any randomized algorithm consequently has the same output distribution under both targets. If its forecaster outputs a and b with probabilities p_a and p_b, then $p_a + p_b \leq 1$, so one of these probabilities is at most $1/2$. On the corresponding target, whose unique correct prediction is a or b, the algorithm's output fails with probability at least $1/2$. Hence no algorithm can provide the corresponding learning guarantee for $\epsilon < 1$ and $\delta < 1/2$. □

Together, these results delineate the scope of Theorem 11.10.

Chapter notes

TO DO

active sensing?

Exercises

11.1 *Causal modeling of a narrative.* Consider the following narrative about a simple greenhouse controller.

In the morning, the soil is dry, the irrigation pump is off, the water valve is closed, the reservoir is not empty, no leaks have been detected, no alarm is sounding, and the plants are not wilting. The controller is in automatic mode, and manual override is not active. If the soil is dry and the controller is in automatic mode, the pump turns on. If manual override is active, the pump turns off. If the pump is on and the reservoir is not empty, the valve opens. If the pump is on and the reservoir is empty, an alarm sounds. If the valve is open and no leak has been detected, water reaches the soil. If the valve is open and a leak has been detected, an alarm sounds. If water reaches the soil, the soil becomes wet. If the soil is dry and water does not reach it, the plants begin to wilt. If the soil is wet, the plants stop wilting. If an alarm is sounding, the controller turns the pump off and closes the valve.

(a) Choose Boolean fluents for representing the relevant features of the greenhouse state. For each fluent, state its intended meaning. In particular, decide carefully whether to represent certain pairs of features by using one fluent and its negation, or using two separate fluents. Explain your choices.
(b) Represent the morning state as a set of literals over your fluents.
(c) Write an initial version of a domain consisting of causal implications that capture the causal relationships described in the narrative.
(d) Examine your causal implications for possible conflicts. Identify any state in which two rules could trigger both a fluent and its negation, or in which two separately introduced fluents should be incompatible but could both become true.
(e) Revise your domain so that it satisfies the chapter's standing no-conflict assumption. You may do this by adding preconditions to rules, by changing your choice of fluents, or by explicitly replacing a fluent with the negation of another fluent. Explain each revision.
(f) Identify the order of each causal implication in your revised domain.

11.2 *Computing the temporal evolution from causal implications.* Consider the set of Boolean fluents

Q	=	"an access request is active",	C	=	"the credential is valid",
N	=	"the PIN is correct",	A	=	"the alarm is sounding",
Y	=	"access is granted",	U	=	"the door is unlocked",
O	=	"the door is open",	H	=	"the close timer is active",
R	=	"the access attempt is recorded".			

Let the initial state $w(0)$ satisfy the literals

$$\{Q, C, N, \neg A, \neg Y, \neg U, \neg O, \neg H, \neg R\}.$$

The domain c is given by the following causal implications:

$$\begin{aligned}
Q \wedge C \wedge N \wedge \neg A \wedge \neg H &\rightsquigarrow Y, & Q \wedge \neg C &\rightsquigarrow A,\\
Q \wedge \neg N &\rightsquigarrow A, & A &\rightsquigarrow \neg Y,\\
A &\rightsquigarrow \neg U, & A &\rightsquigarrow \neg O,\\
Y \wedge \neg A \wedge \neg H &\rightsquigarrow U, & U \wedge \neg A \wedge \neg H &\rightsquigarrow O,\\
O &\rightsquigarrow R, & O \wedge \neg H &\rightsquigarrow H,\\
H &\rightsquigarrow \neg O, & H &\rightsquigarrow \neg U,\\
H &\rightsquigarrow \neg Y, & H &\rightsquigarrow \neg Q,\\
H &\rightsquigarrow \neg H.
\end{aligned}$$

(a) For each time $t = 0, 1, 2, 3, 4, 5$, list the literals caused by $w(t)$. Remember that a literal caused at time t affects $w(t+1)$, not $w(t)$.
(b) Compute $w(1), w(2), w(3), w(4), w(5), w(6)$. For each fluent at each time, indicate whether its value is caused by a rule firing or persists by inertia.
(c) Identify any fluents that are false in both $w(0)$ and $w(6)$, but true at some intermediate time. Explain why observing only the initial and final states would hide these events.
(d) Identify the first time t at which the world reaches a fixed point, meaning $w(t+1) = w(t)$.
(e) Repeat the computation for the first three time steps under the modified initial state in which N is false instead of true. Explain how an incorrect PIN changes the causal chain.

11.3 *Reducing successor-state learning to coordinate-wise learning.* Fix Boolean fluents $X_1, \dots, X_n$. Let c be a domain, and let $F_c : \{0,1\}^n \to \{0,1\}^n$ be its successor-state function, so that $F_c(x)$ is the state following x in any world obeying c. For each coordinate i, let $f_i^c : \{0,1\}^n \to \{0,1\}$ be the ith coordinate of F_c. Suppose there is a PAC learning algorithm L for a class of Boolean representations $\mathcal{H}$, and suppose that for every domain c in some class of representations C, each coordinate function f_i^c belongs to $\mathcal{H}$.
(a) Given examples of the form

$$(x^{(1)}, F_c(x^{(1)})), \dots, (x^{(m)}, F_c(x^{(m)})),$$

explain how to construct, for each coordinate i, a labeled sample for learning f_i^c.
(b) Run L separately on the n coordinate samples to obtain hypotheses $h_1, \dots, h_n$. Define the vector-valued hypothesis

$$H(x) = (h_1(x), \dots, h_n(x)).$$

Show that if each h_i has error at most ϵ/n with respect to the marginal distribution on current states, then

$$\Pr[H(x) \neq F_c(x)] \leq \epsilon.$$

(c) Choose the confidence parameters for the n runs of L so that, with probability at least $1 - \delta$, all coordinate hypotheses satisfy the accuracy guarantee needed in part (b).

(d) Conclude that if $\mathcal{H}$ is PAC-learnable, then the class of successor-state functions induced by $\mathcal{C}$ is PAC-learnable as a vector-valued prediction problem. State the resulting sample complexity in terms of the sample complexity of L, n, ϵ, and δ.
(e) Explain why this reduction learns the successor-state map F_c, but does not necessarily recover a causal domain c whose implications explain the transition.

11.4 *Restricted order is necessary for efficient causal learning.* Prove that if monotone domains of unrestricted order are efficiently PAC-learnable from consecutive-state examples, then polynomial-size DNF formulas are efficiently PAC-learnable.