A Alchemy mechanics

A.1 The chemistry

We consider three perceptual features along which stones can vary: size, color, and shape. Potions are able to change stone perceptual features, but how a specific potion can affect a particular stone's appearance is determined by the stone's (unobserved) latent state $\mathbf{c}$. Each potion deterministically transforms stones according to a hidden, underlying transition graph sampled from the set of all connected graphs formed by the edges of a cube (see Figure 7a). The corners of the cube represent different latent states, defined by a 3-dimensional coordinate $\mathbf{c} \in \{-1, 1\}^3$. Potion effects align with one axis and direction of the cube, such that one of the coordinates $\mathbf{c}$ is modified from -1 to 1 or 1 to -1. In equations:

$$\mathbf{c} = \mathbf{c} + 2\mathbf{p}\mathbb{1}_{(\mathbf{c},\mathbf{c}+2\mathbf{p})\in G}(\mathbf{c}) \tag{1}$$

where $\mathbf{p} \in \mathcal{P}$ is the potion effect $\mathcal{P} := \{\boldsymbol{e}^{(0)}, \boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, -\boldsymbol{e}^{(0)}, -\boldsymbol{e}^{(1)}, -\boldsymbol{e}^{(2)}\}$ where $\boldsymbol{e}^{(i)}$ is the i^{th} basis vector and G is the set of edges which can be traversed in the graph.

The latent state of the stone also determines its reward value $R \in \{-3, -1, 1, 15\}$, which can be observed via the brightness of the reward indicator (square light) on the stone:

$$R(\mathbf{c}) = \begin{cases} 15, & \text{if} \sum_i c_i = 3 \\ \sum_i c_i, & \text{else.} \end{cases} \tag{2}$$

The agent only receives a reward for a stone if that stone is successfully placed within the cauldron by the end of the trial, which removes the stone from the game.

The latent coordinates $\mathbf{c}$ are mapped into the stone perceptual feature space to determine the appearance of the stone. We call this linear mapping the stone map or S and define it as $S\colon \{-1, 1\}^3 \to \{-1, 0, 1\}^3$:

$$S(\mathbf{c}) = \mathbf{S}_{\text{rotate}}\mathbf{S}_{\text{reflect}}\mathbf{c} \tag{3}$$

where $\mathbf{S}_{\text{rotate}}$ denotes possible rotation around 1 axis and rescaling. Formally: $\mathbf{S}_{\text{rotate}} \sim U(\{\boldsymbol{I}_3, \overline{R_x}(45°), \overline{R_y}(45°), \overline{R_z}(45°)\})$, where I is the identity matrix, and $\overline{R_i}(\theta)$ denotes an anti-clockwise rotation transform around axis i by $\theta = 45°$, followed by scaling by $\frac{\sqrt{2}}{2}$ on all other axes, in order to normalize values to be in $\{-1, 0, 1\}$. $\mathbf{S}_{\text{reflect}}$ denotes reflection in the x, y and z axes: $\mathbf{S}_{\text{reflect}} = \text{diag}(\mathbf{s})$ for $\mathbf{s} \sim U(\{-1, 1\}^3)$.

The potion effect $\mathbf{p}$ is mapped to a potion color by first applying a linear mapping and then using a fixed look-up table to find the color. We call this linear mapping the potion map $P\colon \mathcal{P} \to \mathcal{P}$.

$$P(\mathbf{p}) = \mathbf{P}_{\text{reflect}}\mathbf{P}_{\text{permute}}\mathbf{p} \tag{4}$$

where $\mathbf{P}_{\text{reflect}}$ is drawn from the same distribution as $\mathbf{S}_{\text{reflect}}$ and $\mathbf{P}_{\text{permute}}$ is a 3x3 permutation matrix i.e. a matrix of the form $[\boldsymbol{e}^{(\pi(0))}, \boldsymbol{e}^{(\pi(1))}, \boldsymbol{e}^{(\pi(2))}]^T$, $\pi \sim U(\text{Sym}(\{0, 1, 2\}))$ where $\text{Sym}(\{0, 1, 2\})$ is the set of permutations of $\{0, 1, 2\}$.

The directly observable potion colors are then assigned according to:

$$P_{\text{color}} = \begin{cases} (\boldsymbol{e}^{(0)}, -\boldsymbol{e}^{(0)}) \to (\text{green}, \text{red}) \\ (\boldsymbol{e}^{(1)}, -\boldsymbol{e}^{(1)}) \to (\text{yellow}, \text{orange}) \\ (\boldsymbol{e}^{(2)}, -\boldsymbol{e}^{(2)}) \to (\text{turquoise}, \text{pink}) \end{cases} \tag{5}$$

Note that this implies that potion colors come in pairs so that, for example, the red potion always has the opposite effect to the green potion, though that effect may be on color, size, or shape, depending on the particular chemistry of that episode. It can also have an effect on two of those three perceptual features simultaneously in the case where $\mathbf{S}_{\text{rotate}} \neq \boldsymbol{I}_3$. This color pairing of potions is consistent across all samples of the task, constituting a feature of Alchemy which can be meta-learned over

many episodes. Importantly, due to the potion map P, the potion effects $\mathbf{p}$ in each episode must be discovered by experimentation.

G is determined by sampling a graph topology (Figure 7d), which determines which potion effects are possible. Potions only have effects if certain preconditions on the stone latent state are met, which constitute 'bottlenecks' (darker edges in Figure 7d). Each graph consists of the edges of the cube which meet the graph's set of preconditions. Each precondition says that an edge parallel to axis i exists only if its value on axis j is a where $j \neq i$ and $a \in \{-1, 1\}$. The more preconditions, the fewer edges the graph has.

Only sets of preconditions which generate a connected graph are allowed. We denote the set of connected graphs with preconditions of this form $\mathbb{G}$. Note that this is smaller than the set of all connected graphs, as a single precondition can rule out 1 or 2 edges of the cube. As with the potion color pairs, this structure is consistent across all samples and may be meta-learned.

We find that the maximum number of preconditions for any graph $G \in \mathbb{G}$ is 3. We define $\mathbb{G}^n := \{G \in \mathbb{G} | N(G) = n\}$ where $N(G)$ is the number of preconditions in G. The sampling distribution is $n \sim U(0, 3)$, $G \sim U(\mathbb{G}^n)$. Of course, there is only one graph with 0 preconditions and many graphs with 1, 2, or 3 preconditions so the graph with 0 preconditions is the most common and is sampled 25% of the time.

A 'chemistry' is a random sampling of all variables $\{G, \mathbf{P}_{\text{permute}}, \mathbf{P}_{\text{reflect}}, \mathbf{S}_{\text{reflect}}, \mathbf{S}_{\text{rotate}}\}$ (subject to the constraint rules described above), which is held constant for an episode (Figure 6). Given all of the possible permutations, we calculate that there are 167,424 total chemistries that can be sampled.

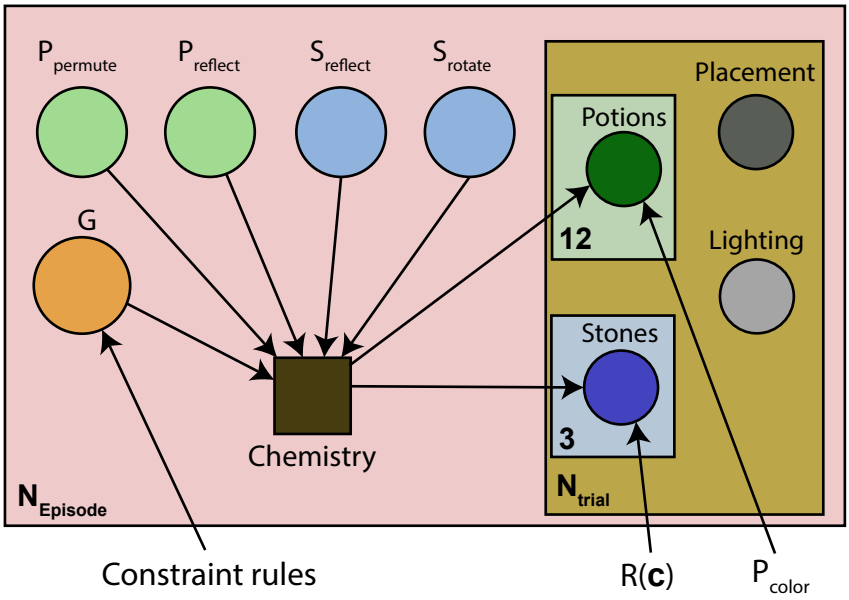

Figure 6: The generative process for sampling a new task, in plate notation. Constraint rules $\mathbb{G}$, P_{color}, and $R(\mathbf{c})$ are fixed for all episodes (see Section A.1). Every episode, a set $\{G, \mathbf{P}_{\text{permute}}, \mathbf{P}_{\text{reflect}}, \mathbf{S}_{\text{reflect}}, \mathbf{S}_{\text{rotate}}\}$ is sampled to form a new chemistry. Conditioned on this chemistry, for each of $N_{trial} = 10$ trials, $N_s = 3$ stones and $N_p = 12$ potions are sampled, as well as random placement and lighting conditions, to form the perceptual observation for the agent. For clarity, the above visualization omits parameters for normal or uniform distributions over variables (such as lighting and placement of stones/potions on the table).

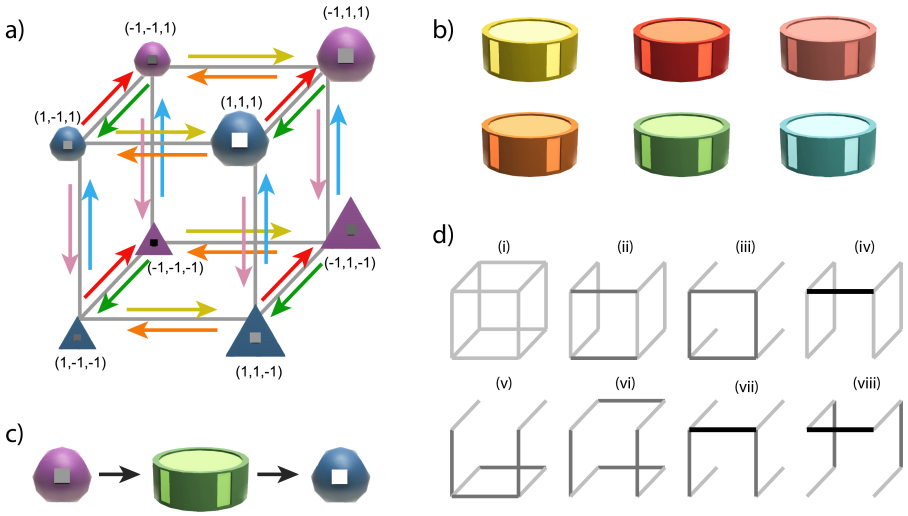

Figure 7: a) Depiction of an example chemistry sampled from Alchemy, in which the perceptual features happen to be axis-aligned with the latent feature coordinates (listed beside the stones). Stones transform according to a hidden transition structure, with edges corresponding to the application of corresponding potions, as seen in b). c) For example, applying the green potion to the large purple stone transforms it into a large blue stone, and also increases its value (indicated by the square in center of stone becoming white). d) The possible graph topologies for stone transformation. Darker edges indicate 'bottlenecks', which are transitions that are only possible from certain stone latent states. In topologies with bottlenecks, more potions are often required to reach the highest value stone. If the criteria for stone states are not met, then the potion will have no effect (e.g. if topology (v) has been sampled, the yellow potion in the example given in (a) will have no effect on the small purple round stone). Note that we can apply reflections and rotations on these topologies, yielding a total of 109 configurations.

A.2 Episode structure

Episodes end at the end of 10 trials, with a fixed number of steps per trial. There are 20 steps per trial for symbolic Alchemy (200 for the entire episode), and 1800 frames (18000 for the entire episode, at a frame rate of 30 fps) for 3D Alchemy.

The chemistry is procedurally generated at the beginning of every episode, and held constant for the entire episode. Conditioned on this chemistry, stones and potions are sampled at the beginning of every trial, with different placement and lighting (in 3D Alchemy).

B Training details

B.1 Compute and hyperparameters

For each experiment running VMPO on the symbolic Alchemy environment we used 460 TPUv2 core hours and 1.4 million CPU core hours at 2.4GHz. For each experiment running VMPO on the full 3d Alchemy environment we used 2500 TPUv3 core hours and 1.7 million CPU core hours at 2.4GHz. For each experiment running IMPALA on the full 3d Alchemy environment we used 560 TPUv3 core hours and 200 thousand CPU core hours at 2.4GHz. However, please note that VMPO and IMPALA are general purpose deep RL agents designed for completely different tasks, and so it is not surprising that they are not efficient at Alchemy. We expect that new agents designed to solve the meta-learning challenge proposed by Alchemy would be able to be much more data efficient.

Coarse hyperparameter searches were conducted in order to balance resource constraints against the complexity of the task, doing sweeps (2-3 values) over learning rate and agent-specific hyperparameters such as the epsilon temperature and target update period for VMPO. Default hyperparameters were taken either from the original VMPO [46, 55] and Impala [16] papers, or swept in consultation with the authors, based on their best estimate of what could work for Alchemy.

Table 2: Architecture and hyperparameters for VMPO.

SETTING	VALUE
IMAGE RESOLUTION:	96X72X3
NUMBER OF ACTION REPEATS:	4
AGENT DISCOUNT:	0.99
RESNET NUM CHANNELS:	64, 128, 128
TRXL MLP SIZE:	256
TRXL NUMBER OF LAYERS:	6
TRXL NUMBER OF HEADS:	8
TRXL KEY/VALUE SIZE:	32
ϵ_η:	0.5
ϵ_α:	0.001
T_{TARGET}:	100
β_π:	1.0
β_V:	1.0
$\beta_{\text{PIXEL CONTROL}}$:	0.001
$\beta_{\text{KICKSTARTING}}$:	10.0
β_{STONE}:	2.4
β_{POTION}:	0.6
$\beta_{\text{CHEMISTRY}}$:	10.0

Table 3: Architecture and hyperparameters for Impala.

SETTING	VALUE
IMAGE RESOLUTION:	96X72X3
NUMBER OF ACTION REPEATS:	4
AGENT DISCOUNT:	0.99
LEARNING RATE:	0.00033
RESNET NUM CHANNELS:	16, 32, 32
LSTM HIDDEN UNITS:	256
β_π:	1.0
β_V:	0.5
β_{ENTROPY}:	0.001
$\beta_{\text{PIXEL CONTROL}}$:	0.2
$\beta_{\text{KICKSTARTING}}$:	8.0

Algorithm 1 Ideal Observer

Input: stones s, potions p, belief state b
Initialise rewards = { }
for all $s_i \in s$ **do**
  for all $p_j \in p$ **do**
    $s_{\text{poss}}, p_{\text{poss}}, b_{\text{poss}}, b_{\text{probs}}$ = simulate use potion(s, p, s_i, p_j, b)
    $r = 0$
    for all $s', p', b', prob \in s_{\text{poss}}, p_{\text{poss}}, b_{\text{poss}}, b_{\text{probs}}$ **do**
      $r = r$ + $prob$ * Ideal Observer(s', p', b')
    end for
    rewards$[s_i, p_j] = r$
  end for
  s', r = simulate use cauldron(s, s_i)
  rewards[s_i, cauldron] = r + Ideal Observer(s', p, b)
end for
return argmax(rewards)

Algorithm 2 Oracle

Input: stones s, potions p, chemistry c
Initialise rewards = { }
for all $s_i \in s$ **do**
  for all $p_j \in p$ **do**
    s', p' = simulate use potion(s, p, s_i, p_j, c)
    rewards$[s_i, p_j]$ = Oracle(s', p', c)
  end for
  s', r = simulate use cauldron(s, s_i)
  rewards[s_i, cauldron] = r + Oracle(s', p, c)
end for
return argmax(rewards)

B.2 Auxiliary task losses

Auxiliary prediction tasks include: 1) predicting the number of stones currently present that possess each possible perceptual feature (e.g. small size, blue color etc), 2) predicting the number of potions of each color, or 3) predicting the ground truth chemistry. Auxiliary tasks contribute additional losses which are summed with the standard RL losses, weighted by coefficients ($\beta_{stone} = 2.4$, $\beta_{potion} = 0.6$, $\beta_{chem} = 10.0$). These hyperparameters were determined by roughly balancing the gradient norms of variables which contributed to all losses. All prediction tasks use an MLP head, (1) and (2) use an L2 regression loss while (3) uses a cross entropy loss. Prediction tasks (1) and (2) were always done in conjunction and are collectively referred to as 'Predict: Features', while (3) is referred to as 'Predict: Chemistry' in the results.

The MLP for 1) has 128x128x13 units where the final layer represents 3 predictions for each perceptual feature value e.g. the number of small stones, medium stones, large stones and 4 predictions for the brightness of the reward indicator. The MLP for 2) has 128x128x6 units where the final layer represents 1 prediction for the number of potions of each possible color. 3) is predicted with an MLP with a sigmoid cross entropy loss and has size 256x128x28 where the final layer represents predictions of the symbolic representations of the graph and mappings ($\mathbf{S}_{\text{rotate}}, \mathbf{S}_{\text{reflect}}, \mathbf{P}_{\text{reflect}}, \mathbf{P}_{\text{permute}}, G$). More precisely, $\mathbf{S}_{\text{rotate}}$ is represented by a 4 dimensional 1-hot, $\mathbf{S}_{\text{reflect}}$ and $\mathbf{P}_{\text{reflect}}$ are represented by 3 dimensional vectors with a 1 in the i^{th} element denoting reflection in axis i, $\mathbf{P}_{\text{permute}}$ is represented by a 6 dimensional 1-hot and G is represented by a 12-dimensional vector for the 12 edges of the cube with a 1 if the corresponding edge exists and a 0 otherwise.

Algorithm 3 Random heuristic

Input: stones s, potions p, threshold t
s_i = random choice(s)
if reward(s_i) > t **or** (reward(s_i) > 0 **and** empty(p)) **then**
  return s_i, cauldron
end if
p_i = random choice(p)
return s_i, p_i

C Additional results

Training curves show that VMPO agents train faster in the symbolic version of Alchemy vs the 3D version (Figures 8 and 9), even though agents are kickstarted in 3D.

As seen in Figure 9, the auxiliary task of predicting features appears much more beneficial than predicting the ground truth chemistry, the latter leading to slower training and more variability across seeds. We hypothesize that this is because predicting the underlying chemistry is possibly as difficult as simply performing well on the task, while predicting simple feature statistics is tractable, useful, and leads to more generalizable knowledge.

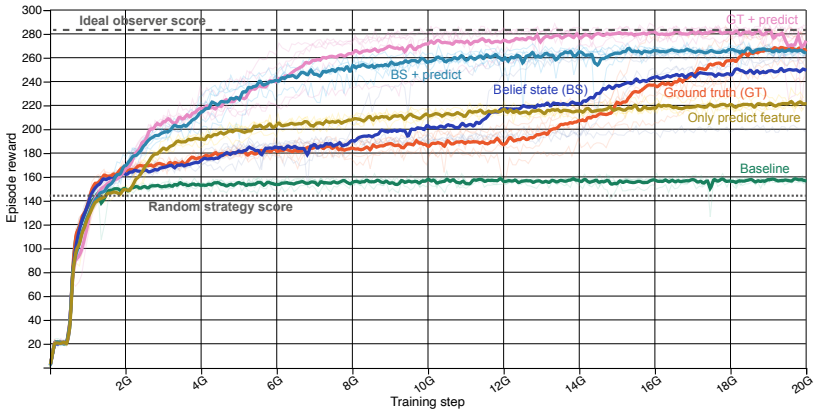

Figure 8: 3D with symbolic input training. Data are smoothed by bucketing and averaging over 2M steps per bucket (for a total of 1000 points). Thin lines indicate individual replicas (5 per condition).

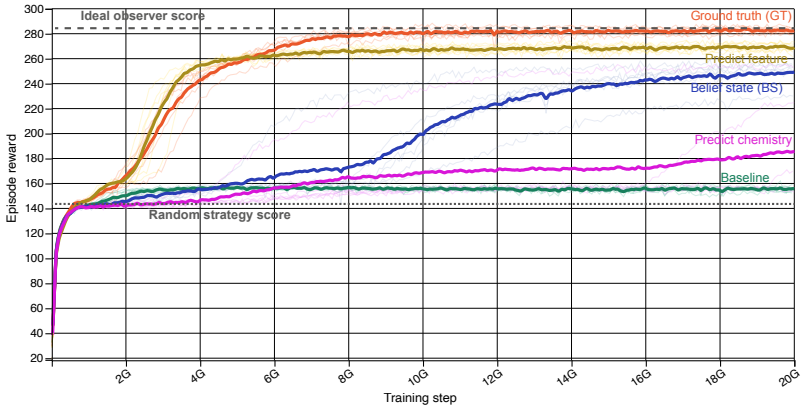

Figure 9: Symbolic alchemy training. Data are smoothed by bucketing and averaging over 2M steps per bucket (for a total of 1000 points). Thin lines indicate individual replicas (5 per condition).

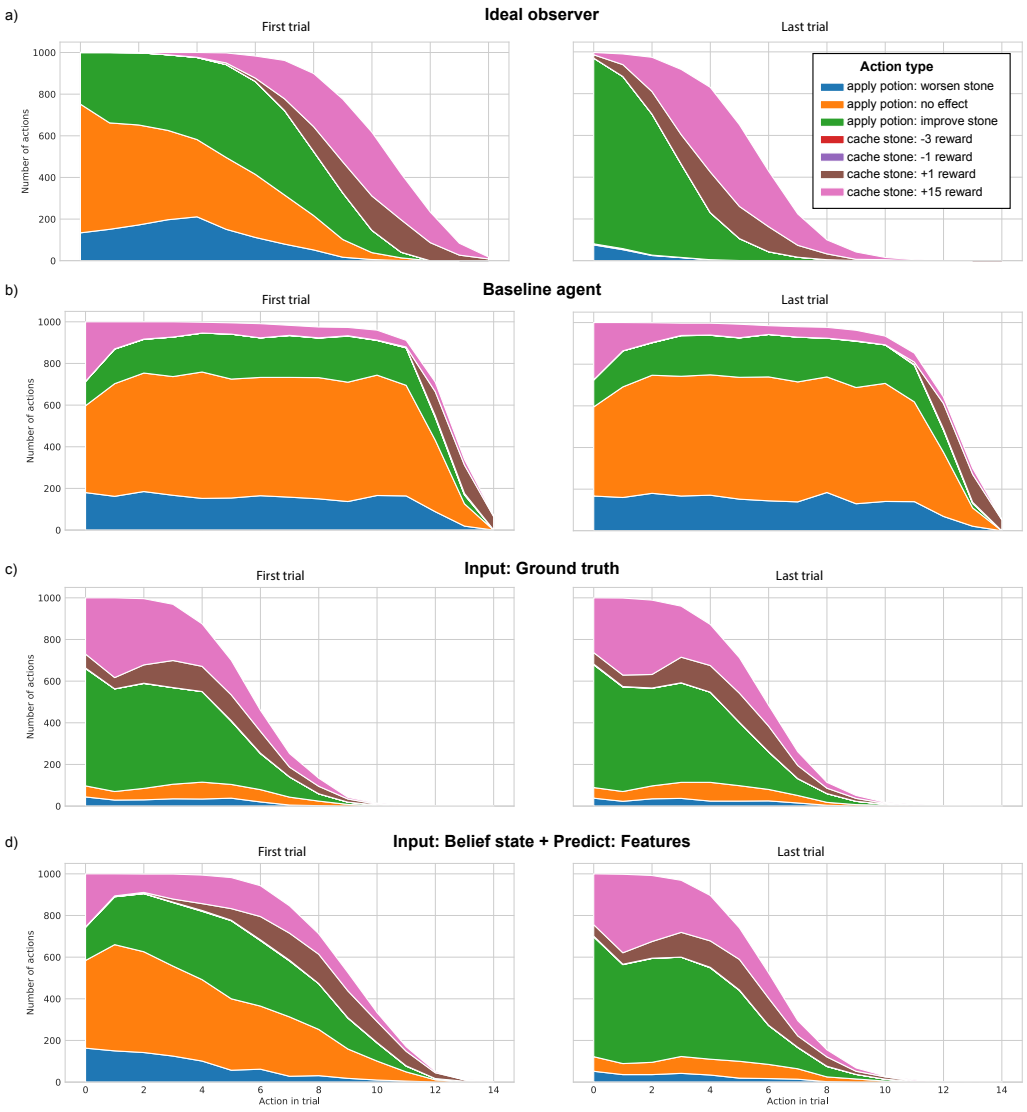

Figure 10: Comparing action types throughout trial in 3D alchemy, for a) ideal observer, b) baseline agent, c) agent with ground truth information as input, and d) agent with belief state as input and feature prediction auxiliary task. All agents include the symbolic observations as input. The ideal strategy is to in trial 1 perform a lot of exploratory actions (yielding actions that lead to no effect on the stone), but then in later trials perform only actions that change the value of the stone. Unlike the ideal observer, the baseline agent (b) and agent with ground truth input (c) display no change between the first and last trials, indicating an inability to adapt strategies (exploration vs exploitation) throughout the course of the episode.

D Licenses and documentation

The Alchemy environment and analysis tools can be found at https://github.com/deepmind/dm_alchemy and are released under the Apache License 2.0. Further licensing details and all documentation, including a colab tutorial, can be found at this repository.

Code will be updated and maintained by a subset of the authors with support from DeepMind and contributions from the community.