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

Jing-Jing Li[1*] and Anne G. E. Collins[1,2*]

[1]Helen Wills Neuroscience Institute, University of California, Berkeley.
[2]Department of Psychology, University of California, Berkeley.

*Corresponding author(s). E-mail(s): jl3676@berkeley.edu; annecollins@berkeley.edu;

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 1,026 human participants on 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.

Keywords: computational cognitive modeling, abstraction, hierarchy, meta-learning, decision-making, transfer, composition

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 *contexualize* one another. An earlier decision might affect the policy for a later decision: if you had just preheated the oven, your policy on 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 (Figure 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 [1]. 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 [2–13]. Such abstractions can form over time and states, serving as a foundation of efficient and flexible learning by

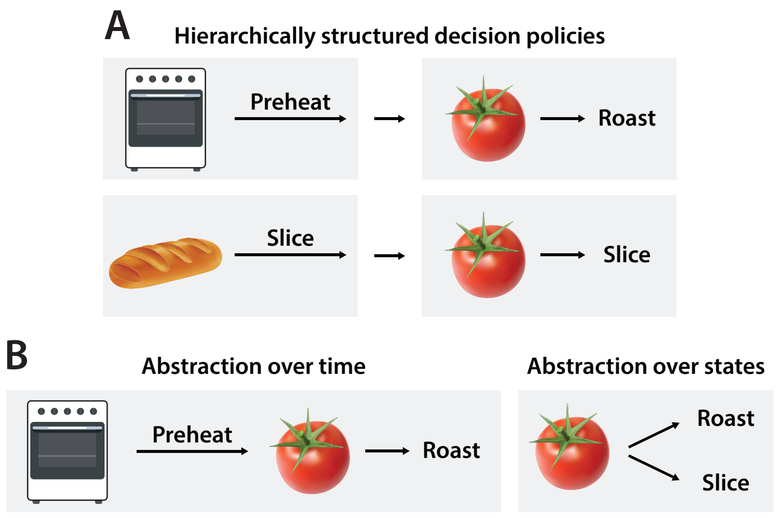

Fig. 1 Hierarchically structured decision policies and abstractions of policy information. A: In a hierarchically structured decision policy, the decision at a later time (e.g., what action to take with a tomato) is conditional on not only the immediate state (e.g., tomato), but also other related states 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.

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 [14, 15].

When hierarchical representations are temporally abstracted, a sequence of actions are *chunked* together via some policy [2, 5, 16, 17]. These action chunks can be organized by subgoals that decompose a bigger task into smaller subtasks that are easier to solve [3, 4]. In terms of our tomato example, the actions of preheating the oven and roasting the tomato can be chunked 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 (Figure 1B). Cognitive scientists have used the options framework [18] from hierarchical reinforcement learning to model temporally abstracted policies [2, 5]. Unlike classic reinforcement learning [19], 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.

In state abstractions, similar states are grouped to form compressed representations of the full state space [20, 21]. For example, both courses of actions illustrated in Figure 1A involve the tomato, which can be compressed into a single abstract state representation (Figure 1B). Recent work has shown that humans learn compressed policies over states to jointly maximize reward and representation efficiency [22–24]. 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 [6, 25–27]. In our example, the abstract tomato state and the skill of slicing tomatoes can be further shared with related tasks such as making a pizza, which may require sliced tomatoes as toppings.

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 [5, 7] 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 [5] without catastrophic forgetting of existing policy representations [7] in this task. Building on these findings, we incorporated novel structural designs with robust controls to further disentangle the interactions between different types of abstractions in learning and representing hierarchical policies. 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 Results

2.1 Experimental design

We tested 1,026 human participants on a decision-making task where they could learn, transfer, and recompose multiple sets of hierarchical policies (Figure 2). Our paradigm extended prior work [5, 7] to adopt more engaging stimuli and separate the action spaces between trial stages. 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

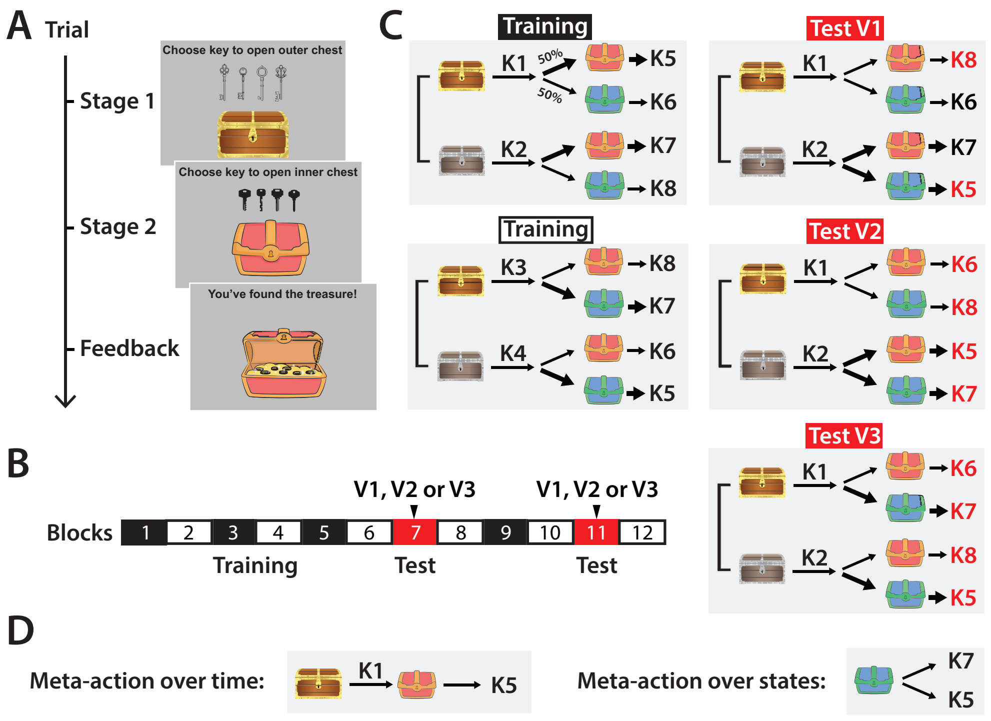

Fig. 2 The task paradigm. A: Participants learned to unlock two nested chests (gold/silver in stage 1 followed by red/blue in stage 2) by finding the correct keys through trial-and-error via deterministic feedback. They could only proceed to the next stage (pseudo-randomly determined) after selecting 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. 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 (highlighted in red); meta-actions over states learned in training were only preserved in V3, among all test structures (emphasized in bold).

be unlocked by two sets of four keys (denoted as K1-K4 and K5-K8; Figure 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 (Figure 2B). There were 32-60 trials (up to performance criterion, see Methods) in Blocks 1-2 and 32 trials in each following block. Five different hierarchically structured chest-key (stimulus-action) mappings were used in the experiment: two in training and control blocks for all participants, and three in test blocks, denoted as V1, V2, and V3 (Figure 2C). 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*. We define a meta-action as a pair of actions chunked together either over time or over states (Figure 2D). A meta-action over time abstracts two atomic actions into a policy that specifies a compound action sequence spanning both stages, whereas 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. 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). Each participant experienced one of the following test conditions in 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.

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

2.2 Human learning performance

Observed human behavior qualitatively replicated findings in prior studies where applicable [5, 7]. 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 (Figure 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 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

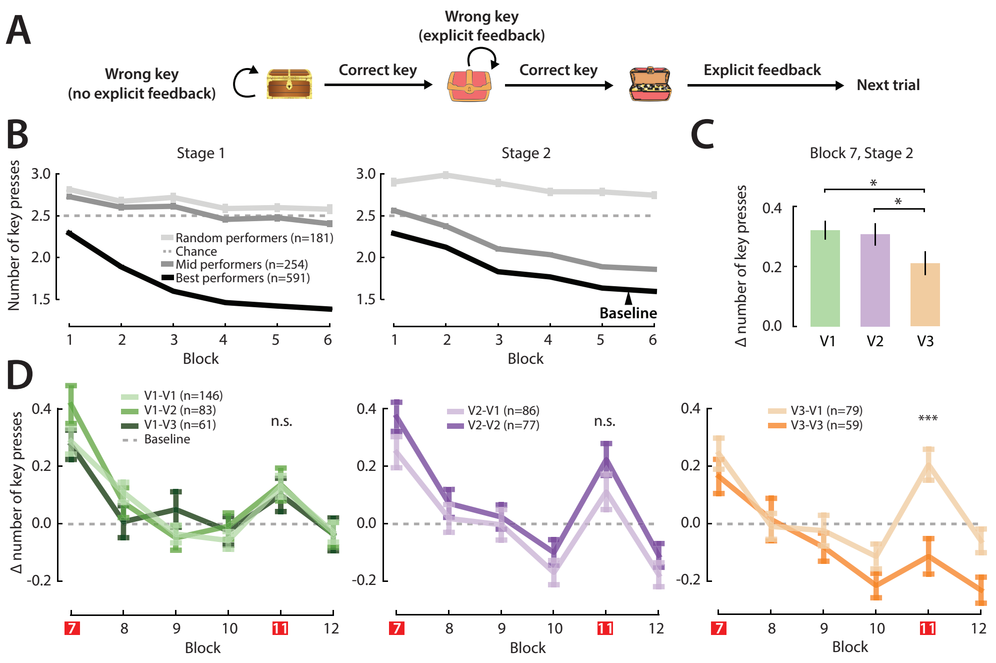

Fig. 3 Summary of human behavior. A: Task performance was measured by the number of key presses the participant made until reaching the correct choice in each stage, averaged over the first 10 trials of each block. On each trial, the participant had to press the correct key to advance from stage 1 to 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 or V2. 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.

same action on the same trial, 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.

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 (Figure 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 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 Extended data. 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 [5] (Figure 3C for best performers, Figure A1B for mid performers). Notably, the increase in the number of key presses from baseline was significantly lower in V3 than V1 or V2 (one-tailed t-test p=0.018 between V1 and V3 and p=0.039 between V2 and V3). This discrepancy indicated that V3's consistent meta-actions over states (K6-K8 and K5-K7) with the training structures facilitated new learning and transfer. On the other hand, due to the lack of discrepancy between V1 and V2 (two-tailed t-test 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 [7]. 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 (one-tailed t-test $p=3.8 \times 10^{-3}$ for V1-V1, p=0.025 for V2-V2, and $p=7.7 \times 10^{-4}$ for V3-V3), indicating transfer of learned structures (Figure 3D for best performers, Figure A1A 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 p=0.93 between V1-V1 and V1-V2, and p=0.18 between V2-V1 and V2-V2), while it was significantly worse in V3-V1 than V3-V3 (one-tailed t-test $p=7.9 \times 10^{-5}$). 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 V1. However, no transfer was observed between V1 and V3 since performance only improved in V3-V3 but not V3-V1.

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

The behavioral analyses above present strong 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

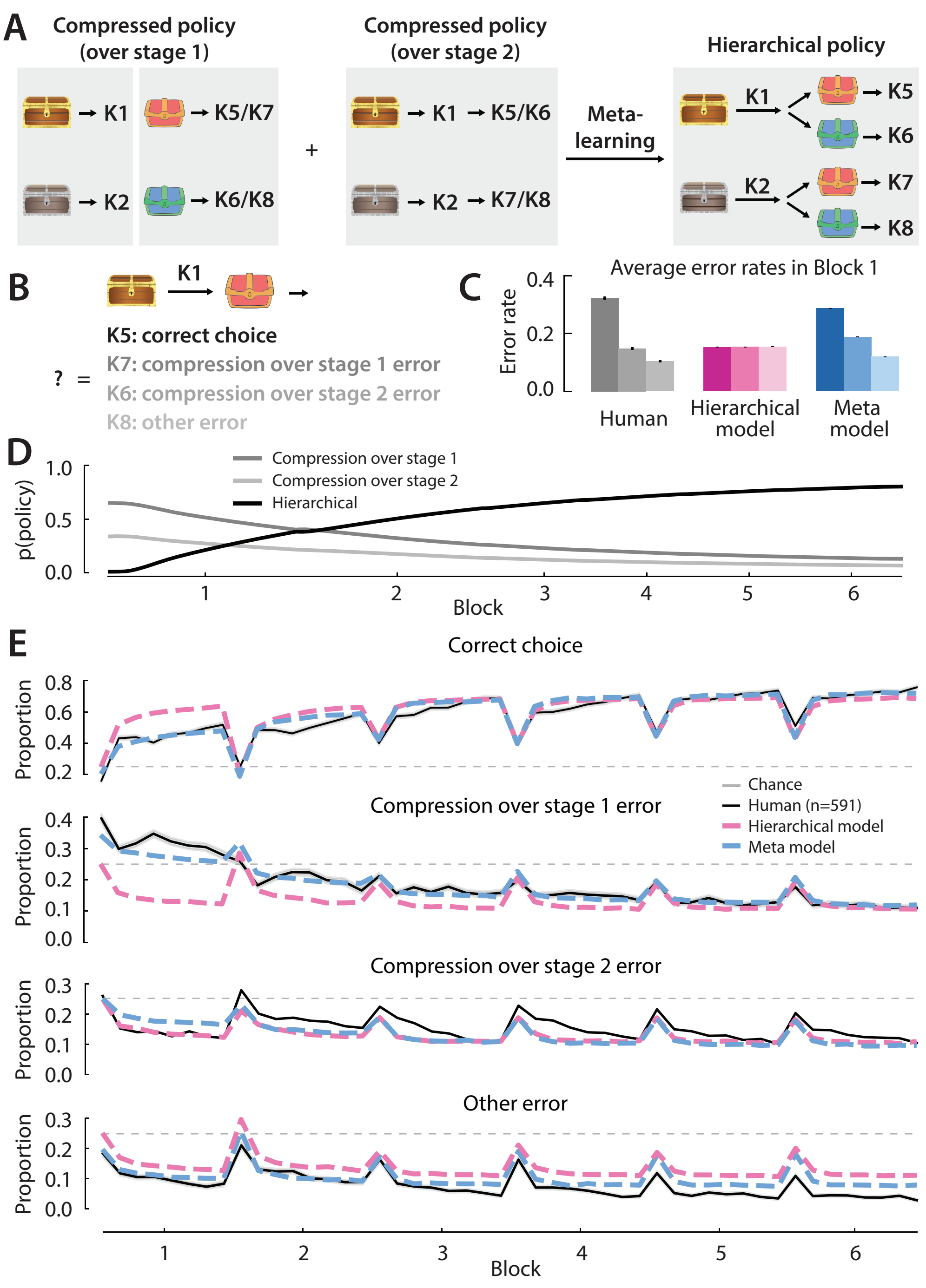

Fig. 4 Compressed policies unfolded into hierarchical policies via meta-learning. A: We modeled two types of compressed policies and a hierarchical policy, which used information from different stages to choose in stage 2. The policy that was compressed over stage 1 disregarded information in stage 1 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.

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.

2.3.1 Modeling the emergence of hierarchical policies

Compressed policies can form over the state space of either stage 1 or stage 2 (Figure 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 Figure 4B). In Block 1, compression errors, especially over stage 1, dominated the wrong choices made by humans on the first attempt of each trial (Figure 4C for best performers, Figure A1C 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 (Figure 4A). The meta model started learning with a small, non-zero prior on the hierarchical structure; its priors on the compressed structures were determined by a fitted parameter. The meta model successfully produced the error distribution in early human choices, while the hierarchical model failed to (Figure 4C). The meta model predicted that humans started learning with a higher preference for compression over stage 1, and slowly switched to making choices based on hierarchical rather than compressed policies (Figure 4D). 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 (Figure 4E for best performers, Figure A1D for mid performers).

2.3.2 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 contexualize 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 (Figure 5A, B). In the former case, the hierarchy is constructed in

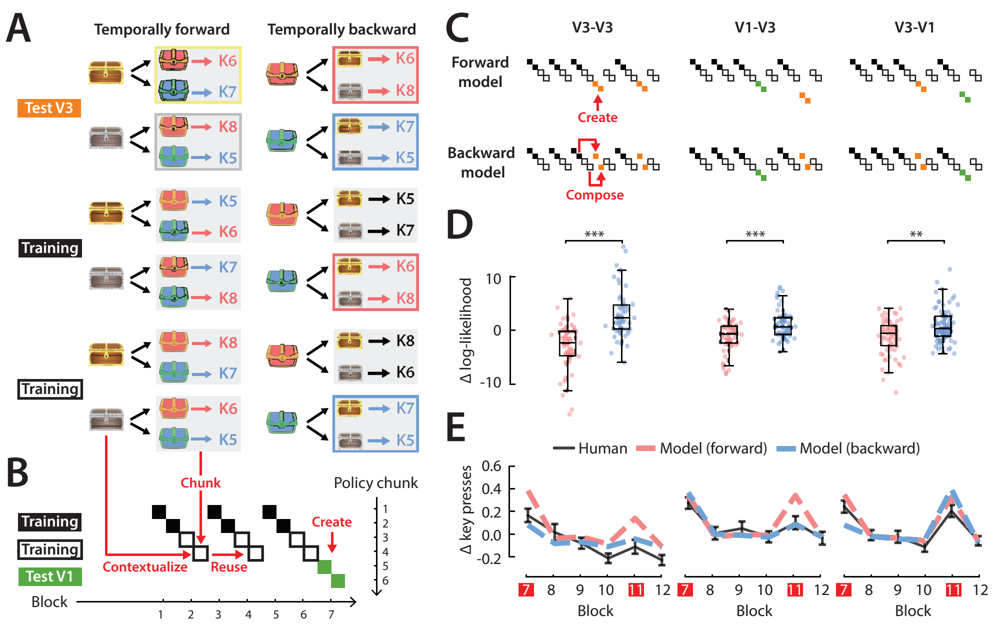

Fig. 5 The learned hierarchical policies in stage 2 followed a temporally backward structure. A: Temporally forward structures use earlier information (stage 1) to contextualize policy chunks defined over later information (stage 2), while temporally backward structures use later information (stage 2) 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).

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) contexualizes 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 chunks learned during training (the second chunk from each training structure, whose borders are highlighted in Figure 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 flexibly composed learned policy chunks (Figure 5C). All visualizations similar to Figure 5C in this manuscript illustrate the most likely

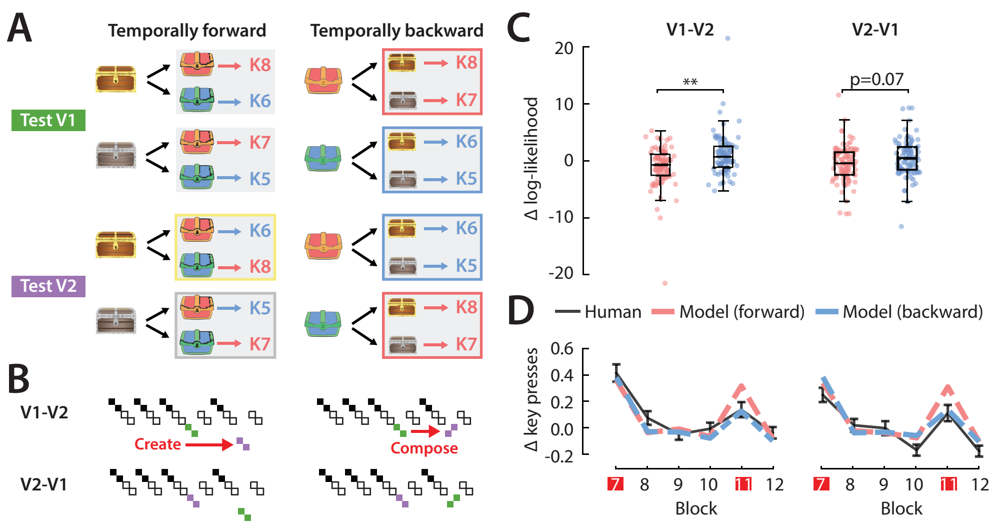

Fig. 6 Temporally backward structures enabled efficient composition between V1 and V2. A: Temporally backward structures allowed for recomposition of learned policy chunks between V1 and V2, while temporally forward structures did not. B, C: The forward model created two sets of structures to represent 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.

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 (Figure 5D; one-tailed t-test $p=8.3 \times 10^{-7}$ for V3-V3, $p=8.7 \times 10^{-4}$ for V1-V3, and $p=2.8 \times 10^{-3}$ 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 (Figure 5E). Consistent results were observed in the mid performers (Figure A2).

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 (Figure 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 (Figure 6A). Therefore, the backward model could compose learned policy chunks between V1 and V2, while the forward model needed to create a new set of policy chunks upon learning each test structure (Figure 6B). The backward model fitted human choices better in terms of both the likelihood metric (Figure 6C) and capturing the qualitative pattern of performance improvement between V1 and V2 in human

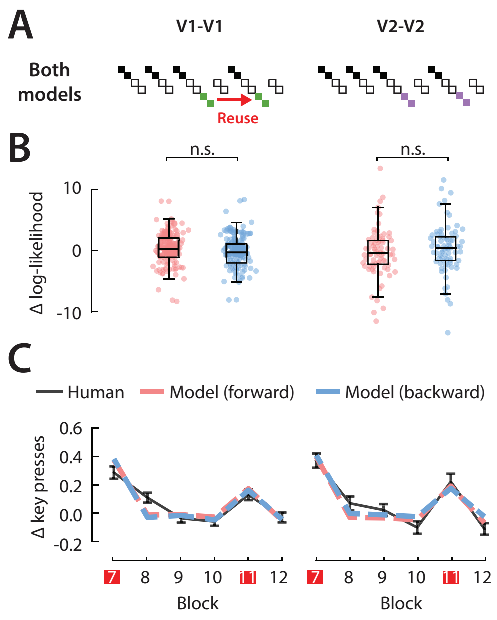

Fig. 7 The forward and backward models both successfully captured human behavior in repeating V1 and V2 test conditions. A: Both models created new policy chunks upon V1 or V2, which were reused when the same test structure repeated. B: The likelihoods of the forward and backward models were not significantly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

behavior (Figure 6D). The backward model also matched human behavior better qualitatively in the mid performers, though there were no significant differences in model likelihoods (Figure A2).

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 (Figure 7A). When fitted to human data, the forward and backward models were indistinguishable by the likelihood metric (Figure 7B). Both models successfully accounted for the performance improvement between test blocks in V1-V1 and V2-V2.

3 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. This meta-cognitive process is resource-rational [28–30]: under the constraint of limited cognitive resources, humans sacrifice some reward in the beginning to prioritize the effortful task of structure learning.

Contrary to our expectations based on previous work [2, 5], 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). 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 [2, 18].

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 [31, 32]. 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 [33, 34], 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.

4 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.

5 Methods

5.1 Experiment and data

5.1.1 Task

Our task paradigm extends that used by [5, 7] to feature more engaging stimuli (treasure chests instead of geometric shapes) and separate the action spaces between stages 1 and 2 (left hand for stage 1 and right hand for stage 2), which allowed us to investigate the policies human participants used in stage 2 with minimal interference from their policies in stage 1. 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; Figure 3A).

5.1.2 Participants

All participants were recruited online through the Research Participation Program at the University of California, Berkeley. In total, 1,026 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: Experiment 1 tested all combination conditions of V1 and V2, while Experiment 2 included all combinations of V1 and V3. Table 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 difference in performance or error types, which indicated that there were no external factors driving any learning differences between experiments. Therefore, data from Experiments 1 and 2 were combined in all analyses.

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

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

5.1.3 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. We set $k = 3$, which 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 (Figure 3B).

5.2 Modeling

5.2.1 Hierarchical model

Algorithm 1 outlines the hierarchical model, which extends and improves on prior work [5, 35]. Since our current work focuses on modeling choices in stage 2 independent of choices in stage 1, the latter is skipped in our models. However, stage 1 policies can be modeled independently from stage 2 using a similar algorithm [5].

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 contexualized by a combination of the block number and the identity of the stimulus that serves as the context of the policy chunks (stage

Algorithm 1 Fully hierarchical model

```
Require: Parameters $\theta = \{\alpha, \beta, \lambda, \epsilon\}$
for $t = 1, 2, ..., T$ do                                              ▷ Skip stage 1
    Observe context $c_t$ and state $s_t$
    Compute alternative context in the same block $c'_t$
    if $c_t$ is new then                                               ▷ New context
        Initialize prior $\Pr(TS_i \mid c_t\,;\, t) \propto \sum_c \Pr(TS_i \mid c\,;\, t)$ for all $TS_i$
        Create a new task-set $TS_{c_t}$ with uninformative values
        $\Pr(TS_{c_t} \mid c_t\,;\, t) = \frac{\lambda}{\lambda+1}$                              ▷ CRP
        Normalize $\Pr(TS_i \mid c_t\,;\, t) \leftarrow \frac{\Pr(TS_i \mid c_t\,;\, t)}{\lambda+1}$
    end if
    while $r_t = 0$ do
        Add context-pairing bias based on $c'_t$ to task-set priors for $c_t$
        Sample a deep-copy of $TS_t$ based on $\Pr(TS_i \mid c_t\,;\, t)$ for all $TS_i$
        for $a$ that has been tried on this trial do
            $TS_t(s_t, a\,;\, t) = -\infty$
        end for
        $\pi = \text{softmax}\big(\beta \cdot TS_t(s_t, A\,;\, t)\big) \cdot (1-\epsilon) + \frac{1}{|A|} \cdot \epsilon$
        Sample action $a_t \sim \pi$ and observe reward $r_t$
        if $r_t = 1$ then                                              ▷ Bayesian update
            $\Pr(TS_i \mid c_t\,;\, t+1) \propto \Pr(TS_i \mid c_t\,;\, t) \cdot \pi(a_t)$ for all $TS_i$
        else
            $\Pr(TS_i \mid c_t\,;\, t+1) \propto \Pr(TS_i \mid c_t\,;\, t) \cdot \big(1 - \pi(a_t)\big)$ for all $TS_i$
        end if
        Re-sample $TS_t$ based on $\Pr(TS_i \mid c_t\,;\, t+1)$ for all $TS_i$
        $TS_t(s_t, a_t\,;\, t) \leftarrow TS_t(s_t, a_t\,;\, t) + \alpha \cdot \big(r_t - TS_t(s_t, a_t\,;\, t)\big)$   ▷ Update value
    end while
end for
```

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 (CRP) parameterized by λ [36]: 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 λ (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 Figures 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:

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

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 [35].

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

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

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).

5.2.2 Meta-learning model

The meta-learning model, outlined in Algorithm 2, extends the fully hierarchical model to include two compressed policies over states (Figure 4A). To support the meta-learning mechanism, two additional parameters (m and β_{meta}) are included in the model: m defines the prior probability of the compressed policy over stage 2, and β_{meta} controls the inverse temperature of a softmax transformation applied to the policy probabilities at decision time.

At the beginning of learning, the prior of π_H, the hierarchical policy, is fixed at a small, non-zero value (0.01 in our analyses) to model a weak, but non-vanishing, initial belief of hierarchical structure. The prior of π_{C_2}, the compressed policy over stage 2, is parameterized by m, which is optimized in the fitting process. The last prior probability of π_{C_1}, the compressed policy over stage 1, can then be calculated as $1-0.01-m$ since all priors sum to 1 (Line 1). During learning, these belief probabilities over policies are updated using Bayes' rule after observing the reward (Lines 25-29).

Algorithm 2 Meta-learning model

```
Require: Parameters $\theta = \{\alpha, \beta, \lambda, \epsilon, m, \beta_{\text{meta}}\}$
 1: Initialize policy priors $\Pr(\pi\,;\,0) = [0.99 - m, m, 0.01]$ 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:             $\pi_{C_1} = \text{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS_t(s'_t, A\,;\,t)\big)/2\big)$
15:             $\pi_{C_2} = \text{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS'_t(s_t, A\,;\,t)\big)/2\big)$
16:         else
17:             $\pi_{C_1} = \text{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS'_t(s_t, A\,;\,t)\big)/2\big)$
18:             $\pi_{C_2} = \text{softmax}\big(\beta \cdot \big(TS_t(s_t, A\,;\,t) + TS_t(s'_t, A\,;\,t)\big)/2\big)$
19:         end if
20:         $\pi_H = \text{softmax}\big(\beta \cdot TS_t(s_t, A\,;\,t)\big)$                     ▷ Hierarchical policy
21:         $p_t(\pi) = \text{softmax}(\beta_{\text{meta}} \cdot \Pr(\pi\,;\,t))$
22:         $\pi_m = \big(\sum_{i \in \{C_1, C_2, H\}} p_t(\pi_i) \cdot \pi_i\big) \cdot (1-\epsilon) + \frac{1}{|A|} \cdot \epsilon$         ▷ Meta-policy
23:         Sample action $a_t \sim \pi_m$ and observe reward $r_t$
24:         Update task-set belief probabilities as in Algorithm 1
25:         if $r_t = 1$ then                                                 ▷ Bayesian update
26:             $\Pr(\pi\,;\,t+1) \propto \Pr(\pi\,;\,t+1) \cdot \pi(a_t)$
27:         else
28:             $\Pr(\pi\,;\,t+1) \propto \Pr(\pi\,;\,t+1) \cdot \big(1 - \pi(a_t)\big)$
29:         end if
30:         Update value in resampled task-set in $c_t$ as in Algorithm 1
31:     end while
32: end for
```

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 (Lines 21-22).

5.2.3 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:

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

Data and code availability. All analysis, modeling, and plotting code used to produce results and figures in this manuscript can be found at: https://github.com/jl3676/learning_hierarchy. All human behavioral data will be uploaded to the repository upon acceptance of this manuscript.

Supplementary information. See Extended data for supplementary figures.

Acknowledgements. This work was supported by the NIMH grant R01MH119383. We thank Liyu Xia, Milena Rmus, and Daniel Ehrlich for helpful discussions throughout the project.

Appendix A Extended data

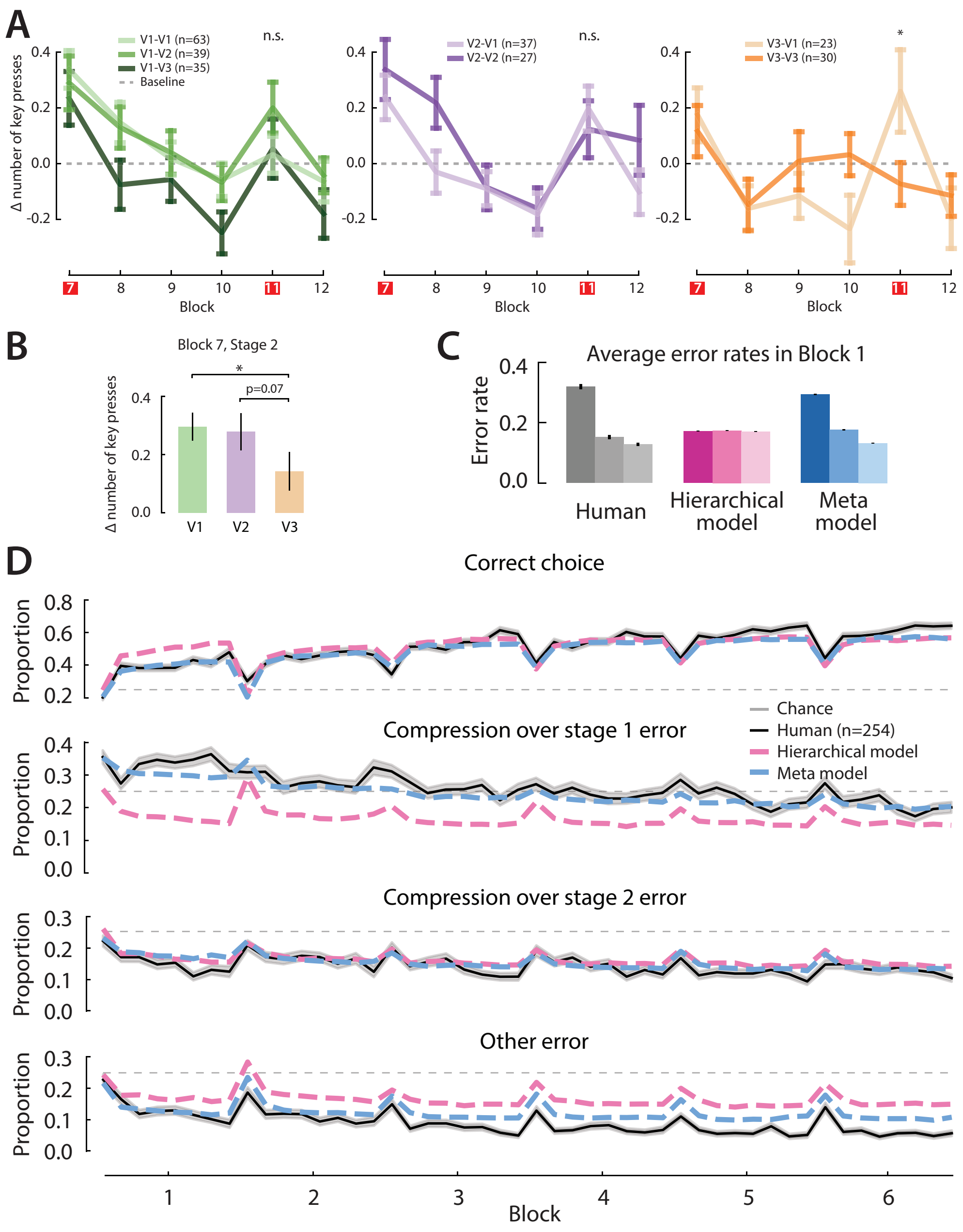

Fig. A1 Learning behavior of the mid performers cluster. A: Post-training learning curves by test condition. B: Human performance was better in the first test block of V3 than V1 and V2. C: The meta model captured the error rate patterns in early learning of humans, which the fully hierarchical model failed to. D: Human error curves in training were better matched by the meta model than the hierarchical model.

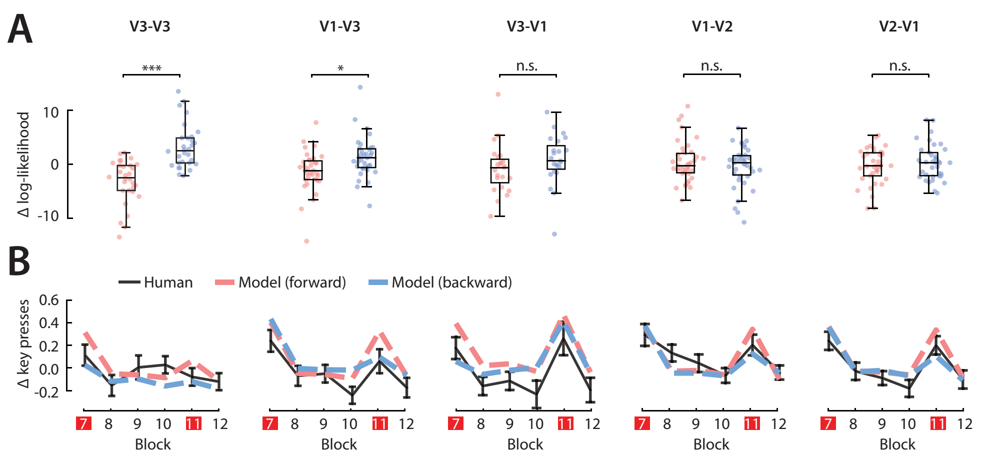

Fig. A2 Comparisons of the temporally forward and backward models on mid performer data. A: In V3-V3 and V1-V3, the backward model had significantly higher likelihoods than the forward model, while the models did not produce significantly different likelihoods in V3-V1, V1-V2, and V2-V1. B: Overall, the backward model captured qualitative patterns in human learning performance better than the forward model.

References

[1] Wise, T., Emery, K., Radulescu, A.: Naturalistic reinforcement learning. Trends in Cognitive Sciences (2023)

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

[3] Diuk, C., Schapiro, A., Córdova, N., Ribas-Fernandes, J., Niv, Y., Botvinick, M.: Divide and conquer: hierarchical reinforcement learning and task decomposition in humans. Computational and robotic models of the hierarchical organization of behavior, 271–291 (2013)

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

[5] Xia, L., Collins, A.G.: Temporal and state abstractions for efficient learning, transfer, and composition in humans. Psychological review **128**(4), 643 (2021)

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

[7] Li, J.-J., Xia, L., Dong, F., Collins, A.G.: Credit assignment in hierarchical option transfer. In: CogSci... Annual Conference of the Cognitive Science Society. Cognitive Science Society (US). Conference, vol. 44, p. 948 (2022). NIH Public

Access

[8] Badre, D., D'Esposito, M.: Functional magnetic resonance imaging evidence for a hierarchical organization of the prefrontal cortex. Journal of cognitive neuroscience **19**(12), 2082–2099 (2007)

[9] Badre, D.: Cognitive control, hierarchy, and the rostro–caudal organization of the frontal lobes. Trends in cognitive sciences **12**(5), 193–200 (2008)

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

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

[12] Solway, A., Diuk, C., Córdova, N., Yee, D., Barto, A.G., Niv, Y., Botvinick, M.M.: Optimal behavioral hierarchy. PLoS computational biology **10**(8), 1003779 (2014)

[13] Tomov, M.S., Yagati, S., Kumar, A., Yang, W., Gershman, S.J.: Discovery of hierarchical representations for efficient planning. PLoS computational biology **16**(4), 1007594 (2020)

[14] Franklin, N.T., Frank, M.J.: Compositional clustering in task structure learning. PLoS computational biology **14**(4), 1006116 (2018)

[15] Lake, B.M., Ullman, T.D., Tenenbaum, J.B., Gershman, S.J.: Building machines that learn and think like people. Behavioral and brain sciences **40**, 253 (2017)

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

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

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

[19] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT press, ??? (2018)

[20] Abel, D., Arumugam, D., Lehnert, L., Littman, M.: State abstractions for lifelong reinforcement learning. In: International Conference on Machine Learning, pp. 10–19 (2018). PMLR

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

[22] Lai, L., Gershman, S.J.: Policy compression: An information bottleneck in action selection. Psychology of learning and motivation **74**, 195–232 (2021)

[23] Lai, L., Huang, A.Z., Gershman, S.J.: Action chunking as policy compression (2022)

[24] Lai, L., Gershman, S.J.: Human decision making balances reward maximization and policy compression. PLOS Computational Biology **20**(4), 1012057 (2024)

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

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

[27] Lehnert, L., Littman, M.L., Frank, M.J.: Reward-predictive representations generalize across tasks in reinforcement learning. PLoS computational biology **16**(10), 1008317 (2020)

[28] Simon, H.A.: A behavioral model of rational choice. The quarterly journal of economics, 99–118 (1955)

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

[30] Lieder, F., Griffiths, T.L.: Resource-rational analysis: Understanding human cognition as the optimal use of limited computational resources. Behavioral and brain sciences **43**, 1 (2020)

[31] Collins, A.G., Frank, M.J.: 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 (2012)

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

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

[34] Molinaro, G., Collins, A.G.: A goal-centric outlook on learning. Trends in

Cognitive Sciences (2023)

[35] Collins, A., Koechlin, E.: Reasoning, learning, and creativity: frontal lobe function and human decision-making. PLoS biology **10**(3), 1001293 (2012)

[36] Pitman, J.: Combinatorial Stochastic Processes: Ecole D'eté de Probabilités de Saint-flour Xxxii-2002. Springer, ??? (2006)