Training Garrabrant inductors to predict counterfactuals

Tsvi Benson-Tilsen

The ideas in this post are due to Scott, me, and possibly others. Thanks to Nisan Stiennon for working through the details of an earlier version of this post with me.

We will use the notation and definitions given in `https://github.com/tsvibt/public-pdfs/blob/master/decision-theory/notation/main.pdf`. Let $\overline{\mathbb{P}}$ be a universal Garrabrant inductor and let $\overline{U} : \mathbb{N}^+ \to \mathrm{Expr}(2^\omega \to \mathbb{R})$ be a sequence of utility function machines. We will define an agent schema $(A_n^{U_n})$.

We give a schema where each agent selects a single action with no observations. Roughly, $A_n^{U_n}$ learns how to get what it wants by computing what the $A_i^{U_i}$ with $i < n$ did, and also what various traders predicted would happen, given each action that the $A_i^{U_i}$ could have taken. The traders are rewarded for predicting what (counterfactually) would be the case in terms of bitstrings, and then their predictions are used to evaluate expected utilities of actions currently under consideration. This requires modifying our UGI and the traders involved to take a possible action as input, so that we get a prediction (a "counterfactual distribution over worlds") for each action.

More precisely, define

$$\begin{aligned} A_n^{U_n} := \text{let } \hat{\mathbb{P}}_n &:= \text{Counterfactuals}(n) \\ &\text{return } \arg\max_{a \in \mathrm{Act}} \hat{\mathbb{E}}_n[a](U_n) \end{aligned}$$

where

$$\hat{\mathbb{E}}_n[a](U_n) := \sum_{\sigma \in 2^n} \hat{\mathbb{P}}_n[a](\sigma) \cdot U_n(\sigma).$$

Here $\hat{\mathbb{P}}_n$ is a dictionary of belief states, one for each action, defined by the function $\text{Counterfactuals} : \mathbb{N}^+ \to (\mathrm{Act} \to \Delta(2^\omega))$ using recursion as follows:

Algorithm 1: $\text{Counterfactuals} : \mathbb{N}^+ \to (\mathrm{Act} \to \Delta(2^\omega))$

input : $n \in \mathbb{N}^+$
output : A dictionary of belief states $\mathbb{P} : \mathrm{Act} \to \Delta(2^\omega)$

$\text{hist}_{n-1} \leftarrow$ array of belief states of length $n-1$;
for $i \leq n-1$ **do**
  $\hat{\mathbb{P}}_i \leftarrow \text{Counterfactuals}(i)$;
  $a_i \leftarrow \arg\max_{a \in \mathrm{Act}} \sum_{\sigma \in 2^i} \hat{\mathbb{P}}_i[a](\sigma) \cdot U_i(\sigma)$;
  $\text{hist}_{n-1}[i] \leftarrow \hat{\mathbb{P}}_i[a_i]$
for $(a : Act)$ **do**
  $\mathbb{P}[a] \leftarrow \text{MarketMaker}(\text{hist}_{n-1}, \text{TradingFirm}'(a, a_{\leq n-1}, \text{hist}_{n-1}))$
return $\mathbb{P}$

Here, we use a modified form of traders and of the TradingFirm$'$ function from the LIA algorithm given in the logical induction paper. In detail, let traders have the type

$$\mathbb{N}^+ \times \text{Act} \to \text{trading strategy}.$$

On day n, traders are passed a possible action $a \in \text{Act}$, which we interpret as "an action that $A_n^{U_n}$ might take". Then each trader returns a trading strategy, and those trading strategies are used as usual to construct a belief state $\mathbb{P}[a]$. We pass to TradingFirm$'$ the full history $a_{\leq n-1}$ of the actions taken by the previous $A_i^{U_i}$, since TradingFirm$'$ calls the Budgeter function; that function requires computing the traders's previous trading strategies, which require passing the a_i as arguments.

Thus, traders are evaluated based on the predictions they made about logic when given the actual action a_n as input. In particular, the sequence $(\mathbb{P}_n[a_n])$ is a UGI over the class of efficient traders given access to the actual actions taken by the agent $A_n^{U_n}$.

This scheme probably suffers from spurious counterfactuals, but feels like a natural baseline proposal.