Cognition 254 (2025) 105967

Contents lists available at ScienceDirect

Cognition

journal homepage: www.elsevier.com/locate/cognit

Full length article

An algorithmic account for how humans efficiently learn, transfer, and compose hierarchically structured decision policies

Jing-Jing Li [a,*], Anne G.E. Collins [a,b]

[a] Helen Wills Neuroscience Institute, University of California, Berkeley, United States of America
[b] Department of Psychology, University of California, Berkeley, United States of America

ARTICLE INFO

Dataset link: https://github.com/jl3676/learning_hierarchy, https://experiments-ccn.berkeley.edu/learning_hierarchy_task_demo/exp.html?id=demo

Keywords:
Decision-making
Reinforcement learning
Computational cognitive modeling
Abstraction
Hierarchy
Meta-learning
Transfer learning
Compositionality

ABSTRACT

Learning structures that effectively abstract decision policies is key to the flexibility of human intelligence. Previous work has shown that humans use hierarchically structured policies to efficiently navigate complex and dynamic environments. However, the computational processes that support the learning and construction of such policies remain insufficiently understood. To address this question, we tested 1026 human participants, who made over 1 million choices combined, in a decision-making task where they could learn, transfer, and recompose multiple sets of hierarchical policies. We propose a novel algorithmic account for the learning processes underlying observed human behavior. We show that humans rely on compressed policies over states in early learning, which gradually unfold into hierarchical representations via meta-learning and Bayesian inference. Our modeling evidence suggests that these hierarchical policies are structured in a temporally backward, rather than forward, fashion. Taken together, these algorithmic architectures characterize how the interplay between reinforcement learning, policy compression, meta-learning, and working memory supports structured decision-making and compositionality in a resource-rational way.

1. Introduction

From choosing a career to choosing socks, we make decisions all the time in our daily life, whether big or small. In this process, we learn strategies that guide us to make decisions in the future informed by our past experience. Such a strategy can be described by a *policy* function that takes the current *state* of the environment as an input and outputs some *action* to take in this state. For an example of a simple decision policy, imagine that you have a tomato and want to decide what to do with it. What is the first thing that comes to mind? You could slice it, roast it, store it in the fridge, or more. In this case, the tomato is the state, and you used some policy that you have learned through your past experience with tomatoes to choose an action.

However, real-life decisions are rarely isolated from other decisions like in this toy example – they are often interconnected in *structured* ways to *contextualize* one another. An earlier decision might affect the policy for a later decision: if you had just preheated the oven, your policy on how to handle a tomato might prefer the roasting action, while if you had just sliced some sandwich bread, your policy would be more likely to favor slicing the tomato (Fig. 1A). Humans excel at understanding the connections between related decisions, states, and actions, and learning structured policies that guide us to effectively navigate complex and dynamic environments (Wise, Emery, & Radulescu, 2023). A prominent theme in human cognitive representations of decision structures is *hierarchy*: humans learn hierarchically structured decision policies, in which high-level representations form *abstractions* over low-level representations (Badre, 2008; Badre & D'Esposito, 2007; Botvinick, Niv, & Barto, 2009; Collins & Frank, 2013; Diuk et al., 2013; Eckstein & Collins, 2020; Ho et al., 2022; Koechlin, Ody, & Kouneiher, 2003; Li, Xia, Dong, & Collins, 2022; Solway et al., 2014; Tomov, Yagati, Kumar, Yang, & Gershman, 2020; Xia & Collins, 2021). Such abstractions can form over time and states, serving as a foundation of efficient and flexible learning by reducing the computational cost of decision-making and enabling *compositionality*: the ability to re-arrange, combine, and reuse abstracted policy structures in novel ways to create new policies that can solve new tasks (Franklin & Frank, 2018; Lake, Ullman, Tenenbaum, & Gershman, 2017).

When hierarchical representations are temporally abstracted, a sequence of actions are *chunked* together via some policy (Botvinick, 2007; Botvinick et al., 2009; Correa et al., 2023; Xia & Collins, 2021). These action chunks can be organized by subgoals that decompose a bigger task into smaller subtasks that are easier to solve (Diuk et al., 2013; Eckstein & Collins, 2020). In terms of our tomato example, the actions of preheating the oven and roasting the tomato can be chunked

* Corresponding author.
E-mail addresses: jl3676@berkeley.edu (J.-J. Li), annecollins@berkeley.edu (A.G.E. Collins).

https://doi.org/10.1016/j.cognition.2024.105967
Received 6 June 2024; Received in revised form 17 September 2024; Accepted 23 September 2024
Available online 4 October 2024

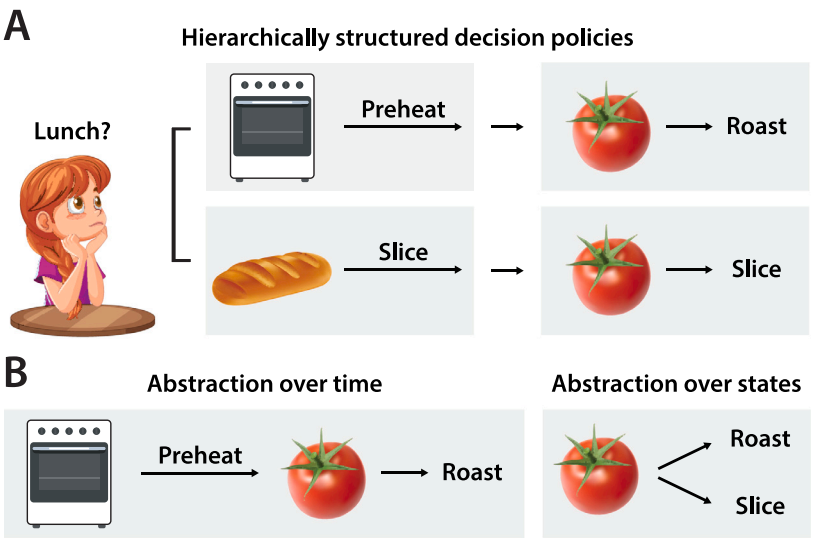

Fig. 1. Hierarchically structured decision policies and abstractions of policy information. A: In a hierarchically structured decision policy, the decision at a later time (e.g., what action to take with a tomato) is conditional on not only the immediate state (e.g., tomato), but also other related states and actions (e.g., having preheated the oven or sliced sandwich bread). B: Hierarchical policies can involve abstractions over time, in which a sequence of actions are chunked together via a policy, and abstractions over states, in which actions are chunked based on similarities between state representations.

or abstracted over time and described by a policy, which can serve as a subpolicy of a hierarchical policy for the higher-level task of making tomato soup (Fig. 1B). Cognitive scientists have used the options framework (Sutton, Precup, & Singh, 1999) from hierarchical reinforcement learning to model temporally abstracted policies (Botvinick et al., 2009; Xia & Collins, 2021). Unlike classic reinforcement learning (Sutton & Barto, 2018), in which the agent samples a single action at each time step, the options framework allows the agent to alternatively sample a policy that it can use to generate a sequence of actions, thus giving rise to temporal abstractions. Under this formulation, options are *temporally forward* structures in which earlier states contextualize or activate policies over later states to generate temporally abstracted action sequences.

In state abstractions, similar states are grouped to form compressed representations of the full state space (Abel, Arumugam, Lehnert, & Littman, 2018; Li, Walsh, & Littman, 2006). For example, both courses of actions illustrated in Fig. 1A involve the tomato, which can be compressed into a single abstract state representation (Fig. 1B). Recent work has shown that humans learn compressed policies over states to jointly maximize reward and representation efficiency (Lai & Gershman, 2021, 2024; Lai, Huang, & Gershman, 2022). State abstractions not only result in lower computational costs, but they are also essential to humans' ability to transfer knowledge between state components shared by related tasks and contexts (Collins, Cavanagh, & Frank, 2014; Collins & Frank, 2013, 2016; Lehnert, Littman, & Frank, 2020). In our example, the skill of slicing tomatoes can be further shared through the abstract tomato state with related tasks such as making a pizza, which may require sliced tomatoes as toppings.

Despite abundant evidence for temporal and state abstractions in human decision-making, it remains unclear how these abstractions are learned and how they shape the representations of hierarchical policies. How does a person learn to make tomato soups and sandwiches from knowing nothing about cooking? Is this learning top-down (start learning from the top of the hierarchy, e.g. with the overall structure of soup recipes and sandwich recipes) or bottom-up (learn the low-level policies for individual steps before the hierarchical structure, e.g. slicing tomatoes and turning on the oven)? How are the learned hierarchical policies represented to facilitate transferring the skill of slicing tomatoes from the task of making a sandwich to the task of making a pizza? At the core of hierarchy and abstractions is the *compression* of policy information over time and states, but we lack a satisfactory understanding of how compression supports hierarchy at the algorithmic level. How do temporal and state abstractions interact with each other to generate hierarchically structured decisions? How does compression support the learning and construction of hierarchical policies? How are hierarchies represented in the mind to enable transfer and composition between policies in structurally related tasks?

To address these questions, we used an experimental protocol that extends the paradigm of Xia and Collins (2021) and Li et al. (2022) to characterize how humans develop and compose hierarchical representations to guide behavior during trial-by-trial learning from deterministic feedback. Prior work has shown that humans can learn hierarchically structured policies and compose them to form new policies by transferring between contexts (Xia & Collins, 2021) without catastrophic forgetting of existing policy representations (Li et al., 2022) in this task. However, existing computational frameworks do not provide a satisfactory account on the algorithmic level for how these hierarchical structures are learned. Current models also cannot explain apparent knowledge transfer effects in some situations where the new rule is different and no transfer is expected. Specifically, they fail to qualitatively capture the transfer effect between different test structures (denoted as V1 and V2 in the current work) observed by Li et al. (2022), suggesting that a more concrete theory of how humans learn and represent hierarchical decision policies may be crucial to understanding how they compose policy structures to transfer efficiently between related problems.

Building on previous findings, we incorporated novel structural designs with robust controls to further investigate how state and temporal abstractions shape the learning of hierarchical policies. Specifically, we ask two questions about how complex decision structures are learned and represented: how do humans learn hierarchical structures from scratch using state abstractions and how are these learned representations ordered temporally? We propose that humans construct hierarchical policies in a bottom-up fashion: they start by learning simpler policies with efficient state abstractions, which are gradually expanded into more complex structures over learning. To evaluate the impact of this previously overlooked meta-learning effect on behavior, we developed a meta-learning model and compared it to the fully hierarchical model of Xia and Collins (2021) on human choice data. Additionally, we tested two alternative hypotheses for how temporal abstractions are ordered in learned hierarchical policy representations: whether earlier states contextualize policies over later states (temporally forward,

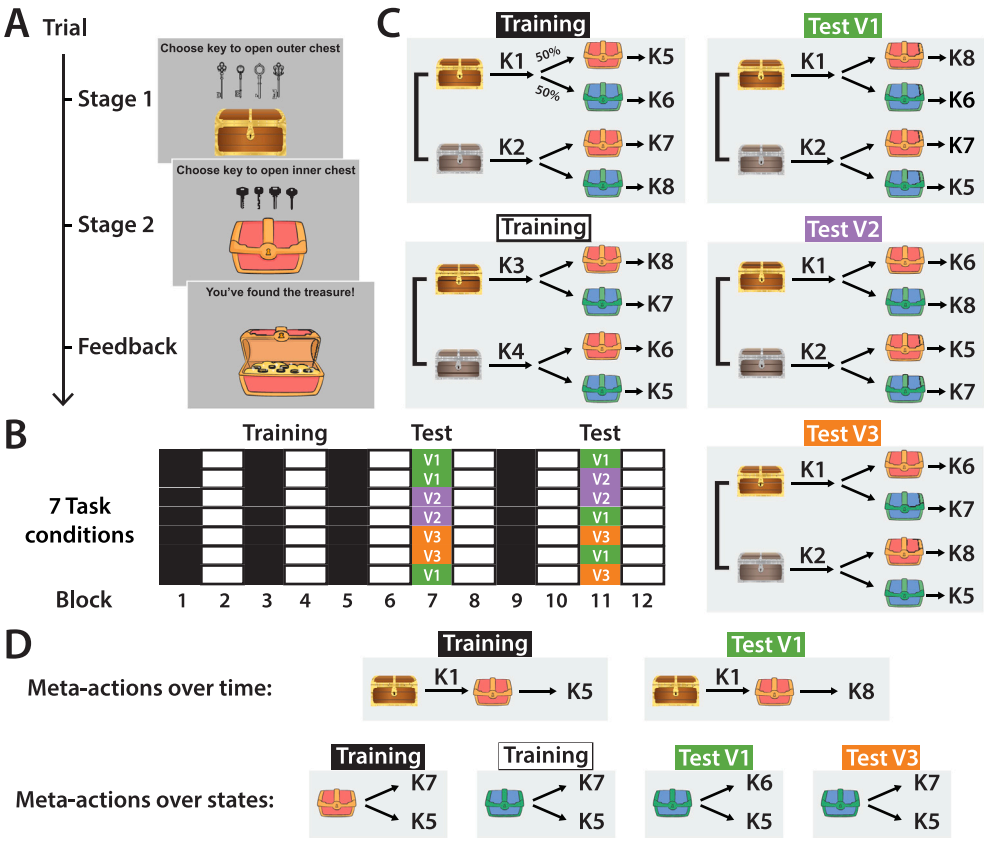

Fig. 2. The task paradigm. A: Participants learned to unlock two nested chests (gold/silver in stage 1 followed by red/blue in stage 2) by finding the correct keys through trial-and-error via deterministic feedback. They could only proceed to the next stage (pseudo-randomly determined) after selecting the correct key. B, C: The experiment consisted of 12 blocks, with 32–60 trials in the first two blocks and 32 trials in each following block. The hierarchically structured stimulus-action mapping changed every block without any explicit cues: the correct key to the inner chest depended on the outer chest's color and the block structure. In the training phase (Blocks 1–6), the block structure alternated between two hierarchical structures illustrated on the left. In each test block (Block 7 or 11), it switched to one of V1, V2, and V3, which are illustrated on the right. All three test block versions shared the stage 1 contingencies of the first training structure, with modified stage 2 contingencies designed to test how participants transfer and recompose knowledge. Seven different conditions defined by test block combinations between Blocks 7 and 11 were tested across participants. D: Actions can be chunked into meta-actions over time (from stage 1 to stage 2) or over states (stage 1 stimuli). Compared to the first training structure, the test structures included partially (V1) or fully (V2 and V3) different meta-actions over time; meta-actions over states learned in training were only preserved in V3, among all test structures. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

which was assumed by Xia & Collins, 2021) or vice versa (temporally backward) by fitting models with corresponding representation structures to human behavior. Coupling behavioral evidence with insights from computational cognitive modeling, we formulated two algorithmic architectures to account for how humans utilize compression to learn hierarchically structured decision policies, and how these structures are represented to enable efficient and flexible learning, transfer, and composition.

2. Methods

2.1. Task

We tested 1026 human participants on a decision-making task where they could learn, transfer, and recompose multiple sets of hierarchical policies (Fig. 2). Our experimental protocol extended our prior task (Li et al., 2022; Xia & Collins, 2021) to include a novel test structure and reduce interference between the action spaces of both stages (allowing us to test new hypotheses), and to use realistic stimuli (making the task more intuitive and engaging). A demo version of the experiment is provided in the "Data and code availability" section at the end of the manuscript. Each trial consisted of two stages, represented by two pairs of treasure chests (gold and silver in stage 1; red and blue in stage 2) that could be unlocked by two sets of four keys (denoted as K1–K4 and K5–K8; Fig. 2A). Participants learned the correct key for each chest through trial-and-error: they had to keep trying different keys until finding the correct one. Unlocking the chest in stage 1 led to stage 2, and unlocking the stage 2 chest led to positive feedback, followed by the next trial. The experiment included 12 blocks, spanning the training phase (Blocks 1–6), post-training tests (Blocks 7 and 11), and post-training control blocks (Fig. 2B). There were 32–60 trials (up to performance criteria) in Blocks 1–2 and 32 trials in each following block. Participants were not explicitly informed of the deterministic reward contingencies of the task. There were explicit block boundaries (an optional 20-second break between every two consecutive blocks), but no instructions or cues for whether the reward contingencies changed between blocks — participants had to infer these changes through trial and error; see task demo in "Data and code availability" for full instructions.

Five different hierarchically structured chest-key (stimulus-action) mappings were used in the experiment: two in training and control blocks that all participants experienced, and three in test blocks, denoted as V1, V2, and V3 (Fig. 2C), that were assigned across participants (see below). Compared to the first training structure, all three versions of test structures had the same stage 1 contingencies, with different degrees of similarity in stage 2: they shared different types of *meta-actions*. These test structures were carefully designed to break learned temporal and state abstractions in different ways to test how learning and transfer would be impacted. We define a meta-action as a pair of actions chunked together either over time or over states (Fig. 2D). A meta-action over time abstracts two atomic actions into a

Table 1
Comparison between test block versions.

Number of new meta-actions	V1	V2	V3
Over time	2	4	4
Over states	2	2	0

policy that specifies a compound action sequence spanning both stages (analogous to chunking the actions of preheating the oven and roasting the tomato in Fig. 1B); a meta-action over states abstracts two actions in stage 2 based only on the stage 2 stimulus, regardless of the stimulus (state) in stage 1 (analogous to chunking the actions of roasting and slicing the tomato in Fig. 1B). V1, but not V2 or V3, preserved two of the four meta-actions over time (K1-blue-K6 and K2-red-K7) in the first training structure; V3 preserved the meta-actions over states (K5–K7 and K6–K8) while V1 and V2 did not (Table 1). Every participant learned the same training and control blocks (Blocks 1–6, 8–10, and 12; colored black-and-white in Fig. 2B). The only difference between-participant was the task condition or test block combination: different groups of participants learned different test structure versions in the test blocks (Blocks 7 and 11), denoted by the test block combinations: V1-V1, V1-V2, V1-V3, V2-V1, V2-V2, V3-V1, and V3-V3.

Our task paradigm extends that used by Xia and Collins (2021) and Li et al. (2022) to enable the novel Test V3 design in addition to V1 and V2, which was not possible in the old paradigm due to its limited action space in stage 2 (the same four keys controlled by the right hand were shared by both stages and the correct action never repeated between stages). To achieve this, we separated the action spaces between stages (left hand for stage 1 and right hand for stage 2), which also allowed us to investigate the policies human participants used in stage 2 with minimal interference from their policies in stage 1. Additionally, our paradigm features more engaging stimuli and more logical relations (unlocking treasure chests with keys) than the old paradigm (associating geometric shapes with key presses). Additional details about the task paradigm are described in Supplementary Methods.

2.2. Data

All participants were recruited online through the Research Participation Program at the University of California, Berkeley. In total, 1026 participants completed the task across all conditions and received credit in eligible courses for their participation. Informed consent was obtained from all participants. The experiments were administered in two batches with four conditions each (all combinations of V1 and V3 in Experiment 1 and all combinations of V1 and V2 in Experiment 2). Since we did not find any significant differences in the shared V1-V1 condition, we combined the data of both experiments in our analyses and presentation where applicable, which is more thoroughly discussed in Supplementary Methods.

To group participants based on their learning strategies and identify poor performers to exclude from data analysis, we applied a clustering analysis on the trial-by-trial performance data using the k-means algorithm. Compared to conventional performance-based exclusion criteria adopted by previous work (Li et al., 2022; Xia & Collins, 2021), which classified participants into two classes (below- and above-chance performers), our approach could theoretically identify more than two performance patterns that might have been present in the data. We clustered participants into three groups based on their trial-by-trial performance in the training phase for each experiment, which is further justified in Supplementary Methods.

In all our analyses, performance was measured by the number of key presses made by the participant until reaching the correct choice in each stage, as they had to keep pressing different keys until reaching the correct one to proceed to the next stage (Fig. 3A). All results held qualitatively when we used the accuracy of the first key press in each stage as the performance metric instead. In stage 1, no explicit feedback was provided when a wrong key was pressed; upon a correct key press, the task transitioned into stage 2. In stage 2, a "wrong key" message was shown upon a wrong key press and an open chest filled with treasures was displayed when the correct key was selected. The lack of explicit reward signals in stage 1 maintained the hierarchical structure of the trial so that explicit reward signals were only observed at the end of each trial (Xia & Collins, 2021). The number of key presses quantified the amount of errors made by the participant in each stage: the more key presses, the more errors. Therefore, this metric is inversely correlated to how optimal the participant's policy was, making it an informative measure of task performance. Based on the observation that participants rarely repeated the same action on the same trial (the average frequency of repeating actions was 0.007 press per trial across all participants), we defined random performance level at 2.5 presses, which is the average of 1, 2, 3, and 4: if the participant did not make any errors, they would make 1 press; if they tried different keys without repeats, they would make up to 4 presses.

2.3. Models

We compared pairs of computational cognitive models by fitting them to human choice data to answer two questions about human learning. First, to understand how humans learn the hierarchical structures without prior knowledge about them, we tested the fully hierarchical model developed by Xia and Collins (2021) against a new meta-learning model, which incorporates the hierarchical model as one module. While the hierarchical model assumes full knowledge about the hierarchical policy structure from the beginning of learning, the meta model learns posterior probabilities of sampling two flat policies in addition to the hierarchical policy via Bayesian inference, which can capture gradual shifts from relying on flat, compressed policies to mastering more complex, hierarchical structures (illustrated in Fig. 4A). Second, to test the temporal order in which humans represent learned hierarchical structures, we compared the meta-learning model with temporally forward structures, in which earlier state information contextualizes policies over later state information, to the meta-learning model with temporally backward structures, in which later state information contextualizes policies over earlier state information (illustrated in Fig. 5A, B).

2.3.1. Hierarchical model

Algorithm 1 outlines the hierarchical model, which extends prior work (Xia & Collins, 2021) by representing lower-level policies using task-sets instead of options without changing model behavior. The benefits of this improvement were two-folded: first, it allowed us to derive analytical likelihoods and fit the model to trial-by-trial human choice data by optimizing likelihoods and obtain best-fitting parameters (contrary to Xia & Collins, 2021); second, it improved the computational efficiency of the model by reducing one level of abstraction. Since our current work focuses on modeling choices in stage 2 conditioning on choices in stage 1, stage 1 policies are skipped in our models. However, stage 1 policies can be modeled independently from stage 2 using a similar algorithm (Xia & Collins, 2021).

Each policy chunk is represented by a tabular task-set, denoted as TS, which stores the value of each action in each state, encoded by the identity of the stimulus in the chunked stage (stage 2 for the forward structure and stage 1 for the backward structure; Line 2). Task-sets are contextualized by a combination of the block number and the identity of the stimulus that serves as the context of the policy chunks (stage 1 for the forward structure and stage 2 for the backward structure; Line 2). The model tracks a repertoire of task-sets and learns the probability of sampling each task-set in each context, which is updated using Bayes' rule over learning. When a new context is encountered, the model creates a new task-set with uninformative values. The initialization of task-set priors in a new context follows a Chinese restaurant process

Algorithm 1 Fully hierarchical model

Require: Parameters $\theta = \{\eta, \beta, \alpha, \epsilon\}$

1: **for** $t = 1, 2, ..., T$ **do** ▷ Skip stage 1
2: Observe context c_t and state s_t
3: Compute alternative context in the same block c'_t
4: **if** c_t is new **then** ▷ New context
5: Initialize prior $\Pr(TS_i \mid c_t\,;\,t) \propto \sum_c \Pr(TS_i \mid c\,;\,t)$ for all TS_i
6: Create a new task-set TS_{c_t} with uninformative values
7: $\Pr(TS_{c_t} \mid c_t\,;\,t) = \frac{\alpha}{\alpha+1}$ ▷ CRP
8: Normalize $\Pr(TS_i \mid c_t\,;\,t) \leftarrow \frac{\Pr(TS_i|c_t\,;\,t)}{\alpha+1}$
9: **end if**
10: **while** $r_t = 0$ **do**
11: Add context-pairing bias based on c'_t to task-set priors for c_t
12: Sample a copy of TS_t based on $\Pr(TS_i \mid c_t\,;\,t)$ for all TS_i
13: **for** a that has been tried on this trial **do**
14: $TS_t(s_t, a\,;\,t) = -\infty$
15: **end for**
16: $\pi = \mathrm{softmax}\big(\beta \cdot TS_t(s_t, A\,;\,t)\big) \cdot (1-\epsilon) + \frac{1}{|A|} \cdot \epsilon$
17: Sample action $a_t \sim \pi$ and observe reward r_t
18: **if** $r_t = 1$ **then** ▷ Bayesian update
19: $\Pr(TS_i \mid c_t\,;\,t+1) \propto \Pr(TS_i \mid c_t\,;\,t) \cdot \pi(a_t)$ for all TS_i
20: **else**
21: $\Pr(TS_i \mid c_t\,;\,t+1) \propto \Pr(TS_i \mid c_t\,;\,t) \cdot \big(1 - \pi(a_t)\big)$ for all TS_i
22: **end if**
23: Re-sample TS_t based on $\Pr(TS_i \mid c_t\,;\,t+1)$ for all TS_i
24: $TS_t(s_t, a_t\,;\,t) \leftarrow TS_t(s_t, a_t\,;\,t) + \eta \cdot \big(r_t - TS_t(s_t, a_t\,;\,t)\big)$
25: **end while**
26: **end for**

(CRP) parameterized by α (Pitman, 2006): the task-sets that have been successful in previous contexts have higher priors than historically less successful task-sets, while the prior on the new task-set is determined by α (Lines 5–8). As a result, the model is able to reuse previously learned task-sets when a structure re-occurs and learn a new one for an unfamiliar structure, as illustrated in Figs. 5C, 6B, and 7A.

In addition to the current context, the model identifies an alternative context c', which is the context defined by the current block and the alternative state. A context-pairing bias is added to the task-set probabilities to model the affinity between task-sets that have co-occurred in the same block — this allows the model to leverage the task-set probabilities it has learned in the alternative context within the same block to inform task-set selection in the current context (Line 11). The bias b_{TS_i,TS_j} is proportional to the number of times two task-sets, TS_i and TS_j have been paired in the same block:

$$b_{TS_i,TS_j} \propto \sum_{\text{block}} \sum_{c,c'} \mathbb{1}_{TS_i = \mathrm{argmax}\,\Pr(TS|c\,;\,t)} \cdot \mathbb{1}_{TS_j = \mathrm{argmax}\,\Pr(TS|c'\,;\,t)},$$

where $\mathbb{1}_x$ is the indicator function which takes on the value 1 if x is true and 0 otherwise. The context-pairing bias is added to the task-set probabilities when the model is still learning the best task-set in the current context, or when the maximum task-set probability is less than 0.5, since a task-set probability higher than 0.5 implies that this task-set is deemed more likely than all others (Collins & Koechlin, 2012).

$$\Pr(TS_t \mid c_t\,;\,t) = w_b \cdot b_{TS_t,TS'_t} + (1 - w_b) \cdot \Pr(TS_t \mid c_t\,;\,t),$$

where $TS'_t = \mathrm{argmax}\,\Pr(TS \mid c'_t\,;\,t)$ is the most likely task-set in the alternative context c' and its probability is equal to the strength of the bias $w_b = \Pr(TS'_t \mid c'_t\,;\,t)$.

On each trial, in stage 2, the model samples a task-set according to the task-set probabilities in the current context (Line 23). It uses working memory to track actions that have been tried on this trial and avoids repeating them by de-valuing them (Line 14). Then, it picks an action by sampling from the softmax-transformed distribution of this task-set with a uniform decision noise (Lines 16–17):

$$\Pr(a \mid c_t, s_t, TS_t\,;\,t) = \frac{\exp\big(\beta \cdot TS_t(s_t, a\,;\,t)\big)}{\sum_{a' \in A} \exp\big(\beta \cdot TS_t(s_t, a'\,;\,t)\big)} \cdot (1-\epsilon) + \frac{1}{|A|} \cdot \epsilon,$$

where A is the action space (the set of all available actions) and ϵ is the uniform noise parameter. After acting with the sampled action a_t, the model observes reward r_t from the environment and updates its task-set belief probabilities (Lines 18–22) and the corresponding value in the re-sampled task-set (Line 24).

2.3.2. Meta-learning model

Algorithm 2 Meta-learning model

Require: Parameters $\theta = \{\eta, \beta, \alpha, \epsilon, p_h, w, \gamma\}$

1: Initialize policy priors $\Pr(\pi\,;\,0) = [(1-p_H)\cdot w, (1-p_H)\cdot(1-w), p_H]$ for $\pi = [\pi_{C_1}, \pi_{C_2}, \pi_H]$
2: **for** $t = 0, 1, ..., T-1$ **do** ▷ Skip stage 1
3: Observe context c_t and alternative context c'_t
4: Observe state s_t and alternative state s'_t
5: **if** c_t and c'_t are new **then**
6: Create a new task-set for each new context as in Algorithm 1
7: Initialize new task-set priors as in Algorithm 1 ▷ CRP
8: **end if**
9: **while** $r_t = 0$ **do**
10: Add context-pairing bias based on c'_t to task-set priors for c_t
11: Sample TS_t and TS'_t from priors given c_t and c'_t
12: De-value all actions that have been tried as in Algorithm 1
13: **if** backward **then**
14: $\pi_{C_1} = \mathrm{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS_t(s'_t, A\,;\,t)\big)/2\big)$
15: $\pi_{C_2} = \mathrm{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS'_t(s_t, A\,;\,t)\big)/2\big)$
16: **else**
17: $\pi_{C_1} = \mathrm{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS'_t(s_t, A\,;\,t)\big)/2\big)$
18: $\pi_{C_2} = \mathrm{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS_t(s'_t, A\,;\,t)\big)/2\big)$
19: **end if**
20: $\pi_H = \mathrm{softmax}\big(\beta \cdot TS_t(s_t, A\,;\,t)\big)$ ▷ Hierarchical policy
21: $N_\pi \leftarrow (1-\gamma)\cdot N_\pi + \gamma$ ▷ Forget
22: $N_\pi\big(a_t \mid c_t, s_t\big) \leftarrow N_\pi\big(a_t \mid c_t, s_t\big) + 1$ ▷ Update count
23: $L_\pi(a_t \mid c_t, s_t) = \frac{N_\pi(a_t|c_t,s_t)}{\sum_{a \in A} N_\pi(a|c_t,s_t)}$ ▷ Compute likelihood
24: $p_t(\pi) = \mathrm{softmax}\big(\beta_{\text{meta}} \cdot \Pr(\pi\,;\,t)\big)$
25: $\pi_m = \big(\sum_{i \in \{C_1, C_2, H\}} p_t(\pi_i) \cdot \pi_i\big) \cdot (1-\epsilon) + \frac{1}{|A|} \cdot \epsilon$ ▷ Meta-policy
26: Sample action $a_t \sim \pi_m$ and observe reward r_t
27: Update task-set belief probabilities as in Algorithm 1
28: **if** $r_t = 1$ **then** ▷ Bayesian update
29: $\Pr(\pi\,;\,t+1) \propto \Pr(\pi\,;\,t) \cdot L_\pi(a_t \mid c_t, s_t)$
30: **else**
31: $\Pr(\pi\,;\,t+1) \propto \Pr(\pi\,;\,t) \cdot \big(1 - L_\pi(a_t \mid c_t, s_t)\big)$
32: **end if**
33: Update value in resampled task-set in c_t as in Algorithm 1
34: **end while**
35: **end for**

The meta-learning model, outlined in Algorithm 2, extends the fully hierarchical model to include two compressed policies over states (Fig. 4A). To support the meta-learning mechanism, three additional parameters (p_H, w, and γ) are included in the model: p_H defines the prior probability of the hierarchical policy and w represents the weight of the compressed policy over stage 1 against the compressed policy over stage 2 (Line 1). During learning, these belief probabilities over policies are updated using Bayes' rule (Lines 28–32). The meta-learning model tracks the number of times each action is sampled given the state for each policy, denoted by $N_\pi(a \mid c, s)$ (Line 22), based on which the likelihood of sampling each policy is computed (Line 23). γ controls the forgetting rate of learned policy sampling frequency distributions for the meta-learning mechanism (Line 21).

The meta model implements an off-policy representation of the compressed policies — its target policy is hierarchical, though it computes the compressed policies by averaging over corresponding stages of the hierarchical policy at decision time (Lines 13–19). To combine these policies into a meta-policy, the model computes an average between them weighted by the softmax-transformed policy belief probabilities with a fixed inverse temperature $\beta_{meta} = 5$ (Lines 24–25).

2.3.3. Temporally forward and backward representations

To test alternative hypotheses about the temporal order of state representations in the learned hierarchical policies, we created two versions of the meta-learning model that implement the temporally forward and backward representations, respectively. Both models have the same complexity (numbers of parameters and computational mechanisms), and they only differ in how the repertoire of learned task-sets, or policy chunks, are contextualized or indexed. In Algorithm 2, the context c_t is defined by the identity of the stage 1 stimulus in the forward model and stage 2 stimulus in the backward model (Line 3), while the state s_t is defined by the identity of the stage 2 stimulus in the forward model and stage 1 stimulus in the backward model. The computation of compression over each stage is adjusted accordingly to the temporal representation order (Lines 13–19).

2.3.4. Model fitting and parameter estimation

All models were fitted at the participant level using maximum likelihood estimation with the global optimization function `differential_evolution` provided by the `optimize` module of the `SciPy` library in python. The optimization objective was the negative log-transformed likelihood of the data given the model parameters, abbreviated as "model likelihood." The model likelihood was computed based on the model algorithm, marginalizing over all task-sets whenever a task-set was sampled. For example, in Algorithm 1, the model likelihood on trial t was calculated as:

$$\mathbb{L}(a_t \mid c_t, s_t\,;\, t) = -\log \sum_i \Pr(TS_i \mid c_t\,;\, t) \cdot TS_i(s_t, a_t\,;\, t).$$

2.3.5. Model comparison

We compared pairs of models using the Akaike Information Criterion (AIC) when model complexity differed, and the log-transformed model likelihood otherwise. Additionally, we validated the fitted models against human behavior by re-simulating choice data with the fitted parameters.

3. Results

3.1. Human learning performance

Observed human behavior qualitatively replicated findings in prior studies where applicable (Li et al., 2022; Xia & Collins, 2021). Specifically, the error rates and types were qualitatively comparable to previous results obtained under the old paradigm during the training phase and test V1 and V2 blocks. Same as Li et al. (2022), we found no significant differences between V1 and V2 in either test block, despite transfer effects between Blocks 7 and 11 (V1-V2 and V2-V1). Quantitative comparisons could not be drawn due to differences in the available actions in stage 2 between the new and old paradigms (in the old paradigm, the available actions in stage 2 were the three incorrect actions in stage 1, while in our paradigm, they are four actions specific to stage 2).

Using an unsupervised k-means clustering algorithm, we divided the PCA-transformed individual learning curves over Blocks 1–6 (number of presses in each stage on each trial of the training phase; see Methods) into three groups (Fig. 3B): random performers (n=181), mid performers (n=254), and best performers (n=591). The learning patterns of these clusters were distinct: the random performers did not learn to

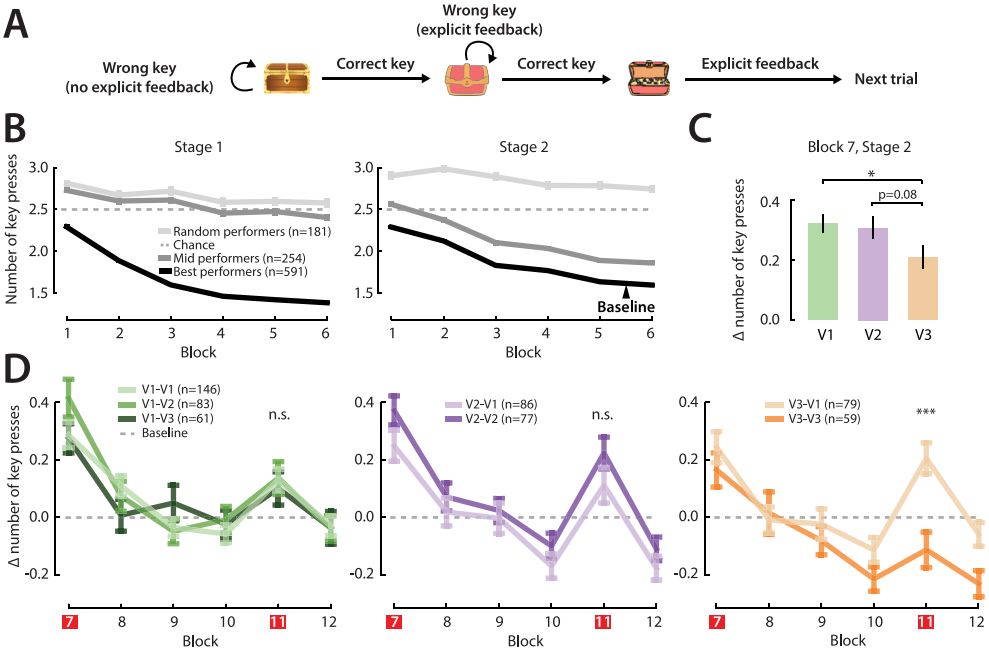

Fig. 3. Summary of human behavior. A: Task performance was measured by the number of key presses the participant made until reaching the correct choice in each stage, averaged over the first 10 trials of each block. On each trial, the participant had to press the correct key to advance from stage 1 to stage 2 (without explicit wrong key feedback) or from stage 2 to the next trial (with a "wrong key" message when a wrong key was selected). The number of key presses made in each stage quantified the amount of errors in the participant's choices and, by proxy, the deviation between their policy and the true task structure. B: The learning curves of all participants were grouped by an unsupervised k-means algorithm into 3 clusters, which corresponded to whether the participants learned to perform better than randomly guessing without replacement (chance) in both stages: the random performers (n=181) were no better than chance in either stage, indicating low task involvement, the mid performers (n=254) were only better than chance in stage 2, and the best performers (n=591) learned to perform better than chance in both stages. C: Across conditions in stage 2, humans made more key presses on average in the first test block (Block 7) compared to the end-of-training baseline (average between Blocks 5 and 6), with significantly less increase in V3 than V1. D: Post-training learning curves for all test conditions. In all figures, we denote $p < 0.001$ as ***, $p < 0.01$ as **, $p < 0.05$ as *, and $p > 0.05$ as n.s. All error bars represent one standard error of the mean.

perform better than chance in the training phase, the mid performers only exceeded chance performance in stage 2 but not stage 1, and the best performers learned to perform better than chance in both stages. Although both the mid and best performers learned stage 2, the mid performers made consistently more error (around 0.2 more key press per trial) than the best performers, suggesting that effectively learning stage 1 facilitated learning in stage 2. In the main text, we will only show results of the best performers. Corresponding results of the mid performers were qualitatively consistent with the main conclusions and included in Supplementary Figures. The random performers were excluded from any further analyses.

Human participants made more errors in the first test block (Block 7) than the end-of-training baseline, which was defined as the average performance on the first 10 trials of Blocks 5 and 6, in all three versions, replicating the negative transfer effect of previously learned policies shown by Xia and Collins (2021) (Fig. 3C for best performers, Fig. B.1B for mid performers). Notably, the increase in the number of key presses from baseline in V3 was significantly lower than in V1, and lower than in V2 on average although not statistically significant (two-tailed t-test t=2.1 and p = 0.037 between V1 and V3 and t=1.8 and p = 0.078 between V2 and V3). The significant difference was only present in one of the two V1 samples of both experiments when tested independently (t=1.0 and p = 0.32 for Experiment 1 with n=68, and t=2.5 and p = 0.013 for Experiment 2 with n=78). This weak discrepancy might suggest that V3's consistent meta-actions over states (K6–K8 and K5–K7) with the training structures have facilitated new learning and transfer. On the other hand, due to the lack of discrepancy between V1 and V2 (two-tailed t-test t=0.28 and p = 0.78), there was no evidence that preserving meta-actions over time (K1–K6 and K2–K7) facilitated transfer, which replicated the findings of Li et al. (2022). Taken together, these results imply that human behavior was more strongly driven by action chunking over states than over time.

Performance improved between both test blocks with repeating test structures (two-tailed t-test t=3.3 and p = 1.3×10^{-3} for V1-V1, t =2.5 and p = 0.015 for V2-V2, and t =5.6 and p = 6.4×10^{-7} for V3-V3), indicating transfer of learned structures (Fig. 3D for best performers, Fig. B.1A for mid performers). Interestingly, performance in the second test block was not significantly different between non-repeating and repeating combinations of V1 and V2 (two-tailed t-test t = −0.090 and p = 0.93 between V1-V1 and V1-V2, and t = −1.3 and p = 0.18 between V2-V1 and V2-V2), while it was significantly worse in V3-V1 than V3-V3 (two-tailed t-test t =3.9 and p = 1.6×10^{-4}). These results suggest that some transfer of learned structures might have occurred between V1 and V2, since participants showed similar amounts of performance improvement between test blocks when V1 was followed by V1 compared to V2, as well as when V2 was followed by V2 compared to V1. However, no transfer was observed between V1 and V3 since performance only improved in V3-V3 but not V3-V1.

3.2. Building a mechanistic understanding of the state and temporal abstractions underlying human choice behavior

The behavioral analyses above presents evidence for compressed action representations over states, which imply state abstractions. However, we lack a mechanistic understanding of how these abstractions contribute to learning. How do state abstractions change over learning? What is their role in the construction of hierarchically structured policies that eventually drive behavior? In this subsection, we introduce an algorithmic account for how compressed policies over states unfold into hierarchical policies and how the temporal structure of state abstractions gives rise to efficient transfer by enabling compositionality. Using cognitive process models, we show that our framework can explain the error patterns in human behavior, which alternative accounts fail to.

3.2.1. Modeling the emergence of hierarchical policies

Compressed policies can form over the state space of either stage 1 or stage 2 (Fig. 4A). In a fully compressed policy over stage 1, the state in stage 2 is represented independently from stage 1 stimulus (e.g., gold-red and silver-red are compressed into red), whereas a fully compressed policy over stage 2 assumes that the state is independent from stage 2 stimulus (e.g., gold-red and gold-blue are compressed into gold). Compressing over states in different ways leads to distinct meta-actions (in the first training structure, K5–K7 and K6–K8 if compressed over stage 1, and K5–K6 and K7–K8 if compressed over stage 2), which result in different wrong choices (e.g., K7 or K6 instead of K5). Thus, choices made on each trial can be classified into four types: correct choices, compression over stage 1 errors, compression over stage 2 errors, and other errors (example illustrated in Fig. 4B). In Block 1, compression errors, especially over stage 1, dominated the wrong choices made by humans on the first attempt of each trial (Fig. 4C for best performers, Fig. B.1C for mid performers), suggesting that humans relied on compressed policies in early learning.

Based on this observation, we hypothesized that humans developed compressed policies into hierarchically structured policies over learning. To test this hypothesis, we fitted two models to human choice data: a fully hierarchical model that learned hierarchical policies without compression, and a meta-learning model that learned posterior probabilities of all compressed and hierarchical policies using Bayesian inference (Fig. 4A). The meta model started learning with a parameterized prior on the hierarchical structure; its priors on the compressed structures were determined by another fitted parameter. It fitted to human choices significantly better (two-tailed t-test t = 24, p< 10^{-4}) than the fully hierarchical model based on the AIC (Fig. B.6A). Most parameters were recoverable, despite noisy recovery of the forgetting rate γ and some value ranges of the learning rate η and hierarchical prior p_H (Fig. B.6B). The meta model successfully produced the error distribution in early human choices, while the hierarchical model failed to, both at the group level (Fig. 4C) and at the individual level (Fig. B.6C). Due to the complexity of our models, we do not aim to explain nuanced individual learning differences; rather, the focus of our analyses is to capture qualitative group behavioral patterns. The meta model predicted that humans started learning with a higher preference for over-simplified, compressed policies, and slowly switched to making choices based on hierarchical policies instead (Fig. 4D), exceeding the ceiling performance of the compressed policies (Fig. B.7) or a mixture of the three policies without the Bayesian learning process (Fig. B.8). This gradual meta-learning process allowed the meta model to reproduce error patterns in human choices that drove learning, particularly the imbalance in error types and the decrease in compression errors throughout learning (Fig. 4E for best performers, Fig. B.1D for mid performers).

3.2.2. Modeling the representation structure of hierarchical policies

Building on the meta-learning of hierarchically structured policies, we further investigated how these policies were represented and how state abstractions shaped these representations. Compressed policies over states can unfold into hierarchical policies in two ways, depending on which type of state abstractions dominates: compressed policies over stage 2 use the stage 1 stimulus (gold/silver) to contextualize action chunks, while compressed policies over stage 1 use the stage 2 stimulus (red/blue). These action chunks can expand into policy chunks by incorporating the stimulus in the compressed stage (Fig. 5A, B). In the former case, the hierarchy is constructed in a *temporally forward* manner, where earlier information (stage 1 stimulus) contextualizes policy chunks defined on later information (stage 2 stimulus). On the contrary, when later information (stage 2) contextualizes policy chunks defined on earlier information (stage 1), the hierarchy follows a *temporally backward* organization. By design, V3 preserved the policy chunks learned in training if the hierarchical policy structures were temporally backward, but not if they were temporally forward: policy

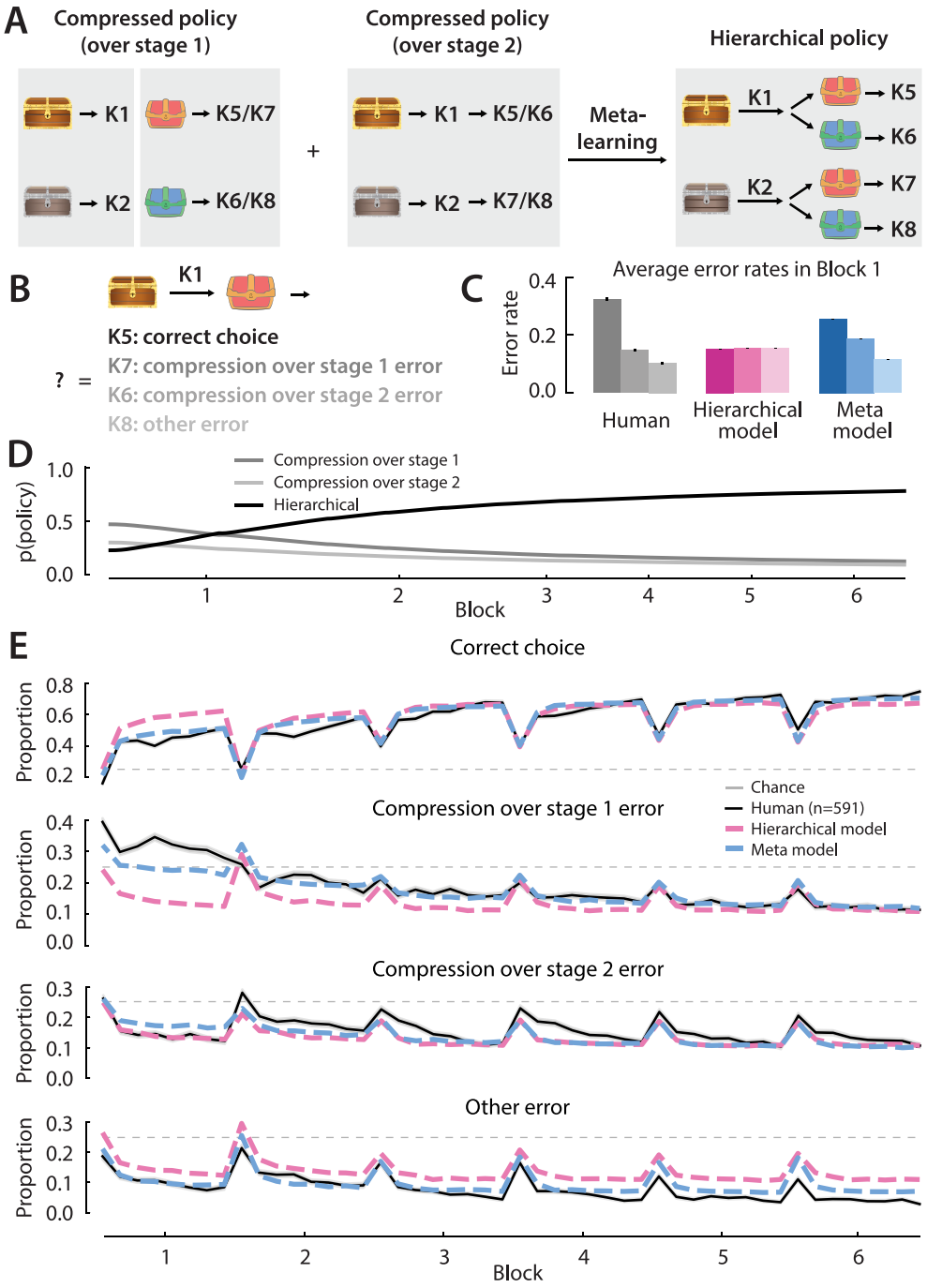

Fig. 4. Compressed policies unfolded into hierarchical policies via meta-learning. A: We modeled two types of compressed policies and a hierarchical policy, which used information from different stages to choose in stage 2. The policy that was compressed over stage 1 disregarded information in stage 1 when sampling an action in stage 2 — its stage 2 policy depended solely on the stage 2 stimulus. On the other hand, the compressed policy over stage 2 depended solely on stage 1 to act in stage 2. The hierarchical policy was optimal for the task structure — it used both stages to inform action selection in stage 2. We compared a fully hierarchical model (i.e., the hierarchical policy) to a meta-learning model that used Bayesian inference to learn the probability of sampling each of the three policies. B: Choices in stage 2 can be classified into 4 types: correct choices, compression over stage 1 errors (indicating stage 1 was disregarded), compression over stage 2 errors (indicating stage 2 was disregarded), and other errors. C: In early learning (Block 1), wrong presses made by humans were dominated by compression errors, especially compression over stage 1 errors, which was explained by the meta model but not the hierarchical model. D: The meta model's learned policy probabilities over training indicated a shift from favoring compressed policies to relying on the hierarchical policy. E: The qualitative error patterns in human learning were successfully captured by the meta model but not the hierarchical model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

chunks learned during training (the second chunk from each training structure, whose borders are highlighted in Fig. 5A) could be composed to solve V3 only if the structures were temporally backward.

As a result, upon encountering V3, a model that implemented temporally forward structures created a new set of policy chunks to represent the hierarchical policy, while a backward structured model

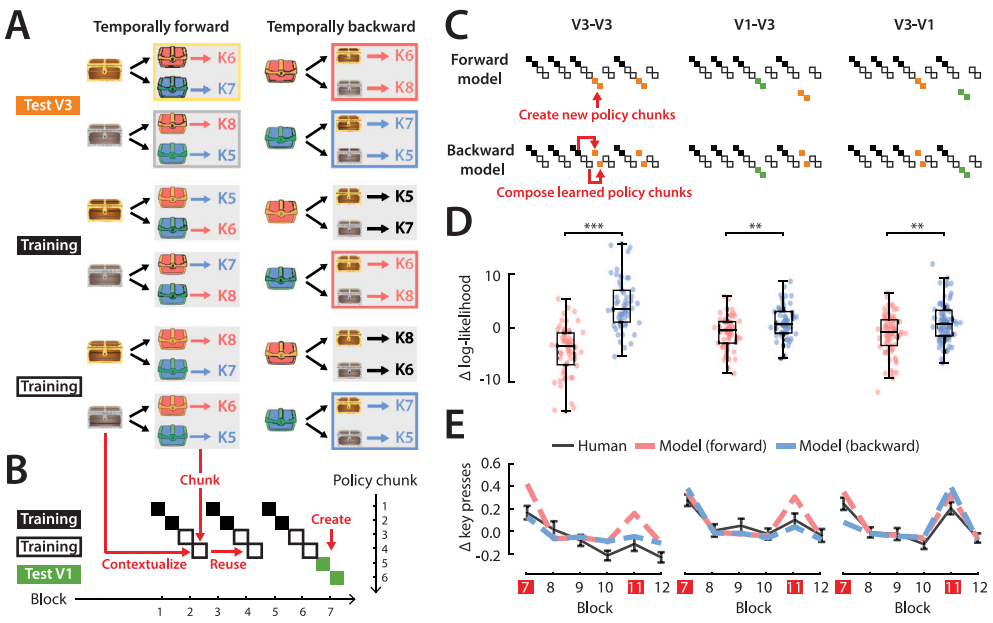

Fig. 5. The learned hierarchical policies in stage 2 followed a temporally backward structure. A: Temporally forward structures use earlier information (stage 1) to contextualize policy chunks defined over later information (stage 2), while temporally backward structures use later information (stage 2) to contextualize policy chunks over earlier information (stage 1). Temporally backward structures enable the policy for V3 to be composed by re-contextualizing chunks learned during training, while temporally forward structures do not allow such composition. We tested two models with temporally forward and backward structures, respectively. B: Both models learned two policy chunks to represent each block context, which could be reused between blocks. The models could create new policy chunks upon a new block structure. C: The forward model created new policy chunks upon first learning V3, while the backward model efficiently recomposed learned policy chunks to represent its V3 policy. D: Fitted to human choice data, the backward model had significantly higher likelihoods than the forward model in all three test block combinations containing V3. E: Overall, the backward model captured human behavior better than the forward model, particularly in the test blocks (Blocks 7 and 11).

flexibly composed learned policy chunks (Fig. 5C). All visualizations similar to Fig. 5C in this manuscript illustrate the most likely policy chunk (task-set) selections made in model simulations with best-fit parameters to human data on the last trial of each block. This enabled the backward model to learn and transfer structures more efficiently than the forward model in the V3-V3, V1-V3, and V3-V1 conditions. When fitted to human choice data, the backward model produced higher likelihoods than the forward model in all conditions (Fig. 5D; two-tailed t-test $t = -7.1$ and $p = 2.4 \times 10^{-9}$ for V3-V3, $t = -2.7$ and $p = 9.9 \times 10^{-3}$ for V1-V3, and $t = -2.7$ and $p = 7.9 \times 10^{-3}$ for V3-V1). The better fit to human behavior of the backward model also manifested in its ability to capture the qualitative pattern that humans were less prone to errors (they make fewer presses) when learning V3 than V1 (Fig. 5E). Consistent results were observed in the mid performers (Fig. B.2).

In addition to the faster learning of V3, temporally backward structures could also account for the transfer between V1 and V2 observed in human behavior (Fig. 3D). When viewed as temporally backward, the hierarchical structures of V1 and V2 shared the same policy chunks, which was not true for temporally forward structures (Fig. 6A). Therefore, the backward model could compose learned policy chunks between V1 and V2, while the forward model needed to create a new

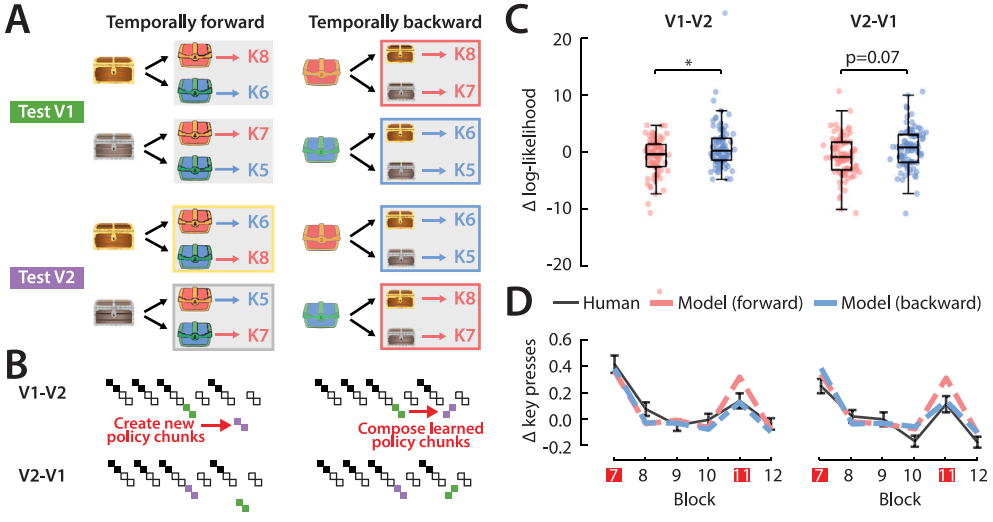

Fig. 6. Temporally backward structures enabled efficient composition between V1 and V2. A: Temporally backward structures allowed for recomposition of learned policy chunks between V1 and V2, while temporally forward structures did not. B, C: The forward model created two sets of structures to represent V1 and V2, while the backward model could efficiently compose one set of policy chunks to represent both. D: Fitted to human choice data, the backward model had higher likelihoods than the forward model in both V1-V2 and V2-V1. E: The backward model reproduced the performance improvement between V1 and V2 (in Blocks 7 and 11), which the forward model failed to.

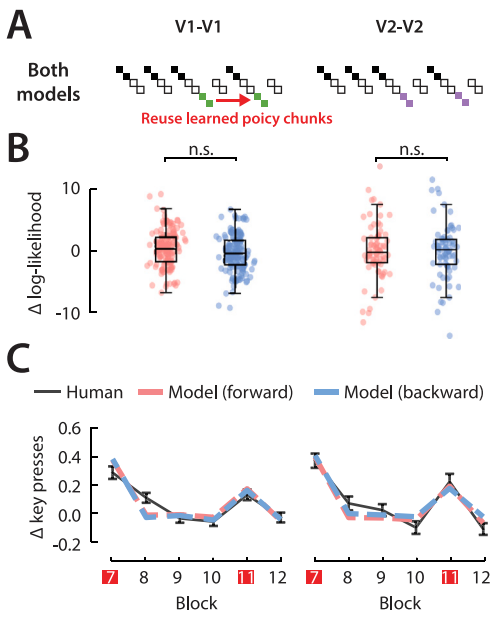

Fig. 7. The forward and backward models both successfully captured human behavior in repeating V1 and V2 test conditions. A: Both models created new policy chunks upon V1 or V2, which were reused when the same test structure repeated. B: The likelihoods of the forward and backward models were not significantly different. C: Both models reproduced the performance improvement between repeating V1 and V2 blocks.

set of policy chunks upon learning each test structure (Fig. 6B). The backward model fitted human choices better in terms of both the likelihood metric (Fig. 6C) and capturing the qualitative pattern of performance improvement between V1 and V2 in human behavior (Fig. 6D). The backward model also matched human behavior better qualitatively in the mid performers, though there were no significant differences in model likelihoods (Fig. B.2).

The forward and backward models generated indistinguishable predictions on V1-V1 and V2-V2, where both models created new policy chunks upon learning the test block for the first time and reused them when the test block repeated (Fig. 7A). When fitted to human data, the forward and backward models were indistinguishable by the likelihood metric (Fig. 7B). Both models successfully accounted for the performance improvement between test blocks in V1-V1 and V2-V2.

4. Discussion

Our findings highlight processes at multiple levels of abstraction that support the acquisition and representation of hierarchically structured decision policies in a complex, dynamic learning environment. We characterize the important role of state abstractions as building blocks of hierarchical policies and show how their temporal structure enables efficient learning, transfer, and composition.

Our computational framework, backed by data, provides a compelling algorithmic account for the slow, bottom-up construction process of hierarchical policies: simpler, compressed policies serve to bootstrap complex, hierarchical structures, which emerge incrementally through meta-learning. This theory bridges our understanding of compression and hierarchy via a mechanistic description of how the trade-off between representation efficiency and reward evolves over trial-by-trial learning. Our results suggest that humans prioritize representing policies efficiently through compression in early learning, which gradually unfold into more computationally expensive structures that maximize reward rate. In terms of the real-life example illustrated in Fig. 1, our theory would predict that people focus on learning smaller policies for individual cooking steps first, like slicing tomatoes, and then build them up into larger, hierarchical recipes. This meta-cognitive process is resource-rational (Gershman, Horvitz, & Tenenbaum, 2015; Lieder & Griffiths, 2020; Simon, 1955): under the constraint of limited cognitive resources, humans trade off rewards with the effortful task of structure learning. Notably, our analysis showed that humans may spend up to around 100 trials to fully learn hierarchical task structures (Fig. 4D), which suggests that overlooking this meta-learning process in computational modeling may confound the interpretation of choice behavior in tasks with complex structures.

Contrary to our expectations based on previous work (Botvinick et al., 2009; Xia & Collins, 2021), the structures learned by humans to represent hierarchical policies appear to be temporally backward rather than forward: the immediate information before decision-making (stage 2 stimulus) contextualizes a policy over earlier information held in memory (stage 1 stimulus). In our running tomato example (Fig. 1), the temporally backward policy representation would support a decision-making process where the person thinks back to the action they performed previously (preheating the oven or slicing sandwich bread) before choosing an action (roast or slice) on the tomato. By contrast, the more standard "forward" hierarchical reinforcement learning option model would instead assume that participants pre-load the "roasting" policy as they preheat the oven, such that they already know what to do with the tomato at the next stage. Although both temporally forward and backward structures can be flexibly transferred and composed to facilitate new learning, a temporally backward one may be more resource-rational, since it allows hierarchy to emerge without the effortful process of re-contextualizing compressed policies. This temporally backward structural organization is a departure from the standard options framework in hierarchical reinforcement learning, which implies the opposite (temporally forward) representation structure (Botvinick et al., 2009; Sutton et al., 1999).

Moreover, the temporally backward decision process reveals the integral role of working memory in forming and executing hierarchical policies: the earlier information (stage 1 stimulus) is held in memory until decision time, when the later information that contextualizes the trial (stage 2 stimulus) is observed. Here, working memory connects different levels of abstractions over time, allowing policy information at multiple timescales to be integrated to guide decision-making. This novel insight corroborates the perspective that working memory and reinforcement learning are intertwined processes that facilitate each other and should not be considered separately (Collins & Frank, 2012; Yoo & Collins, 2022). Future research should explore applications of temporally backward structures with a working memory mechanism to solve hierarchical reinforcement learning problems in artificial intelligence.

Another potential extension is to apply our framework to model hierarchical policies and abstractions in goal-directed decision-making and planning (Daw, Gershman, Seymour, Dayan, & Dolan, 2011; Molinaro & Collins, 2023), in which the agent additionally learns the transition structure of the states and uses this information to choose actions to achieve specific outcomes. As a temporally forward process, would planning strengthen temporally forward hierarchical representations and inhibit temporally backward ones? Would it impact the roles of temporal and state abstractions in forming hierarchical policies? Since our current task paradigm focuses on investigating abstractions, all state transitions in the task are unstructured (i.e., random) to minimize potential confounds. Building on the algorithmic framework developed in our current work, incorporating planning in the decision-making process may help us gain a more complete understanding of how abstractions support learning and decision-making in real life.

Future work should also aim to replicate our main findings on samples that are more representative of the general human population to evaluate the generalizability of our results to human learning overall. Although we collected data on a large number of participants (n = 1026), the population we sampled from was limited to students enrolled in college-level courses at a specific university, which might have introduced bias in our results. However, we note that the old

task paradigm with similar training but different test conditions was replicated on Amazon Mechanical Turk, which is more representative of humans in general (Xia & Collins, 2021). Despite the potential limitations imposed by our participant population, our main findings were valid on the majority of our sample (75% of all participants including top and mid performers), which implies promising generalizability to a broader human population.

5. Conclusions

Our algorithmic framework characterizes how the interplay between various cognitive processes supports structured decision-making, including reinforcement learning, policy compression, meta-learning, and working memory. We emphasize the important contributions of state abstractions in forming hierarchical policies and challenge the conventional conception of temporal abstractions by introducing the novel temporally backward structure. These algorithmic architectures serve as backbones of compositionality, enabling humans to efficiently and flexibly generalize knowledge between related tasks — a hallmark of human intelligence. We hope our work will inspire and inform the development of machine learning architectures that can learn and generalize like humans.

Supplementary information

See Supplementary Methods for supplementary methods and Supplementary Figures for supplementary figures.

CRediT authorship contribution statement

Jing-Jing Li: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Anne G.E. Collins:** Writing – review & editing, Supervision, Resources, Methodology, Funding acquisition, Conceptualization.

Data and code availability

All data and code used to produce results and figures in this manuscript can be found at: https://github.com/jl3676/learning_hierarchy. A demo of the full online behavioral task used for data collection is available at: https://experiments-ccn.berkeley.edu/learning_hierarchy_task_demo/exp.html?id=demo.

Acknowledgements

This work was supported by the NIMH grant R01MH119383. We thank Liyu Xia for helpful discussions throughout the project.

Appendix A. Supplementary Methods

A.1. Task

The experiment consisted of 12 blocks, with an optional 20-second break between every two consecutive blocks. The first two blocks had a minimum of 32 and a maximum of 60 trials: after completing 32 trials, participants skipped ahead to the next block as soon as they reached the criterion of less than 1.5 key presses per trial in each stage averaged over the past 10 trials. All other blocks included 32 randomly ordered trials with 8 trials for each combination of stage 1 and stage 2 stimuli pair. The trials were pseudo-randomly ordered such that in each block, there were never more than three consecutive iterations of the same stimulus in stage 1. Furthermore, among the trials that had the same stage 1 stimulus in each block, there were never more than three consecutive iterations of the same stimulus in stage 2. In each stage, participants were instructed to choose from one of four keys on their keyboards (Q, W, E, and R for stage 1, and U, I, O, and P for stage 2). When an invalid key was selected, a feedback message was shown to remind the participant which four keys to choose from. The experiment only advanced to the next stage upon a correct key press or until 10 key presses had been made in the current stage. In stage 1, when a wrong key was pressed, no explicit feedback was shown (the stimulus remained the same), and upon a correct key press, the experiment transitioned into stage 2; in stage 2, a wrong key press triggered a "wrong key" message, while a correct key press led to positive feedback (unlocked chest filled with gold; Fig. 3A).

Table A.2
Number of participants by experiment, condition, and cluster.

Cluster	Experiment 1				Experiment 2			
	V1-V1	V1-V3	V3-V1	V3-V3	V1-V1	V1-V2	V2-V1	V2-V2
Best	68	61	79	59	78	83	86	77
Mid	27	35	23	30	36	39	37	27
Random	17	20	15	23	22	23	30	31
All	112	116	117	112	136	145	153	135

A.2. Data

A.2.1. Data collection

The experiments were administered in two batches: Experiment 1 tested all combination conditions of V1 and V3, while Experiment 2 included all combinations of V1 and V2. Experiment 1 was conducted between September 2022 and March 2023, and Experiment 2 between September 2023 and December 2023. Each participant only completed the experiment once and all participants of Experiment 1 were excluded from the recruitment for Experiment 2. After inspecting the results of Experiment 1, which differed from previous work (Li et al., 2022) in not only the test conditions, but also the stimuli and action spaces, we additionally conducted Experiment 2, which featured the same test conditions as those adopted by Li et al. (2022), to replicate previous results using the updated task paradigm. All participants of Li et al. (2022) were excluded from the recruitment for both our experiments.

Table A.2 contains the number of participants per condition and experiment, divided by performance-based cluster assignments. We thoroughly compared the V1-V1 data between both experiments and found no significant differences in performance (Fig. B.3) or error types, which indicated that there were no external factors driving any learning differences between experiments. Additionally, we did not find any qualitative differences between our V1-V1 data and those collected by Li et al. (2022), despite the different stimuli and action spaces between the task paradigms. Therefore, data from Experiments 1 and 2 were combined in all analyses.

A.2.2. Participant clustering

To group participants based on their task involvement in a data-driven way, we performed unsupervised clustering on the trial-by-trial performance data in the training phase. For each participant, we used a feature vector containing the numbers of key presses in both stages on the first 32 trials of all training blocks (384 dimensions in total). We performed k-means clustering on the first 10 principal components of these features, which was a generous and safe number since the clustering results were primarily driven by the first 2 principal components and cluster boundaries did not change when we used fewer or more PCs between 5 and 10 (Fig. B.4). We set $k = 3$ to maximized the interpretability of group average learning behavior: the best performers exceeded chance performance in both stages, the mid performers only performed better than chance in stage 2, and the random performers did not perform better than chance in either stage (Fig. 3B). When $k = 2$, the random and mid performers were combined into one cluster, while when $k = 4$, the top performers were broken into two smaller clusters without apparent qualitative learning differences (Fig. B.5).

Appendix B. Supplementary Figures

(see Figs. B.1–B.8)

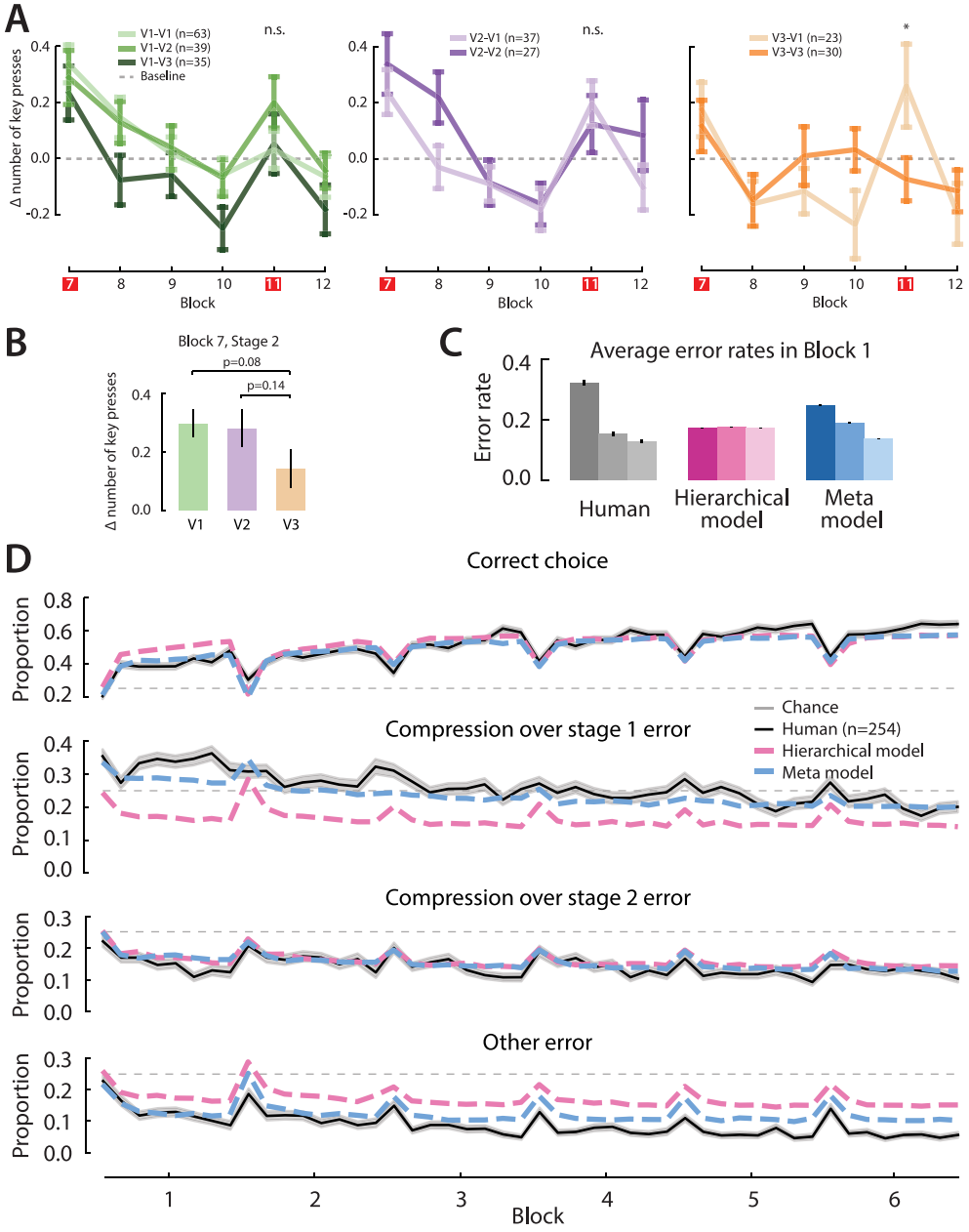

Fig. B.1. Learning behavior of the mid performers cluster. A: Post-training learning curves by test condition. B: Human performance was better in the first test block of V3 than V1 and V2. C: The meta model captured the error rate patterns in early learning of humans, which the fully hierarchical model failed to. D: Human error curves in training were better matched by the meta model than the hierarchical model.

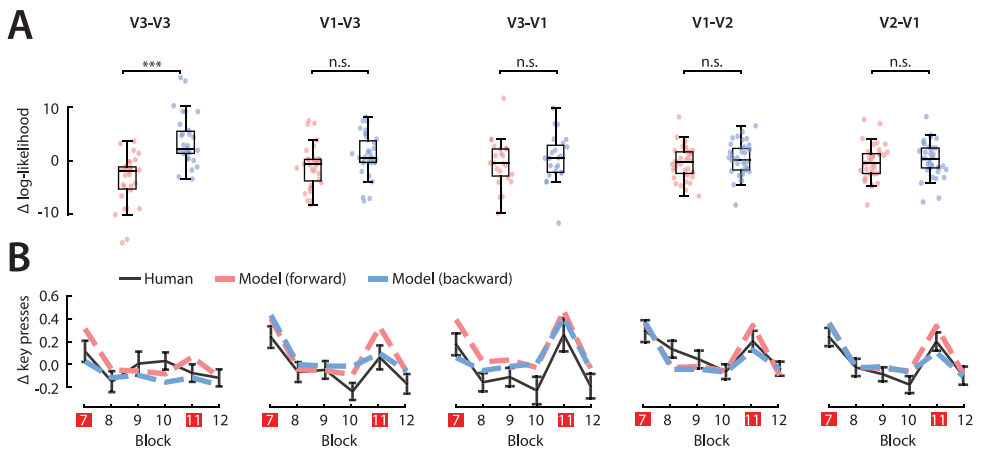

Fig. B.2. Comparisons of the temporally forward and backward models on mid performer data. A: In V3-V3 and V1-V3, the backward model had significantly higher likelihoods than the forward model, while the models did not produce significantly different likelihoods in V3-V1, V1-V2, and V2-V1. B: Overall, the backward model captured qualitative patterns in human learning performance better than the forward model.

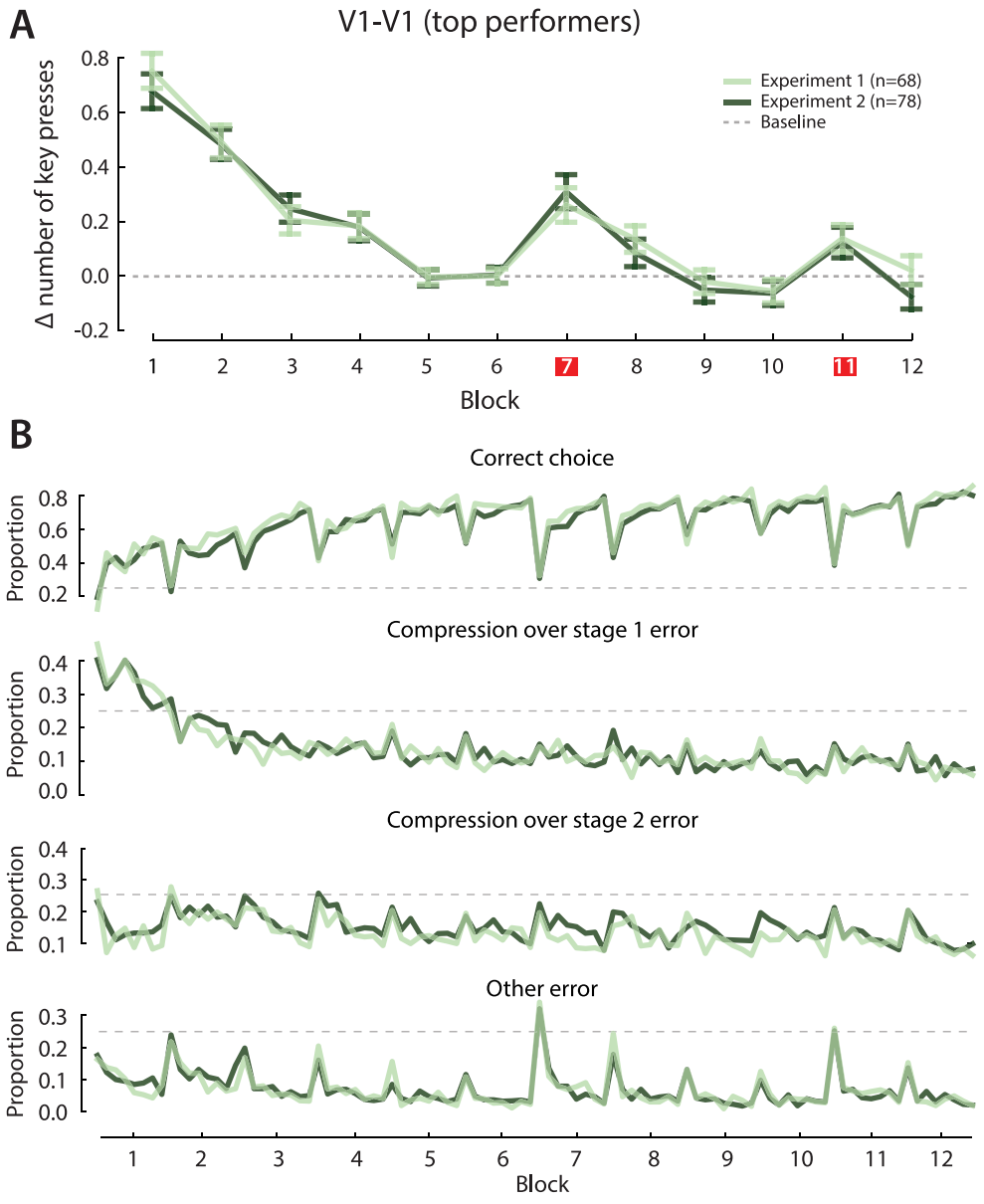

Fig. B.3. Comparing V1-V1 data between Experiment 1 and Experiment 2 within the top performers. A: The normalized average number of key presses in stage 2 across the first 10 trials of each block. B: The learning curves by choice type in the training phase.

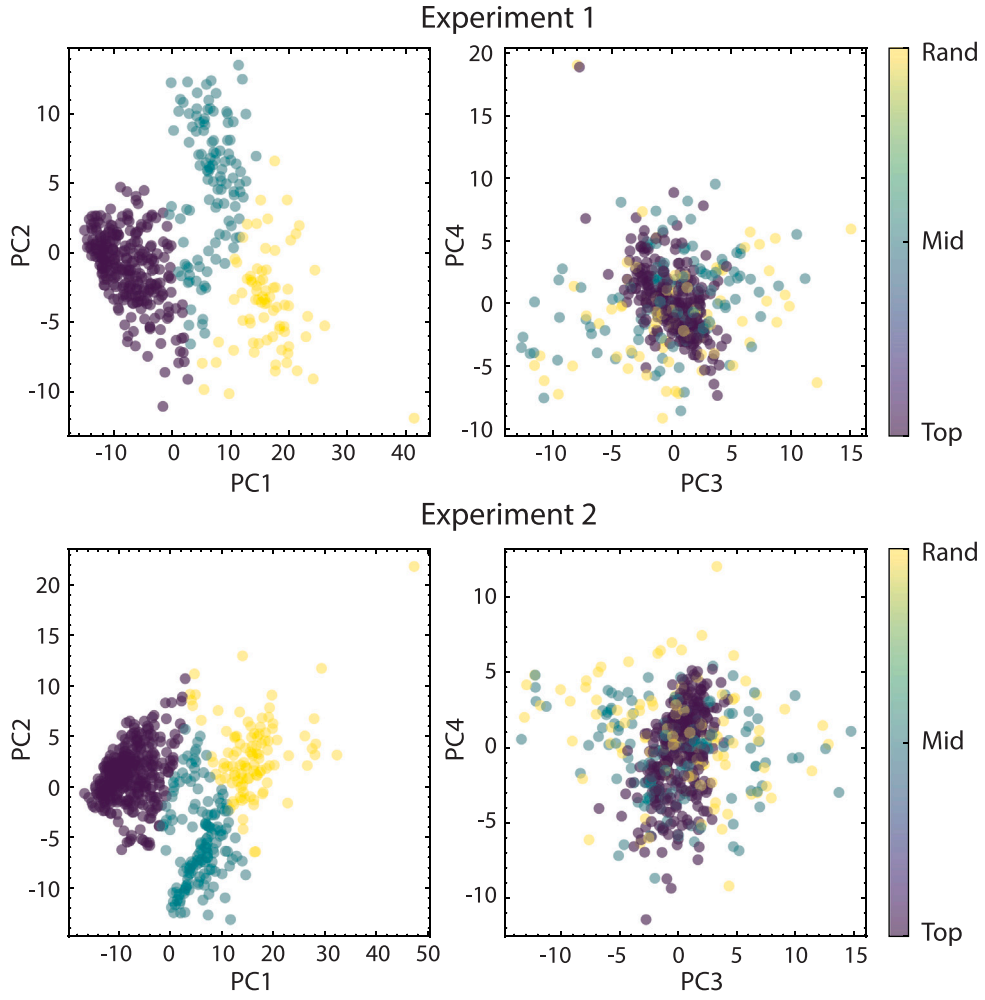

Fig. B.4. Cluster assignments of individual participants projected onto different principal components in both experiments. The first two principal components were substantially more informative in separating the clusters than components 3 and 4.

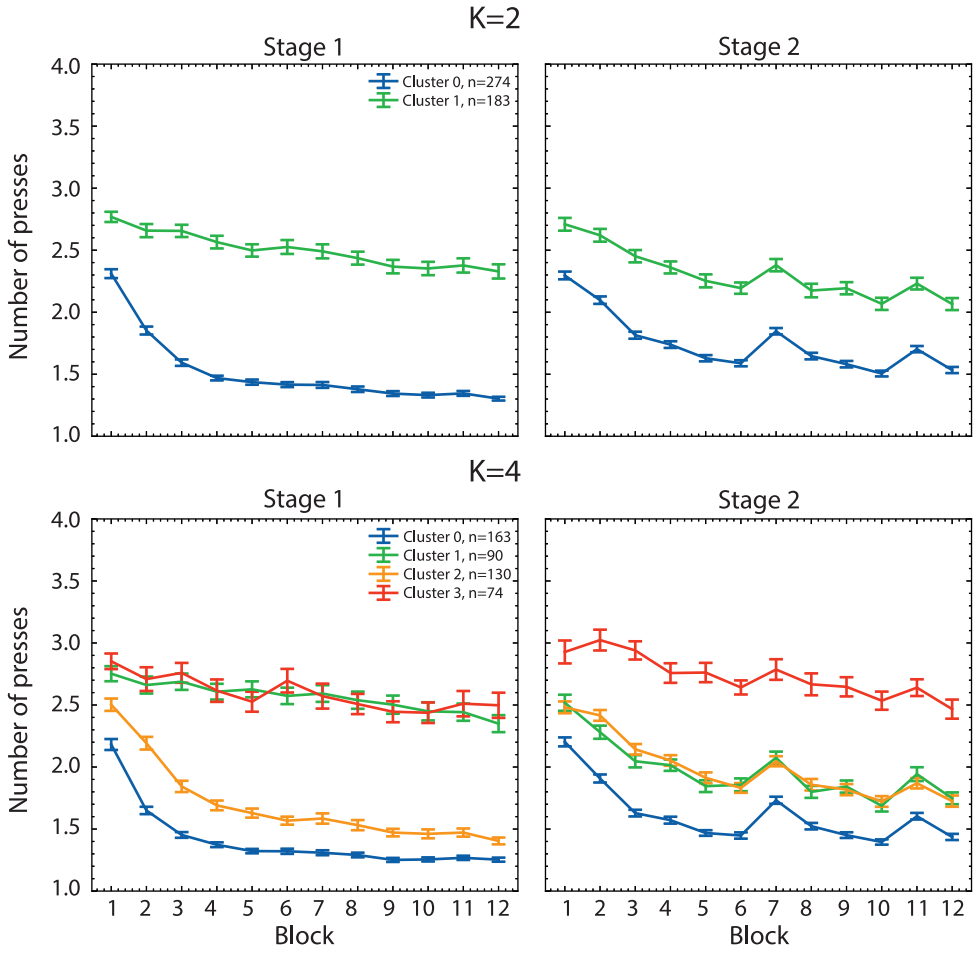

Fig. B.5. Learning curves of all clusters identified by the k-means algorithm applied to Experiment 1 data with $k = 2$ and $k = 4$.

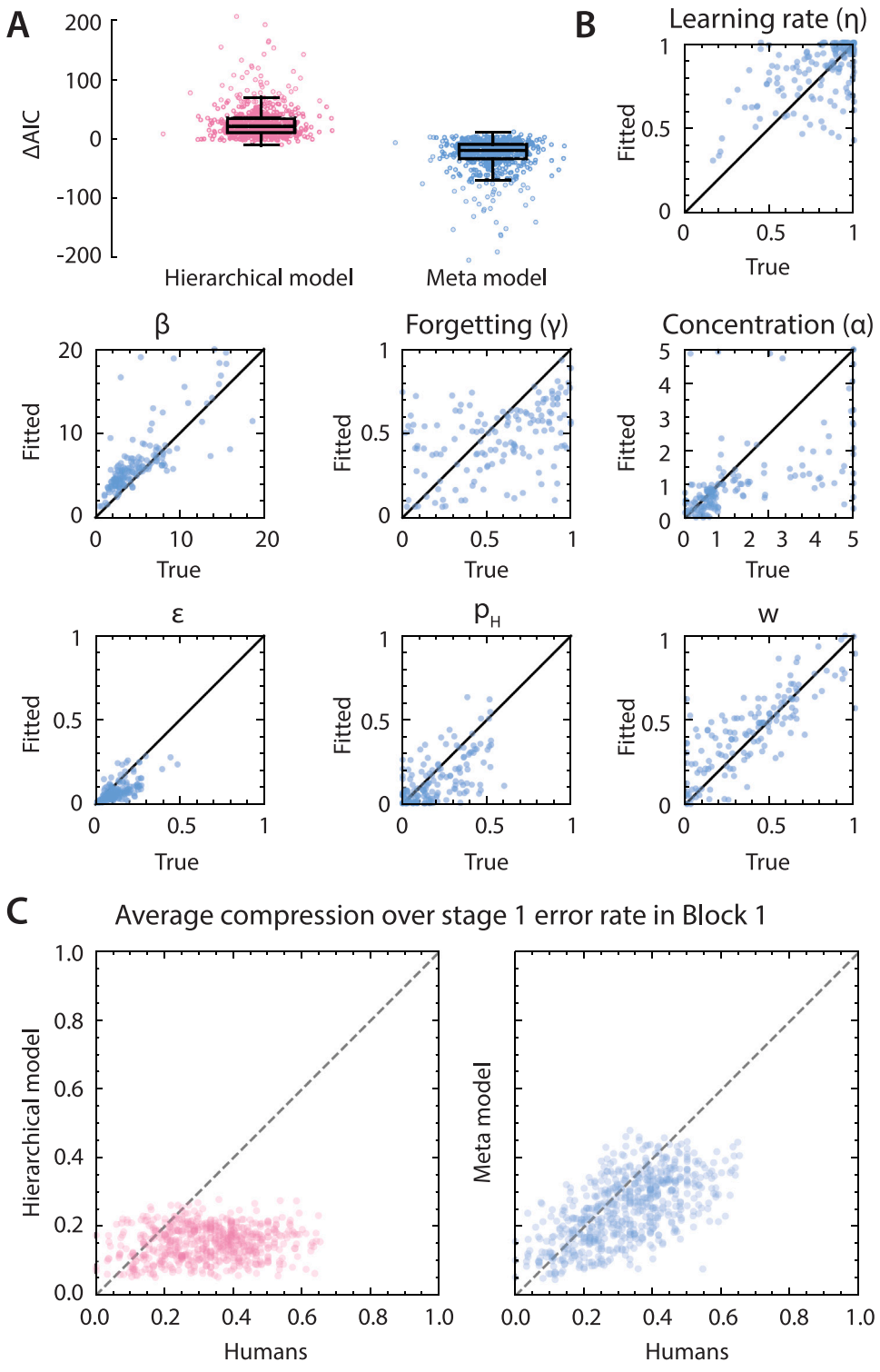

Fig. B.6. Meta-learning model (temporally backward) fit. A: Comparison of the fully hierarchical model and the meta-learning model by AIC on the top performers. B: Parameter recovery analysis of the meta-learning model using the data of top performers under V1-V1 (n = 146). Parameter identification is strong for some parameters (e.g., β), but weaker for others (e.g., η). C: Model recovery of compression over stage 1 error rate in Block 1 at the individual level. Pearson r = 0.14 and p = 7×10^{-4} for hierarchical model and $r = 0.63$ and p< 10^{-4} for the meta model.

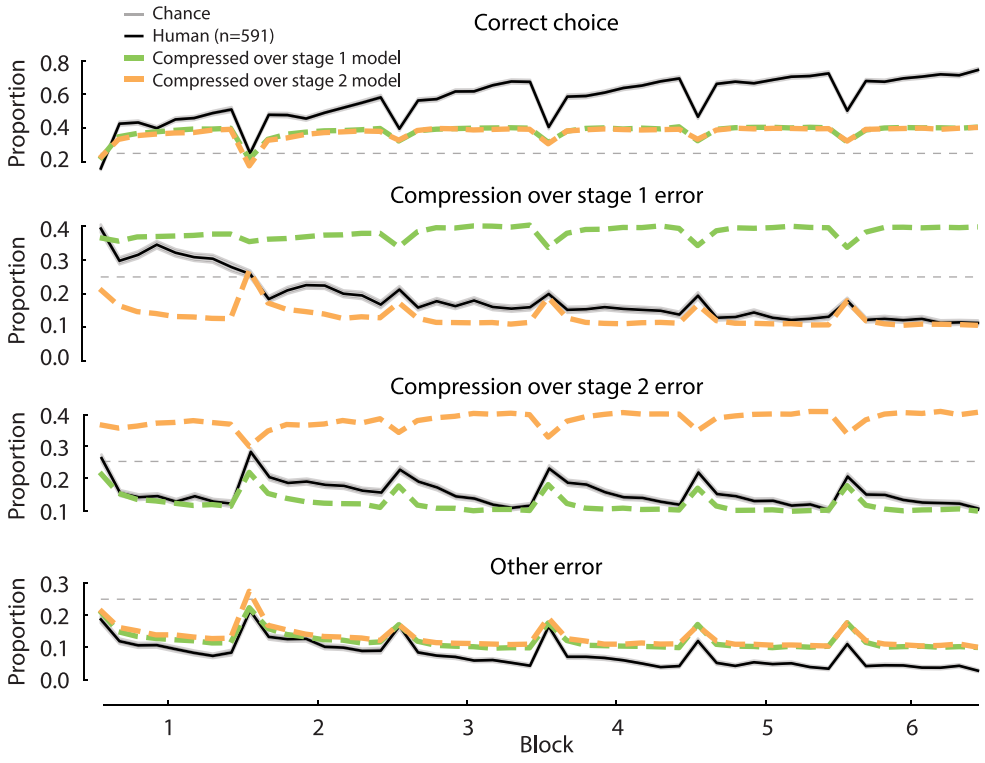

Fig. B.7. Learning curves of the models implementing the compressed policies over stage 1 and stage 2, respectively. The models were fitted to human choice data and the fitted parameters were used to simulate choice data.

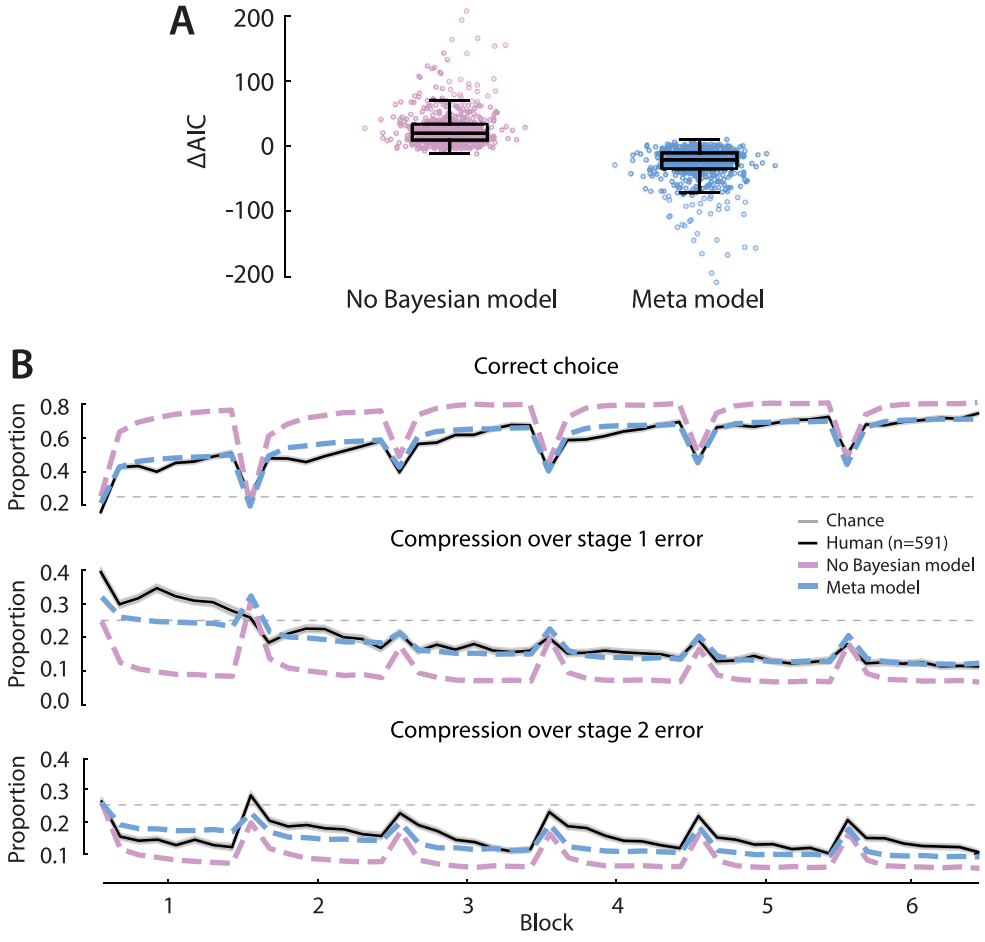

Fig. B.8. Ablating the Bayesian inference mechanism in the meta-learning model. A: Comparison of the model without Bayesian updates (only priors over the three policies) and the meta-learning model by AIC. The meta model fitted significantly better (two-tailed t-test $t = 23$, $p < 10^{-4}$). B: Learning curves of both models compared to humans. The meta model captured the qualitative error patterns in human behavior while the no Bayesian model failed to.

References

Abel, D., Arumugam, D., Lehnert, L., & Littman, M. (2018). State abstractions for lifelong reinforcement learning. In *International conference on machine learning* (pp. 10–19). PMLR.

Badre, D. (2008). Cognitive control, hierarchy, and the rostro–caudal organization of the frontal lobes. *Trends in Cognitive Sciences*, *12*(5), 193–200.

Badre, D., & D'Esposito, M. (2007). Functional magnetic resonance imaging evidence for a hierarchical organization of the prefrontal cortex. *Journal of Cognitive Neuroscience*, *19*(12), 2082–2099.

Botvinick, M. M. (2007). Multilevel structure in behaviour and in the brain: a model of Fuster's hierarchy. *Philosophical Transactions of the Royal Society, Series B (Biological Sciences)*, *362*(1485), 1615–1626.

Botvinick, M. M., Niv, Y., & Barto, A. G. (2009). Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective. *cognition*, *113*(3), 262–280.

Collins, A. G., Cavanagh, J. F., & Frank, M. J. (2014). Human EEG uncovers latent generalizable rule structure during learning. *Journal of Neuroscience*, *34*(13), 4677–4685.

Collins, A. G., & Frank, M. J. (2012). How much of reinforcement learning is working memory, not reinforcement learning? A behavioral, computational, and neurogenetic analysis. *European Journal of Neuroscience*, *35*(7), 1024–1035.

Collins, A. G., & Frank, M. J. (2013). Cognitive control over learning: creating, clustering, and generalizing task-set structure. *Psychological Review*, *120*(1), 190.

Collins, A. G. E., & Frank, M. J. (2016). Neural signature of hierarchically structured expectations predicts clustering and transfer of rule sets in reinforcement learning. *Cognition*, *152*, 160–169.

Collins, A., & Koechlin, E. (2012). Reasoning, learning, and creativity: frontal lobe function and human decision-making. *PLoS Biology*, *10*(3), Article e1001293.

Correa, C. G., Sanborn, S., Ho, M. K., Callaway, F., Daw, N. D., & Griffiths, T. L. (2023). Exploring the hierarchical structure of human plans via program generation. arXiv preprint arXiv:2311.18644.

Daw, N. D., Gershman, S. J., Seymour, B., Dayan, P., & Dolan, R. J. (2011). Model-based influences on humans' choices and striatal prediction errors. *Neuron*, *69*(6), 1204–1215.

Diuk, C., Schapiro, A., Córdova, N., Ribas-Fernandes, J., Niv, Y., & Botvinick, M. (2013). Divide and conquer: hierarchical reinforcement learning and task decomposition in humans. *Computational and Robotic Models of the Hierarchical Organization of Behavior*, 271–291.

Eckstein, M. K., & Collins, A. G. (2020). Computational evidence for hierarchically structured reinforcement learning in humans. *Proceedings of the National Academy of Sciences*, *117*(47), 29381–29389.

Franklin, N. T., & Frank, M. J. (2018). Compositional clustering in task structure learning. *PLoS Computational Biology*, *14*(4), Article e1006116.

Gershman, S. J., Horvitz, E. J., & Tenenbaum, J. B. (2015). Computational rationality: A converging paradigm for intelligence in brains, minds, and machines. *Science*, *349*(6245), 273–278.

Ho, M. K., Abel, D., Correa, C. G., Littman, M. L., Cohen, J. D., & Griffiths, T. L. (2022). People construct simplified mental representations to plan. *Nature*, *606*(7912), 129–136.

Koechlin, E., Ody, C., & Kouneiher, F. (2003). The architecture of cognitive control in the human prefrontal cortex. *Science*, *302*(5648), 1181–1185.

Lai, L., & Gershman, S. J. (2021). Policy compression: An information bottleneck in action selection. *Psychology of Learning and Motivation*, *74*, 195–232.

Lai, L., & Gershman, S. J. (2024). Human decision making balances reward maximization and policy compression. *PLoS Computational Biology*, *20*(4), Article e1012057.

Lai, L., Huang, A. Z., & Gershman, S. J. (2022). Action chunking as policy compression. PsyArXiv.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, *40*, Article e253.

Lehnert, L., Littman, M. L., & Frank, M. J. (2020). Reward-predictive representations generalize across tasks in reinforcement learning. *PLoS Computational Biology*, *16*(10), Article e1008317.

Li, L., Walsh, T. J., & Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. *AI&M*, *1*(2), 3.

Li, J.-J., Xia, L., Dong, F., & Collins, A. G. (2022). Credit assignment in hierarchical option transfer. In *CogSci... annual conference of the cognitive science society. cognitive science society (US). conference, vol. 44* (p. 948). NIH Public Access.

Lieder, F., & Griffiths, T. L. (2020). Resource-rational analysis: Understanding human cognition as the optimal use of limited computational resources. *Behavioral and Brain Sciences*, *43*, Article e1.

Molinaro, G., & Collins, A. G. (2023). A goal-centric outlook on learning. *Trends in Cognitive Sciences*.

Pitman, J. (2006). *Combinatorial stochastic processes: Ecole d'eté de probabilités de saint-flour xxxii-2002*. Springer.

Simon, H. A. (1955). A behavioral model of rational choice. *The Quarterly Journal of Economics*, 99–118.

Solway, A., Diuk, C., Córdova, N., Yee, D., Barto, A. G., Niv, Y., et al. (2014). Optimal behavioral hierarchy. *PLoS Computational Biology*, *10*(8), Article e1003779.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, *112*(1–2), 181–211.

Tomov, M. S., Yagati, S., Kumar, A., Yang, W., & Gershman, S. J. (2020). Discovery of hierarchical representations for efficient planning. *PLoS Computational Biology*, *16*(4), Article e1007594.

Wise, T., Emery, K., & Radulescu, A. (2023). Naturalistic reinforcement learning. *Trends in Cognitive Sciences*.

Xia, L., & Collins, A. G. (2021). Temporal and state abstractions for efficient learning, transfer, and composition in humans. *Psychological Review*, *128*(4), 643.

Yoo, A. H., & Collins, A. G. (2022). How working memory and reinforcement learning are intertwined: A cognitive, neural, and computational perspective. *Journal of Cognitive Neuroscience*, *34*(4), 551–568.