Artificial neural networks for model identification and parameter estimation in computational cognitive models

Milena Rmus, Ti-Fen Pan, Liyu Xia, and Anne G. E. Collins

UC Berkeley

September 14, 2023

1 Abstract

Computational cognitive models have been used extensively to formalize cognitive processes. Model parameters offer a simple way to quantify individual differences in how humans process information. Similarly, model comparison allows researchers to identify which theories, embedded in different models, provide the best accounts of the data. Cognitive modeling uses statistical tools to quantitatively relate models to data that often rely on computing/estimating the likelihood of the data under the model. However, this likelihood is computationally intractable for a substantial number of models. These relevant models may embody reasonable theories of cognition, but are often under-explored due to the lack of tools required to relate them to data. We propose to fill this gap using artificial neural networks (ANNs) to map data directly onto model identity and parameters, bypassing the likelihood estimation. Our results show that we can adequately perform both parameter estimation and model identification using our new ANN approach, including for models that cannot be fit using traditional likelihood-based methods. Our new ANN approach will greatly broaden the class of cognitive models researchers can quantitatively consider.

Computational modeling is an important tool for studying behavior, cognition, and neural processes. Computational cognitive models translate scientific theories into algorithms using simple equations with a small number of interpretable parameters to make predictions about the cognitive or neural processes that underlie observable behavioral or neural measures. These models have been widely used to test different theories about cognitive processes that shape behavior and relate to neural mechanisms (Lee and Webb, 2005; Montague et al., 2012; Palminteri et al., 2017; Shultz, 2003). By specifying model equations, researchers can inject different theoretical assumptions into models, and, for most models, simulate synthetic data to make predictions and compare against observed behavior. Researchers can quantitatively arbitrate between different theories by comparing goodness of fit (Akaike, 1998, Wei and Jiang, 2022) across different models. Furthermore, by estimating model parameters that quantify underlying cognitive processes, researchers have been able to characterize important individual differences (e.g. developmental: Eppinger et al., 2013; Hauser et al., 2015; Nussenbaum et al., 2022; Rmus et al., 2023; clinical: Chen et al., 2015; Collins et al., 2014; Gillan et al., 2016; Peterson et al., 2009; Zou et al., 2022) as well as condition effects (Sheynin et al., 2015; Weber et al., 2022).

Researchers' ability to benefit from computational modeling crucially depends on the availability of methods for model fitting and comparison. Such tools are available for a large group of cognitive models (such as, for example, reinforcement learning and drift diffusion models). Examples of model parameter fitting tools include maximum likelihood estimation (MLE, Myung, 2003), maximum a-posteriori (MAP, Cousineau and Helie, 2013), and sampling approaches (Baribault and Collins, 2023; Lee, 2011).Examples of model comparison tools include information criteria such as AIC and BIC (Akaike, 1998; Schwarz, 1978), and Bayesian group level approaches, including protected exceedance probability (Piray et al., 2019; Rigoux et al., 2014). These methods all have one important thing in common - they necessitate computing the likelihood of the data conditioned on models and parameters, thus limiting their use to models with tractable likelihood. However, many models do not have a tractable likelihood. This severely limits the types of inferences researchers can make about cognitive processes, as many models with intractable likelihood might offer better theoretical accounts of the observed data. Examples of such models include cases where observed data (e.g. choices) might depend on latent variables - such as the unobserved rules that govern the choices (Eckstein and Collins, 2020; Frank and Badre, 2012; Solway et al., 2014), or a latent state of engagement (e.g. attentive/distracted, Ashwood et al., 2022; Findling et al., 2021) a participant/agent might be in during the task. In these cases, computing the likelihood of the data often demands integrating over the latent variables (rules/states) across all trials, which grows exponentially and thus is computationally intractable. This highlights an important challenge - computing likelihoods is essential for estimating model parameters, and performing fitness comparison/model identification, and alternative models are less likely to be considered or taken advantage of to a greater extent.

Some existing techniques attempt to bridge this gap. For example, Approximate Bayesian Computation (ABC, Lintusaari et al., 2017; Palestro et al., 2018; Sunnåker et al., 2013; Turner et al., 2013; Turner and Sederberg, 2014) approximates the likelihood empirically based on summary statistics of a large number of simulations. Other approximation methods include Inverse Bayesian Sampling (van Opheusden et al., 2020), particle filtering (Djuric et al., 2003), and assumed density estimation (Minka, 2013). ABC-like approaches have recently been extended by using artificial neural networks (ANNs) to amortize the computational cost of such approaches (Boelts et al., 2022; Fengler et al., 2021); it is important to note that these more recent approaches still rely on likelihood approximation methods. As such, these methods are limited in the research they can support: they either require advanced mathematical expertise for effective use and adaptation to specific models, making them less accessible to most researchers, or make assumptions violated by a broad class of models (e.g models with strong inter-trial dependencies). Furthermore, some of these tools are customized for a specific problem, and might not extend to broader application.

To address these limitations we propose to leverage advances in machine learning, in particular artificial neural networks (ANNs) to develop a general method that can be used to estimate parameters and perform model identification for models with and without tractable likelihood, entirely bypassing the likelihood estimation step. ANNs have been successfully used to fit intractable models in different fields, including weather models (Lenzi et al., 2023) and econometric models (Wei and Jiang, 2022). We develop similar approaches to specifically target the intractability estimation problem in the field of computational cognitive science, and to extend it from parameter estimation only to also model identification.

Our approach relies on the property of ANNs as universal function approximators. The ANN structure we implemented was an RNN with feed-forward layers inspired by Dezfouli et al., 2019 (Fig: 1) that is trained to estimate model parameters, or identify which model most likely generated the data based on input data sequences simulated by the cognitive model.

To validate and benchmark our approach, we first compared it against standard model parameter fitting methods (MLE, MAP, ABC) in cognitive models from different families (reinforcement learning, Bayesian Inference) with tractable likelihood. Next, we demonstrated that neural networks can be used for parameter estimation of models with intractable likelihood, and compared it to standard approximation method (ABC). Finally, we showed that our approach can also be used for model identification. Our results showed that our method is highly successful and robust at parameter and model identification while remaining technically lightweight and accessible. We highlight the fact that our method can be applied to standard cognitive data sets (i.e. with arbitrarily small number of participants, and normal number of trials per participant), as the ANN training is fully done on a large simulated data set. Our work contributes to the ongoing research focusing on leveraging artificial neural networks to advance the field of computational modeling, and provides multiple new avenues for maximizing the utility of computational cognitive models.

2 Results

We focused on two distinct artificial neural network (ANNs) applications in cognitive modeling: parameter estimation and model identification. Specifically, we built a network with a structure suitable for sequential data/data with time dependencies (e.g. recurrent neural network (RNN); Dezfouli et al., 2019). Training deep ANNs requires large training data sets. We generated such a data set at minimal cost by simulating a cognitive computational model on a cognitive task a large number of times. Model behavior in the cognitive task (e.g. a few hundred trials of stimulus-action pairs or stimulus-action-outcome triplets (depending on the task) for each simulated agent) constituted ANN's training input; true known parameter values (or identity of the model) from which the data was simulated constituted ANNs' training targets. We evaluated the network's training performance in predicting parameter values/identity of the model in a separate validation set, and tested the trained network on a held out test set. We tested RNN variants and compared their accuracy against traditional likelihood-based model fitting/identification methods using both likelihood-tractable and likelihood-intractable cognitive models. See Methods section for details on the ANN training and testing process.

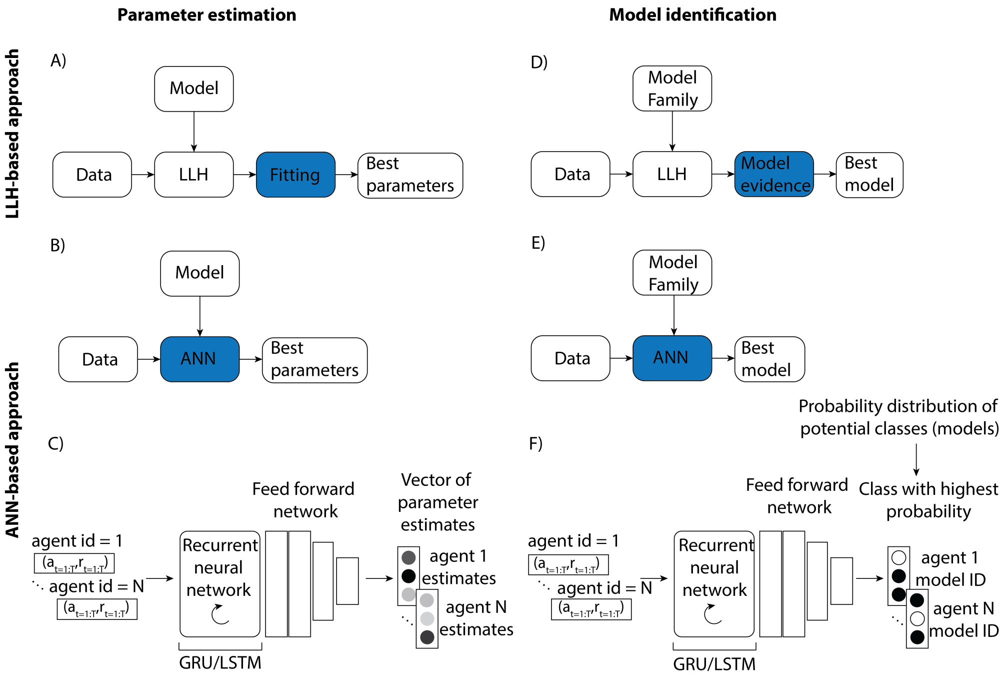


Figure 1: Artificial neural network (ANN) approach. A) Traditional methods rely on computing likelihood of the data under the given model, and optimizing the likelihood to derive model parameter estimates. B) The ANN is trained to map parameter values onto data sequences using a large simulated data set; the trained network can then be used to estimate cognitive model parameters based on new data without the need to compute or approximate likelihood. C) The ANN structure inspired by Dezfouli et al., 2019 is suitable for data with strong inter-trial dependencies: it consists of an RNN and fully connected feed-forward network, with an output containing ANN estimates of parameter values the data was simulated from for each agent. D) As in parameter estimation, traditional tools for model identification rely on likelihood to derive model comparison metrics (e.g. AIC, BIC) that are used to determine which model likely generated the data. E) ANN is instead trained to learn the mapping between data sequences and respective cognitive models the data was simulated from. F) Structure of the ANN follows the structure introduced for parameter estimation, with the key difference of final layer containing the probability distribution over classes representing model candidates, with highest probability class corresponding to the model the network identified as the one that likely generated the agent's data.

2.1 Parameter recovery

2.1.1 Benchmark comparison

First, we sought to validate our ANN method and compare its performance to existing methods by testing it on standard likelihood-tractable cognitive models of different levels of complexity in the same task: 2-parameter ($2P-RL$) and 4-parameter ($4P-RL$) reinforcement learning models commonly used to model behavior on reversal tasks (Gläscher et al., 2009; Hampton et al., 2006; Hauser et al., 2015; Peterson et al., 2009), as well as Bayesian Inference model (BI) and Bayesian Inference with Stickiness ($S-BI$) as an alternative model family that has been found to outperform RL in some cases (Costa et al., 2015; Perfors et al., 2011; Särkkä and Svensson, 2023). We estimated model parameters using multiple traditional methods for computing (maximum likelihood and maximum a-posteriori estimation; MAP and MLE) and approximating (Approximate Bayesian Computation; ABC) likelihood. We used the results of these tools as a benchmark for evaluating the neural network approach. Next, we estimated parameters of these models using two variants of RNNs: with gated recurrent units (GRUs) or Long-Short-Term-Memory units (LSTM).

We used the same held out data set to evaluate all methods (the test set the ANN has not observed yet, see simulation details). For each of the methods we extracted the best fit parameters, and then quantitatively estimated the method's performance as the mean squared error (MSE) between estimated and true parameters across all agents. Methods with lower MSE indicated better relative performance. All of the parameters were scaled for the purpose of loss computation, to ensure comparable contribution to loss across different parameters. To quantify overall loss for a cognitive model we averaged across all individual parameter MSE scores; to calculate fitting method's MSE score for a class of cognitive models (e.g. likelihood tractable models) we averaged across respective method's MSE scores for those models (See Methods for details about method evaluation).

First, we examined the performance of standard model-fitting tools (MLE, MAP and ABC). The standard tools yielded a pattern of results that are expected based on noisy, realistic-size data sets (with several-hundred trials per agent). Specifically, we found that MAP outperformed MLE (Fig. 2A, average MSEs: $MLE = .69, MAP = .37$), since the parameter prior applied in MAP regularizes the fitting process. ABC was also worse compared to MAP (Fig. 2A, average MSE: $ABC = .55$). While fitting process is also regularized in ABC, worse performance in some models can be attributed to signal loss that arises from approximation to the likelihood. Next, we focused on the ANN performance; our results showed that for each of the models, ANN performed better than or just as well as the traditional methods (Fig. 2A, average MSEs for different RNN variants: $GRU = .32, LSTM = .35$). Better network performance was more evident for parameter estimation in more complex models (e.g. models with higher number of parameters such as 4P-RL and S-BI; average MSE across these 2 models: $MLE = .98, MAP = .46, ABC = .74, GRU = .37, LSTM = .44$).

Next, we visualized parameter recovery. We found that for each of the cognitive models the parameter recovery was largely successful (Spearman ρ correlations between true parameter values and estimated values: β $\rho_{MAP}, \rho_{GRU} = [.88, .91]$, α^+ $\rho_{MAP}, \rho_{GRU} = [.51, .54]$, α^- $\rho_{MAP}, \rho_{GRU} = [.87, .87]$, κ: $\rho_{MAP}, \rho_{GRU} = [.68, .78]$, Fig. 2B; all correlations were significant at $p < .001$). For conciseness, we only show recovery of the more complex model parameters from the RL model family (and only MAP method as it performed better compared to ABC and MLE, as well as only GRU since it performed better than LSTM), as we would expect a more complex model to emphasize superiority of a fitting method more clearly compared to simpler models. Recovery plots of the remaining models (and respective fitting methods) can be found in supplementary materials. Our results suggest that 1) ANN performed as well or better than traditional methods in parameter estimation based on the MSE loss; 2) more complex models may limit accuracy of parameter estimation in traditional methods that neural networks appear to be more robust against. We note that for the $4P-RL$ model, parameter recovery was noisy for all methods, with some parameters being less recoverable than others (e.g. α^+ , Fig. 2B). This is an expected property of cognitive models applied to realistic-sized experimental data as found in most human experiments (i.e. a few hundred trials per participant).

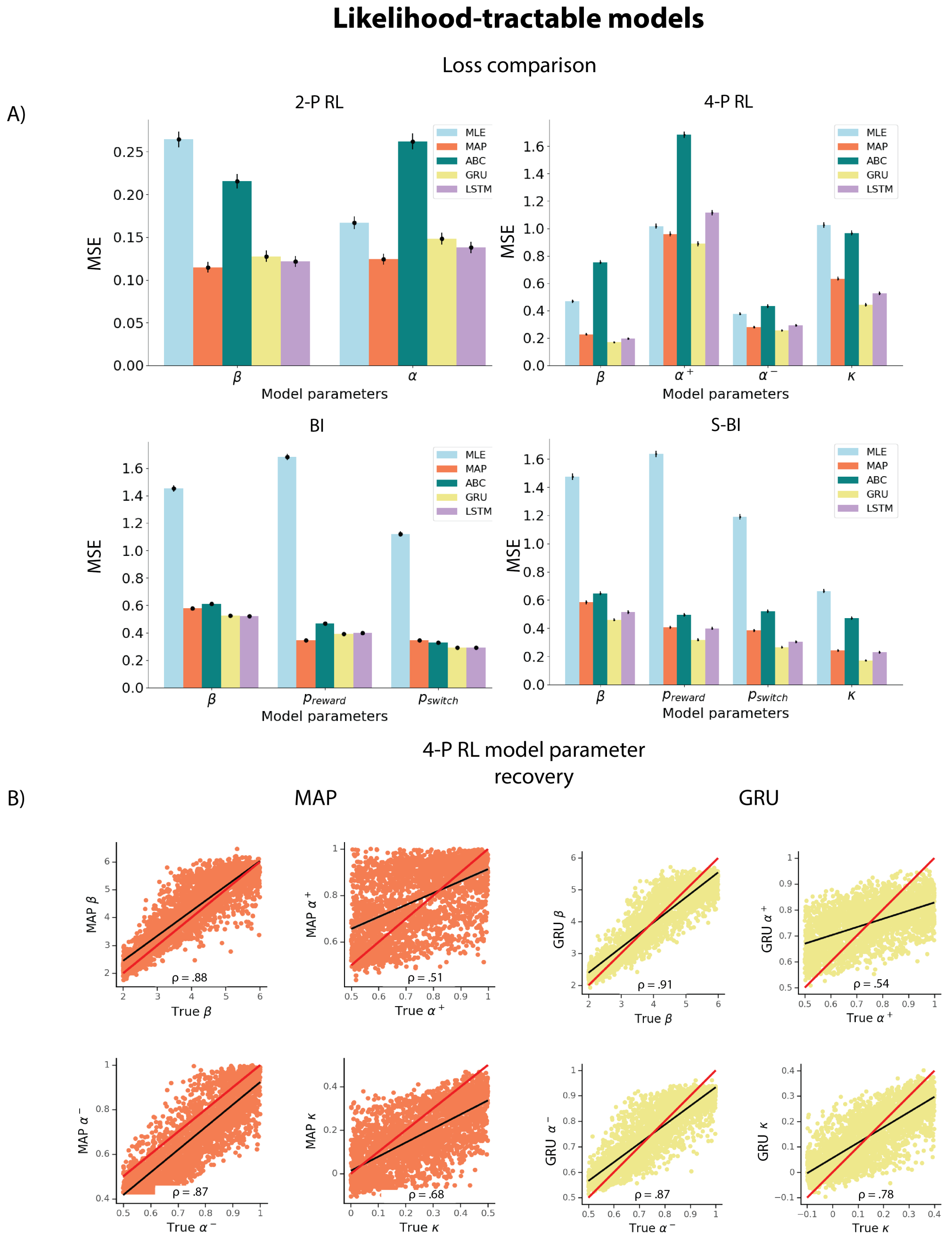


Figure 2: A) Parameter recovery loss from the held out test set for the tractable-likelihood models (2P-RL, 4P-RL, BI, S-BI) using each of the tested methods. Loss is quantified as the mean squared error (MSE) based on the discrepancy between true and estimated parameters. Bars represent loss average for each parameter across all agents, with errorbars representing standard error. B) Parameter recovery from the 4P-RL model using MAP and GRU. ρ values represent Spearman ρ correlation between true and estimated parameters. Red line represents a unity line ($x = x$) and black line represents a least squares regression line. All correlations were significant at $p < .001$.

2.1.2 Testing in cognitive models with intractable likelihood

Next, we tested our method in two examples of computational models with intractable likelihood. As a comparison method, we implemented Approximate Bayesian Computation (ABC), alongside our ANN approach to estimate parameters. The two example likelihood-intractable models we used had in common the presence of a latent state which conditioned sequential updates: RL with latent attentive state ($RL-LAS$), and a form of non-temporal hierarchical reinforcement learning (HRL, Eckstein and Collins, 2020). Since we cannot fit these models using MAP or MLE we used only ABC as a benchmark. Because we found LSTM RNN to be more challenging to train and achieve similar results when compared to GRU, we focused on GRU for the remainder of comparisons. We found that average MSE was much lower for the neural network compared to ABC for both RL-LAS (Fig. 3A, average MSEs: $ABC = .62, GRU = .21$) and HRL (Fig. 3A, average MSEs: $ABC = .28, GRU = .19$). Spearman correlations were noisier for ABC compared to GRU in both models (Fig. 3B, **RL-LAS** : β $\rho_{ABC}, \rho_{GRU} = [.72, .91]$, α $\rho_{ABC}, \rho_{GRU} = [.83, .95]$, T $\rho_{ABC}, \rho_{GRU} = [.5, .81]$; **HRL** : β $\rho_{ABC}, \rho_{GRU} = [.86, .89]$, α $\rho_{ABC}, \rho_{GRU} = [.85, .9]$; all correlations were significant at $p < .001$). Furthermore, some parameters were less recoverable than others (e.g. the T parameter in RL-LAS model, which indexed how long participants remained in an inattentive state); this might be in part due to less straightforward effect of T on behavior; see supplementary materials (Fig. S6). Note that in order to obtain our ABC results we had to perform an extensive exploration procedure to select summary statistics - ensuring reasonable ABC results. Indeed, the choice of summary statistics is not trivial and represents an important difficulty of applying ABC (Sunnåker et al., 2013), that we can entirely bypass using our new neural network approach.

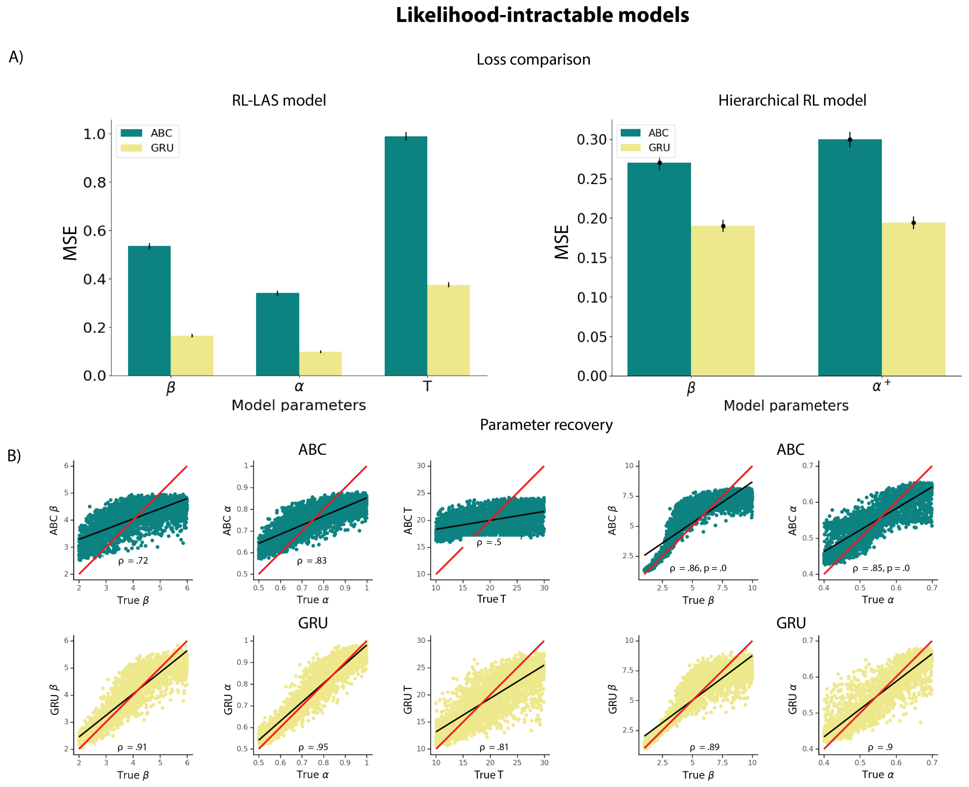


Figure 3: A) Parameter recovery loss from the held out test set for the intractable-likelihood models (RL-LAS, HRL) using ABC and GRU network. Loss is quantified as the mean squared error (MSE) based on the discrepancy between true and estimated parameters. Bars represent MSE average for each parameter across all agents, with errorbars representing standard error. B) Parameter recovery from the RL-LAS and HRL models using ABC (green) and GRU network (yellow). ρ values represent Spearman ρ correlation between true and estimated parameters. Red line represents a unity line ($x = x$) and black line represents a least squares regression line.All correlations were significant at $p < .001$.

2.2 Model identification

We also tested the use of our ANN approach for model identification. Specifically, we simulated data from different cognitive models, and trained the network to make a prediction regarding which model most likely generated the data out of all model candidates. The network architecture was identical to the network used for parameter estimation, except that the last layer became a classification layer (with one output unit per model category) instead of a regression layer (with one output unit per target parameter).

For models with tractable likelihood, we performed the same model identification process using AIC (Akaike, 1998) that relies on likelihood computation, penalized by number of parameters, to quantify model fitness as a benchmark. We note that another common criterion, BIC (Wei and Jiang, 2022), performed more poorly than AIC in our case.The best fitting model is identified based on the lowest AIC score - a successful model recovery would indicate that the true model has the lowest AIC score compared to other models fit to that data. To construct the confusion matrix, we computed best AIC score proportions for all models, across all agents, for data sets simulated from each cognitive model (Fig: 4; see methods).

As shown in Figure 4A, model identification performed using our ANN approach was better compared to the AIC confusion matrix, with less ”confusion” - lower off-diagonal proportions compared to diagonal proportions (correct identification). Model identification using AIC is likely in part less successful due to some models being nested in others (e.g. $2P-RL$ in $4P-RL$, BI in $S-BI$). Specifically, since AIC score represents a combination of likelihood and penalty incurred by the number of parameters it is possible that the data from more complex models is incorrectly identified as better fit by a simpler version of that model (e.g. the model with fewer parameters; an issue which would be more pronounced if we used a BIC confusion matrix). The same phenomenon is observed with the network, but to a much lesser extent, showing better identification out of sample - even for nested models. Furthermore, the higher degree of ANN misclassification observed for $BI/S-BI$ was driven by $S-BI$ simulations with stickiness parameter close to 0, which would render the BI and $S-BI$ non-distinguishable (Fig. S7).

Because we cannot compute the likelihood for our likelihood-intractable models, there is no standard metric for model comparison; thus, we only report the confusion matrices obtained from our ANN approach. In the first confusion matrix we performed model identification for $2P-RL$ and $RL-LAS$, as we reasoned these two models differ by only one mechanism (occasional inattentive state), and thus could potentially pose the biggest challenge to model identification. In the second confusion matrix, we included all models used to simulate data on the HRL task (*HRL model*, *Bayesian inference model*, *Bayesian inference with stickiness model*). In both cases, the network successfully identified the correct models as true models, with a very small degree of misidentification, mostly in the nested models. Based on our benchmark comparison to AIC, and the proof of concept identification for likelihood intractable models, our results indicate that the ANN can be leveraged as a valuable tool in model identification.

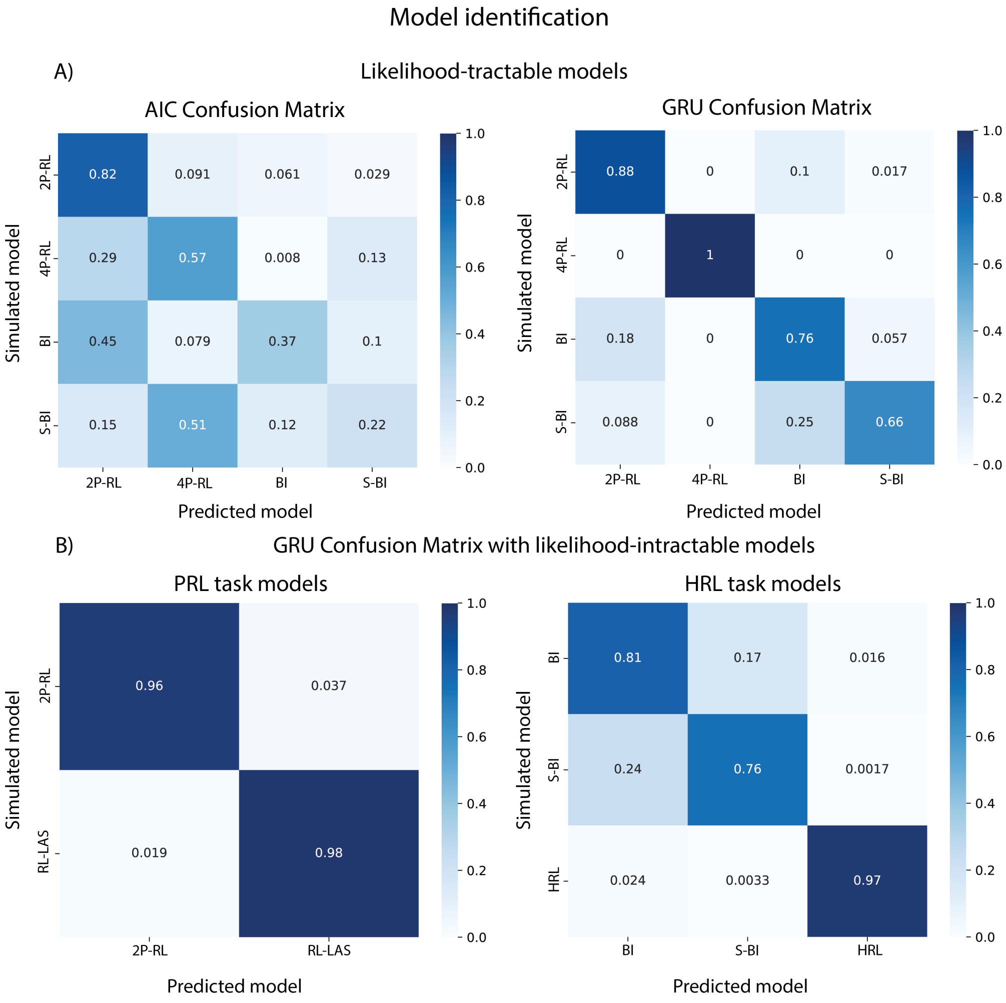


Figure 4: Model identification results.A) Confusion matrix of likelihood-tractable models from PRL task based on 1) likelihood/AIC metric, and 2) ANN identification. AIC confusion matrix revealed a much higher degree of misclassification (e.g. true simulated model being incorrectly identified as a different model). B) Confusion matrix of likelihood-intractable models using ANN (2P-RL and RL-LAS models were simulated on the PRL task; HRL, BI and S-BI models were simulated on the HRL task).

2.3 Robustness test: influence of different input trial lengths

ANNs are sometimes known to fail catastrophically when data is different from the training distribution in minor ways (Liang et al., 2017; Moosavi-Dezfooli and Alhussein Fawzi, 2017; Nguyen et al., 2015; Szegedy et al., 2013). Thus, we investigated the robustness of our method to differences in data format we might expect in empirical data, such as different lengths of trials across participants. Specifically, we conducted robustness experiments by varying the number of trials in each individual simulation contributing to training or test sets, fixing the number of agents in the training set.

To evaluate the quality of parameter recovery, we used the coefficient of determination score (R^2) which normalizes different parameter ranges. We found that the ANNs trained with a longer trial length reach high R^2 scores in long test trials. However, their performance suffers significantly with shorter test trial lengths. The results also show a similar trend in model identification tasks except that training with longer trial lengths doesn't guarantee a better performance. For instance, the classification accuracy between HRL task models of the ANN trained with 300 trials reaches 87% while the ANN trained with 500 trials is 84%.

Data-augmentation practices in machine learning increase robustness of models during training (Shorten and

Khoshgoftaar, 2019) by introducing different types of variability in the training data set (e.g. adding noise, different data sizes). Specifically, slicing time-series data into sub-series is a data-augmentation practice that increases accuracy (Iwana and Uchida, 2021). Thus, we trained our ANN with the fixed number of simulations of different trial lengths. As predicted, we found that the ANNs trained with a mixture of trial lengths across simulations (purple line) consistently yielded better performance across different lengths of test trials for both parameter recovery and model identification (Fig. 5A,B).

We also tested the effects of incorrect prior assumptions about the parameter range on method performance. Specifically we 1) trained the network using data simulated from a narrow range of parameters (theoretically informed) and 2) trained the network based on broader range of parameter values. Next, we tested both networks in making out-of-sample predictions for test data sets that were simulated from narrow and broad parameter ranges respectively. The network trained using a narrow parameter range made large errors at estimating parameters for data simulated outside of the range it was trained on; training the network on a broader range overall resulted in smaller error, with some loss of precision for the parameter values in range of most interest (e.g. the narrow range of parameters the alternative network is trained on). We observed similar results with MAP, where we specified narrow/broad prior (where narrow prior would place high density on a specific parameter range). Notably, training the network using a broader range of parameters while oversampling from a range of interest yielded more accurate parameter estimation compared to MAP with broad priors (Approach described in Fig. S9).

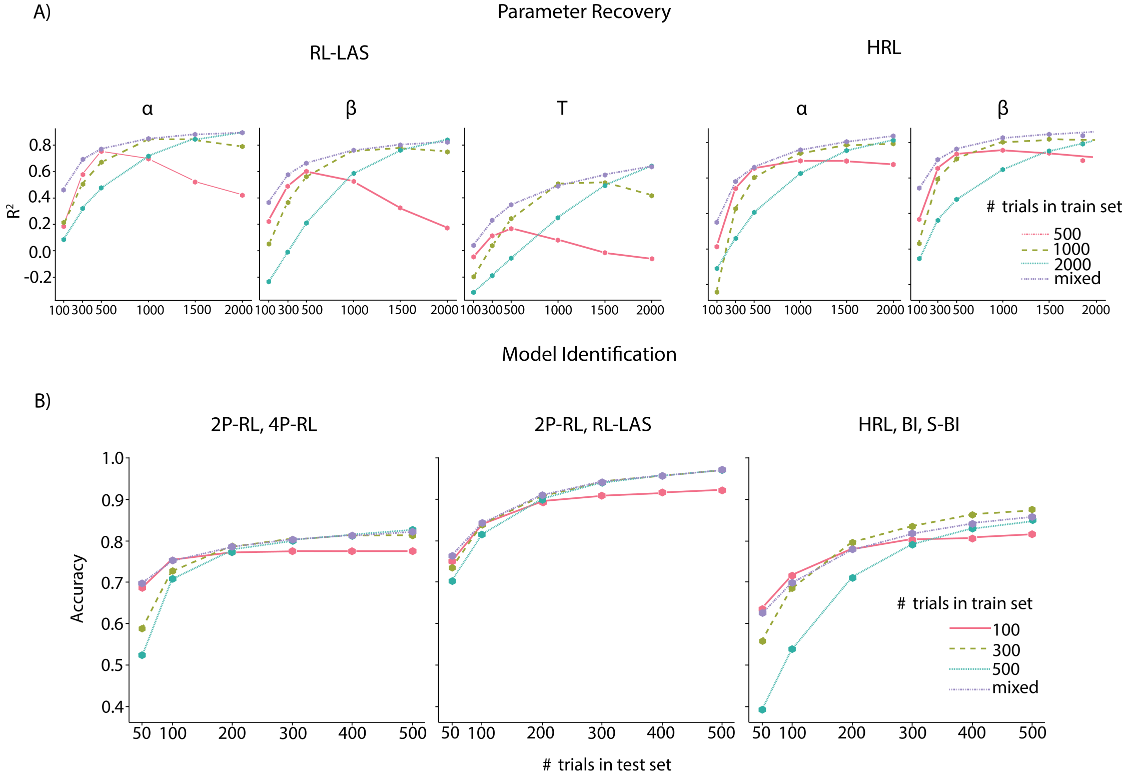


Figure 5: Robustness checks using different training (different line colors) and testing (x-axis) trial lengths. A) Parameter estimation in both RL-LAS and 2P-RL show that training with a mixture of trial lengths (purple line) yields more robust out-of-sample parameter value prediction compared to fixed trial lengths. B) Best model identification results, performed on different combinations of model candidates, were also yielded by mixed trial length training. The number of agents/simulations used for training was kept constant across all the tests (N agents = 30k).

3 Discussion

Our results demonstrate that artificial neural networks (ANNs) can be successfully and efficiently used to estimate best fitting free parameters of likelihood-intractable cognitive models, in a way that is independent of likelihood approximation. ANNs also show remarkable promise in successfully arbitrating between competing cognitive models in model identification. While our method leverages "big data" techniques, it does not require large experimental data sets: indeed, the large training set used to train the ANNs is obtained purely through efficient and fast model simulation. Thus, our method is applicable to any standard cognitive data set with normal number of participants and trials per participants. Furthermore, while our method requires some ability to work with ANNs, it does not require any advanced mathematical skills, making it largely accessible to the broad computational cognitive modeling community.

Our method takes a different approach from other attempts at using neural networks for fitting computational cognitive models. Work by Fengler et al., 2021 showcased Likelihood Approximation Networks (LANs) as a method that approximates likelihood of sequential sampling models, and recovers posterior parameter distributions with high accuracy; more recently, Boelts et al., 2022 used a similar approach with higher data efficiency. These methods attempt to estimate the likelihood of the cognitive models, and consequently need to train the ANNs on an approximation of the likelihood, obtained through ABC-like approaches. While their approach has the benefit of enabling researchers to leverage the broad set of likelihood-dependent tools (including for example hierarchical Bayesian modeling) – something our approach doesn't afford– the need for an approximate likelihood for training makes their approach impractical or inapplicable in many instances of likelihood intractable models. Unlike the work by Fengler et al., 2021 and Boelts et al., 2022, our approach focused on bypassing likelihood approximation entirely, training the network to derive parameter estimates based on data sequences alone. Thus, our approach should be applicable in a broader class of models.

Other approaches have used ANNs for different purposes than fitting cognitive models (Thompson et al., 2022). For example, Dezfouli et al., 2019 leveraged flexibility of RNNs (which inspired our network design) to map data sequences onto separable latent dimensions that have different effects on decision-making behavior of agents, as an alternative to cognitive models that make more restrictive assumptions. Similarly, work by Ger et al., 2023 also used RNNs to estimate RL parameters and make predictions about behavior of RL agents. Our work goes further than this approach in that it focuses on both parameter recovery and model identification of models with intractable likelihood, without relying on likelihood approximation. Furthermore, multiple recent papers (Eckstein et al., 2023; Ji-An et al., 2023) use ANNs as a replacement for cognitive models, rather than as a tool for supporting cognitive modeling as we do, demonstrating the number of different ways ANNs are taking a place in computational cognitive science.

It is important to note that while ANNs may prove to be a useful tool for cognitive modeling, one should not expect that their use immediately fixes or overrides all issues that may arise in parameter estimation and model identification. For instance, we have observed that while ANNs outperformed many of the traditional likelihood-based methods, recovery for some model parameters was still noisy (e.g. Fig. 2). This is a property of cognitive models when applied to experimental applied to data sets that range in hundreds of trials. Similarly, often times model parameters are not meaningful in certain numerical ranges, and sometimes model parameters trade off in how they impact behavior through mathematical equations that define the models - making the parameter recovery more challenging. This is to say ANNs should be treated as a useful tool that is only useful if the researchers apply significant forethought to developing appropriate, identifiable cognitive models.

Relatedly, we have shown that the parameter estimation accuracy varies greatly as a function of the parameter range the network was trained on, along with whether the underlying parameter distribution of the held out test-set is included in that range or not. This is an expected property of ANNs that are known to underperform when the test data systematically differs from training examples (Liang et al., 2017; Nguyen et al., 2015; Szegedy et al., 2013). As such, the range of parameters/models used for inputs constitutes a form of prior that constrains the fit, and it is important to carefully specify it with informed priors (as is done with other methods, such as MAP). We found that training the network using a broader parameter range, while heavily sampling from a range of interest (e.g. plausible parameter values based on previous research) affords both accurate prediction for data generated outside of the main expected range, with limited loss of precision within the range of interest. This kind of practice is also consistent with practices in computational cognitive modeling, where a researcher might specify (e.g. using a prior) that parameter might range between two values, with most falling within a certain, more narrow range.

We compared our artificial neural network approach against existing methods that are commonly used to estimate parameters of likelihood-intractable models (e.g. ABC, Sisson et al., 2018; Sunnåker et al., 2013). While ABC provides a workaround solution, it also imposes significant limitations. Specifically, it is more suitable for data with no sequential-dependencies, and the accuracy of parameter recovery is largely contingent on selection of appropriate summary statistics, which is not always a straightforward problem. Alternative approximation methods (e.g. particle filtering (Djuric et al., 2003); density estimation (Minka, 2013)) may prove to be more robust, but frequently require more advanced mathematical knowledge. Thus, developing more accessible and robust methods is critical for advances in computational modeling and cognitive science, and the rising popularity of deep learning puts neural networks forward as useful tools for this purpose. Our method also offers an advantage of requiring very little computational power. The aim of the project at its current state was not to optimize our ANN training in terms of time and computing resources; nevertheless, we used Nvidia V100 GPUs with 25 GB memory and required at most 1 hour for model training and predictions. This makes the ANN tool useful, as it requires a low amount of computing resources and can be done fast and inexpensively. All of our code will be shared on GitHub.

Since we aimed to provide a proof of concept and comparison to traditional methods, our experiments focused mainly on extensive tests using synthetic data. A critical next step will be to further validate our approach using empirical data (e.g. participant data from the tasks). Similarly, we relied on RNNs due to their flexibility and capacity to handle sequential data. However, it will be important to explore different structures, such as transformers (Devlin et al., 2018), for potentially improved accuracy in parameter recovery/model identification, as well as alternative uses in cognitive modeling. Furthermore, we included both RL and Bayesian inference models to demonstrate our approach can work with different classes of computational models. Future work will include additional models (e.g. sequential decision making models) to further test robustness of our approach.

In conclusion, we propose a novel, accessible method to perform parameter and model identification across a broad class of computational cognitive models for which existing methods were inapplicable. Our work should have important impact on allowing researchers to quantitatively test a broader family of theories than previously possible.

4 Methods

4.1 Tasks

Probabilistic reversal learning task. We have simulated data from different models (see the Models section) on a simple probabilistic reversal learning task (PRL; Cools et al., 2002). In the task, an agent chooses between two actions on each trial, and receives binary outcome ($r = 1$ [reward] or $r = 0$ [no reward]). One of the two actions is correct for a number of trials; a correct action is defined as the action that gets rewarded with higher probability (e.g. $p(r = 1|action = correct) = 0.80$), with $1 - p$ probability of getting no reward if selected. After a certain number of trials, the correct action reverses; thus the action that was previously rewarded with low probability becomes the more frequently rewarded one (Fig: 1). This simple task (and its variants) have been extensively used to provide mechanistic insights into learning from reinforcement, inferring probabilistic structure of the environment, and people's ability (or failure) to update the representation of a correct choice.

Hierarchical reinforcement learning task. We developed a novel task environment that can be solved using a simple but plausible model with intractable likelihood. In this task, an agent observes N arrows (in different colors), each pointing at either left or right direction. The agent needs to learn which arrow is the correct one, by selecting an action that corresponds to either left or right side (consistent with the direction the arrow is pointing at) in order to get rewarded. Selecting the side the correct arrow is pointing at rewards the agent with high probability ($p = .9$); choosing an action by following direction of other arrows leads to no reward ($r = 0$) with same high probability. The correct arrow changes unpredictably in the task, which means that the agent must keep track of which arrow most reliably leads to the reward, and update accordingly upon the change. We refer to this task structure as hierarchical because the choice policy (left/right) depends on the higher-level rule (color) agents choose to follow.

4.2 Cognitive Models

4.2.1 PRL task models

.

We implemented multiple models of the PRL task to test the artificial neural network (ANN) approach to parameter estimation. First, we cover the benchmark models; these are the models that we can fit using traditional methods

(MLE, MAP), as well as the ANN, to ensure that we can justify using the ANN if it performs at least just as well as (or better than) the traditional methods.

Reinforcement learning models family.

Two-parameter reinforcement learning model. We simulated artificial data on the PRL task using a simple 2-parameter reinforcement learning model (2P-RL). The model assumes that the agent tracks the value of each action contingent on the reward history, and uses these values to inform the action selection on each trial.

The model uses simple delta rule to update action values on each trial upon outcome observation, by first computing the reward prediction error (RPE, δ) as the discrepancy between the expected and the observed outcome, and then adjusting the value of the chosen action using the RPE scaled by the learning rate (α) (Sutton and Barto, 2018):

$$\begin{aligned} \delta &= r - Q_t(a) \\ Q_{t+1}(a) &= Q_t(a) + \alpha\delta \end{aligned} \tag{1}$$

We also allowed for counterfactual updating, where the value of the non-chosen action also gets updated on each trial (Eckstein et al., 2022; Hauser et al., 2015):

$$\begin{aligned} \delta_{\text{unchosen}} &= (1-r) - Q_t(1-a) \\ Q_{t+1}(1-a) &= Q_t(1-a) + \alpha\delta_{\text{unchosen}} \end{aligned} \tag{2}$$

The action values are transformed into action probabilities using the softmax function, thus defining a policy where actions with higher value are chosen with higher probabilities. The β parameter controls how deterministic the choices are with higher values of β corresponding to more deterministic choices:

$$P(a) = \frac{\exp(\beta\, Q_t(a))}{\sum_{i=1}^{n_A} \exp(\beta\, Q_t(a_i))} \tag{3}$$

The 2p-RL model contained following free parameters: learning rate (α) and softmax beta (β).

Four-parameter reinforcement learning model. The four parameter RL (4P-RL) model follows the same updating and policy structure as the 2-parameter RL, with 2 main differences. The 4P-RL model differentiates between positive and negative feedback (Niv et al., 2012), by using different learning rates - α^+ and α^- for updating action values after positive and negative outcomes respectively:

$$Q_{t+1}(a) = \begin{cases} Q_t(a) + \alpha^+\delta & \text{if } \delta\ > 0 \\ Q_t(a) + \alpha^-\delta & \text{if } \delta\ \leq 0 \end{cases}$$

Furthermore, 4P-RL model also includes the stickiness parameter κ which captures the tendency to repeat choice from the previous trial:

$$P(a) \propto exp(\beta Q + \kappa\,\text{same}(a, a_{t-1})) \tag{4}$$

Like in the 2P-RL we also included counterfactual updating of values for non-selected actions. The 4P-RL model included following free parameters: positive learning rate (α^+), negative learning rate (α^-), softmax beta (β) and stickiness (κ).

Bayesian models family.

Bayesian inference model. Bayesian inference model (BI) assumes that an agent infers the latent state in the environment, updates the latent state based on new observations, and uses the inference process to make rewarding choices. For instance, in the PRL task, the agent infers a latent state corresponding to the correct action ($C_t : a_{right} = cor$ or $C_t : a_{left} = cor$) at time t. The agent tracks and updates their belief over which one of the two actions is currently the correct one based on 1) their estimate of the switch frequency (p_{switch}) and 2) how noisy the reward is (p_{reward}) from the history of observations up to the previous trial H_{t-1}. On each trial, the belief is updated according to the Bayes rule - based on the prior belief (agent's model of the task) and likelihood of observed evidence (the outcome given the choice):

$$p(C_t = i|r_t, a_t, H_{1:t-1}) = \frac{P(r_t|C_t = i, a_t)P(C_t = i|H_{1:t-1})}{\sum j(P(r_t|C_t = j, a_t)P(C_t = j|H_{1:t-1})}$$

where i and j are in [left/right], $p(C_t = i|H_{1:t-1})$ is the prior probability, and $p(r_t|C_t = i, a_t)$ is the likelihood of outcome given the action. The likelihood is defined in accordance to whether the choice matches the latent state:

$$p(r_t = 1|a_t = i, C_t = i) = p_{reward}$$

where p_{reward} is the parameter controlling the probability of receiving the reward given the choice of correct action. Posterior belief for the correct action is updated to a prior belief for the upcoming trial in accordance with the p_{switch} parameter, which determines the probability that the correct action might have reversed on the current trial:

$$p(C_{t+1} = i|H_{1:t-1}) = (1 - p_{switch})p(C_t = i|H_{1:t-1}) + p_{switch}(1 - p(C_t = i|H_{1:t-1}))$$

Like in the RL models, the action selection in Bayesian models also followed the softmax policy; however, instead of being informed by the Q values the action probabilities were determined by the belief W given the choice and reward history H and the choice parameter β:

$$W_{t+1} = p(C_{t+1} = i|H_{1:t})$$

$$P(a_{t+1}) = \frac{\exp(\beta\, W_i(t+1))}{\sum_{i=j}\exp(\beta\, W_j(t+1))}$$

The BI model included following parameters: inferred probability of reward given the action determined by the current belief (p_{reward}), likelihood of the correct action reversing (p_{switch}) and softmax beta (β).

Bayesian inference model with stickiness. We also added a variation of the Bayesian inference model that accounts for sticky choice behavior (e.g. repeating actions) by introducing a stickiness parameter κ that augments the belief associated with the action chosen on the previous trial:

$$W_{t+1} = p(C_{t+1} = i|H_{1:t}) + \kappa(i = a_t)$$

Intractable likelihood

As a proof of concept, we implemented a simple model that assumes a latent state of agent's attention (engaged/disengaged). This model can't be fit using methods that rely on computing likelihood. While models can have intractable likelihood for a variety of reasons, we focused on leveraging latent variables (e.g. attention state), that are not readily observable in the data. Thus, in the data that is being modeled, only the choices are observed - but not the state the agent was in while executing the choices. The learned choice value which affects the choice likelihood depends on the trial history, including which state the agent was in. Thus, if there are 2 such states, there are 2^N possible sequences that may result in different choice value estimates after N trials. To estimate choice values and likelihood on any given trial one must integrate over the uncertainty of an exponentially increasing latent variable - thus making the likelihood intractable.

RL and latent engagement state . We simulated a version of a 2p-RL model for a probabilistic reversal learning (PRL) task that also assumes that an agent might occupy two of the latent attention states — engaged or disengaged— during the task (RL-LAS). The model assumes that in the engaged state an agent behaves in accordance with the task structure (e.g. tracks and updates values of actions, and uses action values to inform action selection). In the disengaged state, an agent behaves in a noisy way, in that 1) it does not update the Q value of actions , and 2) chooses between the two actions randomly (e.g. uniform policy) instead of based on their value (e.g. through softmax). [1] The agent can shift between different engagement states at any point throughout the task, and the transition between the states is controlled by a parameter τ. Specifically, for each agent we initialized a random value T between 10 and 30 (which roughly maps onto approximately how many trials an agent spends in a latent attention state), and then

[1] Note that assumption 1) is different from a previous version of the model our group considered (Li, Shi, Li, and Collins, 2023; Li, Shi, Li, and Collins, 2023), and is the core assumption that renders the likelihood intractable.

used a non-linear transformation to compute τ: 1-(1/T). The value of τ, thus quantifies the probability of transitioning between the two states. The agent was initialized to be in an attentive state at the onset of trials.

The likelihood of this model can be computed:

$$\begin{aligned}\mathcal{L}(\theta) &= \sum_{t=1}^{T} \log \mathbb{P}(a_t|h_t,\overline{h}_{t-1},\theta)\\ &= \sum_{t=1}^{T} \log\Big(\sum_{l} \mathbb{P}(a_t|h_t,ls_t=l;\theta)\mathbb{P}(ls_t=l,\overline{h}_{t-1};\theta)\Big)\end{aligned}$$

where ls = latent state, l ∈ { 0 = disengaged state, 1 = engaged state }, $\overline{h}_{t-1}$ corresponds to the history of actions and rewards up to the trial t. However, it is in practice intractable, because of the sum over latent states in the equation, which cannot be factored out.

4.2.2 Cognitive models of the HRL task

Bayesian models of the HRL task .

Bayesian models of the HRL task assume an inference process over the latent variable of which arrow is currently the valid arrow, and thus which side (R/L) (given the current trial's set of arrows) is most likely to result in positive outcome. The inference relies on the generative model of the task determined by parameters p_{switch} and p_{reward}, history of trial observations O_t, set of arrows and stochastic choice based on this inference. Initial prior belief over arrows is initialized uniformly $prior = 1/nA$, where nA corresponds to the number of arrows.

To determine the agent policy over arrows at trial t, we first implemented a softmax function with decision parameter β and prior belief of which arrow is the correct one; once the arrow is selected , the agent implements an ε-greedy policy conditioned on the selected arrow A_t to choose a R/L side:

$$P(side(A_t)|A_t) = 1-\varepsilon$$

Likelihood $p(r_t = 1|A_t, side(A_t))$ and posterior are then updated into the prior belief for the next trial using the p_{switch} model of the task parameter:

$$p(C_{t+1} = i|O_{1:t-1}) = (1-p_{switch}) * p(C_t = i|O_{1:t-1}) + p_{switch}(1-p(C_t = i|O_{1:t-1}))$$

This belief is subsequently used to inform arrow choices on the next trial. This model differs from the Bayesian Inference model for the probabilistic task in that 1) p_{reward} and p_{switch} parameters are not free/inferred and 2) the choice of the side is stochastic, allowing for a potential lapse in selecting the side that is not consistent with the selected arrow. This model, thus has following free parameters: decision parameter β and noise parameter ε. Like in the in Bayesian inference model for the PRL task, we also tested the model variant with stickiness κ parameter that biases beliefs associated with the arrow/side chosen on the previous trial. Both models have tractable likelihoods.

Hierarchical reinforcement learning . We also simulated a simple hierarchical reinforcement learning (HRL) model to simulate the performance on a HRL task (see tasks section, S1). This model assumes that an agent tracks the value of each of the arrows, and chooses between the arrows noisily:

$$P(arrow) = \frac{\exp(\beta\, Q_t(arrow))}{\sum_{i=1}^{n_A} \exp(\beta\, Q_t(arrow_i))} \tag{5}$$

We have also explored the model with an assumption that an agent has a tendency to repeat the choice from the previous trial, captured by the stickiness parameter κ:

$$P(arrow) = \frac{\exp(\beta\, Q_t(arrow) + \kappa(arrow = arrow_{t-1}))}{\sum_{i=1}^{n_A} \exp(\beta\, Q_t(arrow_i))} \tag{6}$$

Once the agent chooses the arrow,it greedily chooses the direction based on which side (left/right) the arrow is pointing at (observable). Note that we only know the side the agent selected (left/right), because the arrow the agent chooses is non-observable. The agent then observes an outcome, and updates the value of the selected arrow based on the observed outcome:

$$Q_{t+1}(\text{arrow}) = Q_t(\text{arrow}) + \alpha(r - Q_t(\text{arrow}))$$

In the case of this model, the likelihood is intractable because of the need to integrate over uncertainty of what rule (which arrow) the agent followed on all of the past trials; because the integration exponentially increases with each time point, the likelihood is not tractable beyond the first several trials:

$$\begin{aligned}\mathcal{L}(\theta) &= \sum_{t=1}^{T} \log \mathbb{P}(a_t | h_t, \bar{h}_{t-1}, \theta) \\ &= \sum_{t=1}^{T} \log \mathbb{P}\Big(\sum_{c} \mathbb{P}(a_t | h_t, rule_t = c; \theta)\mathbb{P}(rule_t = c, \bar{h}_{t-1}; \theta)\Big)\end{aligned}$$

where a_t corresponds to the action dictating which side the agent selected (left/right), $\bar{h}_{t-1}$ corresponds to the task history encoding rewards, selected actions/sides, arrow directions, and c correspond to identity/color of the correct arrow.

4.3 Likelihood-dependent methods.

4.3.1 Maximum likelihood and Maximum a posteriori estimation

Maximum likelihood estimation (MLE) represents a cornerstone of modeling that leverages probability theory and estimation of likelihood ($P(D|M,\theta)$) of the data given the model parameters and assumptions (Myung, 2003). The parameter estimates are determined as the values that maximize the likelihood of the data:

$$\begin{aligned}\theta_{MLE} &= argmax P(D|\theta) \\ &= argmax \prod_{i} P(D_i|\theta) \\ &= argmax \sum_{i} log P(D_i|\theta)\end{aligned}$$

Thus, to estimate best fitting parameters via MLE, the likelihood of the data is computed and maximized with respect to parameter values via an optimization algorithm (often a blackbox one, such as fmincon in MATLAB or optimize.minimize from scipy toolbox in python). Maximum a posteriori estimation (MAP) relies on much the same principle, with an addition of a prior $p(\theta)$ to maximize the posterior:

$$\theta_{MAP} = argmax \sum_{i} log P(D_i|\theta) log P(\theta)$$

As a prior for the MAP approach, we used an empirical prior derived from the true simulating distribution of parameters (see supplement for details). We note that this gives an advantage to the MAP method above what would be available for empirical data, allowing MAP to provide a ceiling performance on the test set.

Because MAP and MLE rely on likelihood computation, their use is essentially limited to models with tractable likelihood. We used MAP and MLE to estimate parameters of tractable-likelihood models as one of the benchmarks against which we compared our ANN approach. Specifically, we fit the models to the test-set data used to compute the MSE of the ANN, and compared fit using the same metric across methods (see main text).

4.3.2 Likelihood approximation methods

Because models with tractable likelihood comprise only a small subset of all possible (and likely more plausible) models, researchers have handled the issue of likelihood intractability by implementing various likelihood approximation methods. While there are different likelihood approximation tools, such as particle filtering (Djuric et al., 2003) and assumed density estimation (Minka, 2013), we focus on Approximate Bayesian Computation (ABC; Lintusaari et al., 2017; Palestro et al., 2018; Sisson et al., 2018; Sunnåker et al., 2013), as it is more widely accessible and does not require more extensive mathematical expertise. ABC leverages large scale model simulation to approximate likelihood. Specifically, a large synthetic data set is simulated from a model, with parameters randomly sampled from a specific range for each agent. Summary statistics that describe the data (e.g average accuracy or variance in accuracy) are used to construct the empirical likelihood that can be used in combination with classic methods.

We implemented a basic form of ABC - the rejection algorithm (Sunnåker et al., 2013). This algorithm first samples a set of model parameters θ, simulates the data $\hat{D}$ from the model M using these parameters and computes the predetermined summary statistic $S(\hat{D})$ of the simulated data which we refer to as the sample. The summary statistics of the real data $S(D)$ and the sample $S(\hat{D})$ are then compared - if the distance between the two sets of summary statistics ρ is greater than the predetermined criterion ε, the sample is rejected:

$$\rho(S(\hat{D}), S(D)) \leq \varepsilon$$

The distance metric, like the rejection criterion, is determined by the researcher. The samples that are accepted are the samples with distance to the real data smaller than the criterion, resulting in the conclusion that parameters used to generate the sample data set can plausibly be the ones that capture the target data. Thus, the result of the ABC for each data set is a distribution of plausible parameter values which can be used to obtain point estimates via the mean, median, etc.

ABC is a valuable tool and has previously been combined with ANNs to aid likelihood approximation (Boelts et al., 2022; Fengler et al., 2021). However, it has serious limitations (Sunnåker et al., 2013). For instance, the choice of summary statistics is not a trivial problem, and different summary statistics can yield significantly different results. Similarly, in the case of rejection algorithm ABC, researchers must choose the rejection criterion which can also affect the parameter estimates. A possible way to address this is using cross validation to determine which rejection criterion is the best, but this also requires specification of the set of possible criteria values for the cross validation algorithm to choose from. Furthermore, one of ABC assumptions is independence of data points, which is violated in many sequential decision making models (e.g. reinforcement learning).

To compare our approach to ABC, we used network training set data as a large scale simulation data set, and then estimated parameters of the held out test set also used to evaluate the ANN.

To apply ABC in our case, we needed to select summary statistics that adequately describe performance on the task. We used the following summary statistics to quantify the agent for the models simulated on the PRL task:

- Learning curves: We computed agents' probability of selecting the correct action, aligned to the number of trials with reference to the reversal point. Specifically, for each agent we computed an average proportion of trials where a correct action was selected N trials before and N trials after the correct action reversal point, for all reversal points throughout the task. This summary statistic should capture learning dynamics, as the agent learns to select the correct action, and then experiences dip in accuracy once the correct actions switch, subsequently learning to adjust based on feedback after several trials.

- 3-back feedback integration: The 3-back analysis quantifies learning as well; however, instead of aligning the performance to reversal points, it allowed us to examine agents' tendency to repeat action selection from the previous trial contingent on reward history - specifically the outcome they observed on the most recent 3 trials. Higher probability of repeating the same action following more positive feedback indicates learning/sensitivity to reward as reported in Zou et al., 2022

- Ab-analysis: The Ab-analysis allowed us to quantify probability of selecting an action at trial t,contingent both on previous reward and action selection history (trials $t-2$ and $t-1$, Beron et al., 2021; Zou et al., 2022).

For the models simulated on a hierarchical task we used the learning curves as summary statistics (same as for the PRL), where reversal points were defined as the switch of the correct rule/arrow to follow. In addition, we quantified agent's propensity to stick with the previously correct rule/arrow, where the agent should be increasingly less likely to select the side consistent with the arrow that was correct before the switch as the number of trials since the switch increases. Similarly, we used a version of the 3-back analysis where the probability of staying contingent on the reward history referred to the probability of potentially selecting the same cue across the trial window, based on observed choices of the agent. All summary statistics are visualized in the supplementary figure S8.

4.3.3 Model comparison

To perform benchmark model comparison, we used the Akaike Information Criterion (AIC) metric (Akaike, 1998), commonly used to evaluate relative model fitness, with an aim of identifying the best model candidate that might have generated the data. The AIC score combines model log likelihood and number of parameters to quantify model fitness, while also penalizing for model complexity in order to prevent overfitting:

$$AIC = -2(LLH) + 2K$$

where K corresponds to the number of parameters. The model with the lowest AIC scores corresponds to the best fitting model for the given data. A related metric that is commonly used is the Bayesian Information Criterion (BIC, Schwarz, 1978), which considers the number of observations (N) as well, and similarly uses the lowest score to signal the best fitting model:

$$BIC = -2(LLH) + K * log(N)$$

We used AIC score as it outperformed BIC model comparison, and thus provided us with ceiling benchmark to evaluate the ANN.

To perform proper model comparison, it is essential to not only evaluate the model fitness (overall AIC/BIC score), but also to test how reliably the true models (that generated the data) can be identified/successfully distinguished from others. To do so, we constructed a confusion matrix based on the AIC score (Fig. 4A). We used the test set data simulated from each model, and then fit all candidate models to each of the data sets while also computing the AIC score for each fit. If the models are identifiable, we should observe that AIC scores for true models (e.g. the models the data was simulated from) should be the lowest for that model when it's fit to the data compared to other model candidates.

4.4 Artificial neural network-based method

4.4.1 Parameter recovery

To implement ANNs for parameter estimation we have used the relatively simple neural network structure inspired by the previous work (Dezfouli et al., 2019). In all experiments, we used 1 recurrent GRU layer followed by 3 fully connected dense layers with 2000 dimensional input embeddings (S1). To train the network, we simulated a training data set using known parameters. For each model, we used 30000 training samples, 3000 validation samples, and 3000 test samples that are generated from simulations separately. For probabilistic RL, the input sequence consisted of rewards and actions. For hierarchical RL, the sides (left/right) of three arrow stimuli are added to the rewards and actions sequences. The network output dimension was proportional to the number of model parameters. We used a *tanh* activation in the GRU layer, *reLu* activations in 2 dense layers, and a linear activation at the final output. Additional training details are given below:

- We used *He* normal initialization to initialize GRU parameters (He et al., 2015).
- We used the Adam optimizer with mean square error (MSE) loss and a fixed learning rate of 0.003. Early stopping (e.g. network training was terminated if validation loss failed to decrease after 10 epochs) was applied with a maximum of 200 epochs.
- We selected network hyperparameters with Bayesian optimization algorithms (Bergstra et al., 2013) applied on a validation set. Details of the selected values are shown in Supplementary Materials.

All of the training/validation was run using TensorFlow (Abadi et al., 2016). The training was performed on Nvidia V100 GPUs with 25 GB memory.

Network evaluation. The network predicted the values of parameter on the test set that is unseen in the training and validation. We also conducted robustness tests by varying trial numbers (input length).

To evaluate the output of both ANN and traditional tools we used the following metrics:

- Mean squared error (MSE): To evaluate parameter estimation accuracy we calculated a mean squared error between true and estimated model parameter across all agents. Prior to calculating MSE all parameters were normalized, to ensure comparable contribution to MSE across all parameters. Overall loss for a cognitive model (across all parameters) was an average of individual parameter MSE scores. Overall loss for a class of models (e.g. likelihood-tractable models) was an average across all model MSE scores.

- Spearman correlation (ρ): We used Spearman correlation as an additional metric for examining how estimated parameter values relate to true parameter values, with higher Spearman ρ values indicating higher accuracy. We paired Spearman correlations with scatter plots, to visualize patterns in parameter recoverability (e.g. whether there are specific parameter ranges where parameters are more/less recoverable).

- R-Squared (R^2 or the coefficient of determination): R-Squared represents the proportion of variance in true parameters that can be explained by a linear regression between true and predicted values. It thus indicates the goodness of fit of an ANN model. We calculated an R-Squared score for each individual parameters across all agents and used it as an additional evaluation for how well the data fit the regression model.

Alternative models. We have also tested the network with long short-term memory (LSTM) units since LSTM units are more complex and expressive than GRU units; nevertheless they achieved the similar performance as GRU units but are more computationally expensive, and thus we mostly focused on the GRU version of the model. Since LSTM worked, but not better than GRU, the LSTM results are reported in the supplementary materials.

4.4.2 Model identification

The network structure and training process were similar to that of the network used for parameter recovery, with an exception of the output layer that utilized categorical cross-entropy loss and a softmax activation function. The network validation approach was the same as the one we used for parameter recovery (e.g. based on the held-out test set). We also observed a better performance when training with various trial lengths.

4.4.3 Robustness test: influence of different input trial lengths

For all robustness experiments, we followed the same training procedures as described previously while varying the training data. The details of training data generation are given below:

Parameter Recovery We simulated 30,000 training samples with 2000 trials per simulation in the probabilistic reversal learning task. For shorter fixed trial lengths per training samples (e.g 500), we used the same training set truncated to the first 500 trials. To generate the training data with different trial lengths across training samples, we reused the same training set, with sequences of trials truncated to a given length. There were 6000 training samples of lengths 50, 100, 500, 1000, 1500, and 2000 trials, each.

Model Identification The process of data generation for model identification robustness checks was similar to parameter recovery. However, we only simulated 500 trials for each model because we found no significant increase in accuracy with higher trial numbers.

5 Acknowledgments

We thank Jasmine Collins, Kshitiz Gupta, Yi Liu and Jaeyoung Park for their contributions to the project. We thank Bill Thompson and all CCN lab members for useful feedback on the project. This work was supported by NIH R21MH132974.
Code available at the GitHub Repository.

6 Supplementary materials

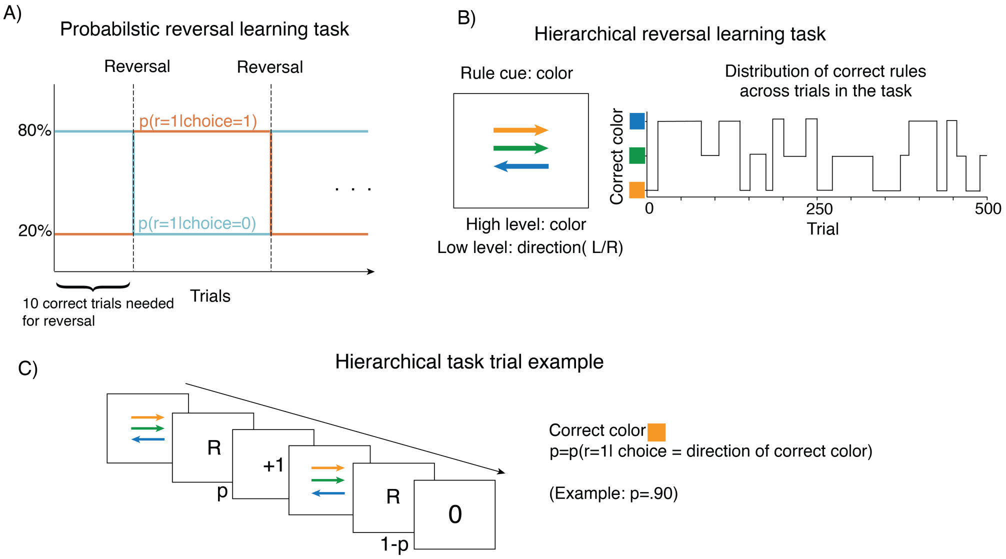

A)
Probabilstic reversal learning task
Reversal
Reversal
80%
20%
p(r=1|choice=1)
p(r=1|choice=0)
. . .
10 correct trials needed for reversal
Trials
B)
Hierarchical reversal learning task
Rule cue: color
High level: color
Low level: direction(L/R)
Distribution of correct rules across trials in the task
Correct color
0
250
500
Trial
C)
Hierarchical task trial example
R
p
+1
R
1-p
0
Correct color
p=p(r=1| choice = direction of correct color)
(Example: p=.90)

Figure S1: Tasks. A) Probabilistic reversal learning task. We simulated artificial agents using cognitive models of behavior on a Probabilistic reversal learning (PRL) task, which provides a dynamic context for studying reward-driven learning (Cools et al., 2001; Lawrence et al., 1999). In this task, an agent chooses between two actions, where one of the actions gets rewarded with higher probability ($p(r) = .80$) and one with lower $(1-p)$. After a certain number of correct trials, the reward probabilities of the two actions reverse. The task provides an opportunity to observe how agents update their model of the task (e.g. correct actions) based on observed feedback. B) Hierarchical reinforcement learning task. In this task, three differently colored arrows represent three potential rules an agent can follow when selecting one of the two actions (left/right) corresponding to the side the chosen arrow is pointing at. Selecting a side consistent with correct arrow is rewarded with probability $p = .90$. Correct arrow switches after a certain number of trials. The task provides a possibility to examine how following latent rules may shape agents' learning behavior.

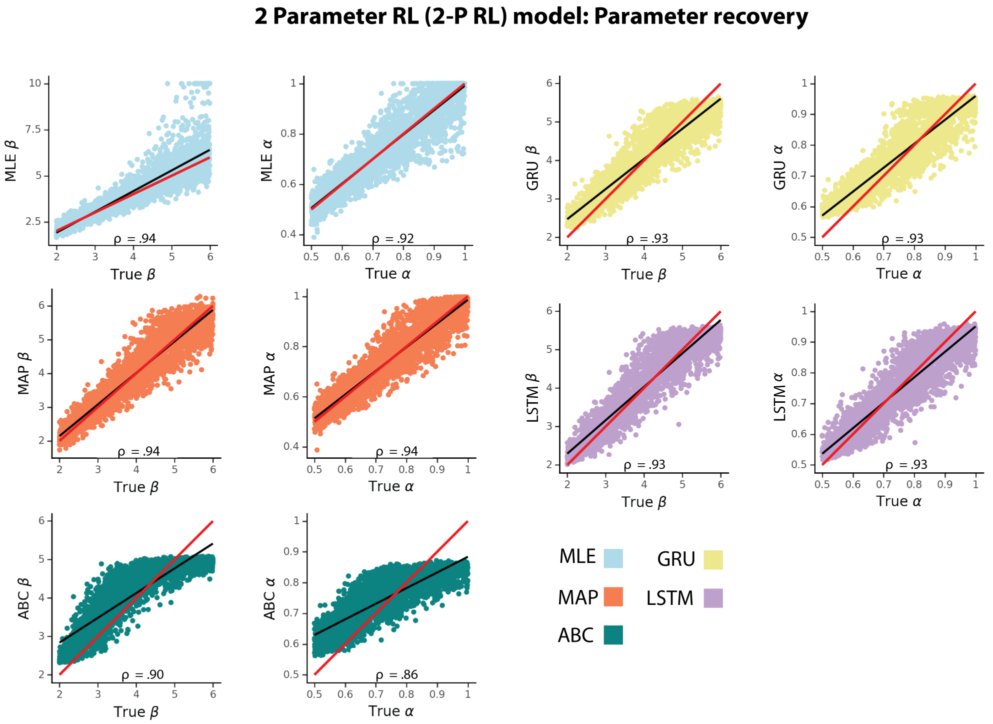


Figure S2: 2 Parameter RL (2P-RL) model parameter recovery using different fitting methods. ρ corresponds to Spearman correlation coefficient, red line represents a unity line (x=x), and black line represents a least squares regression line.

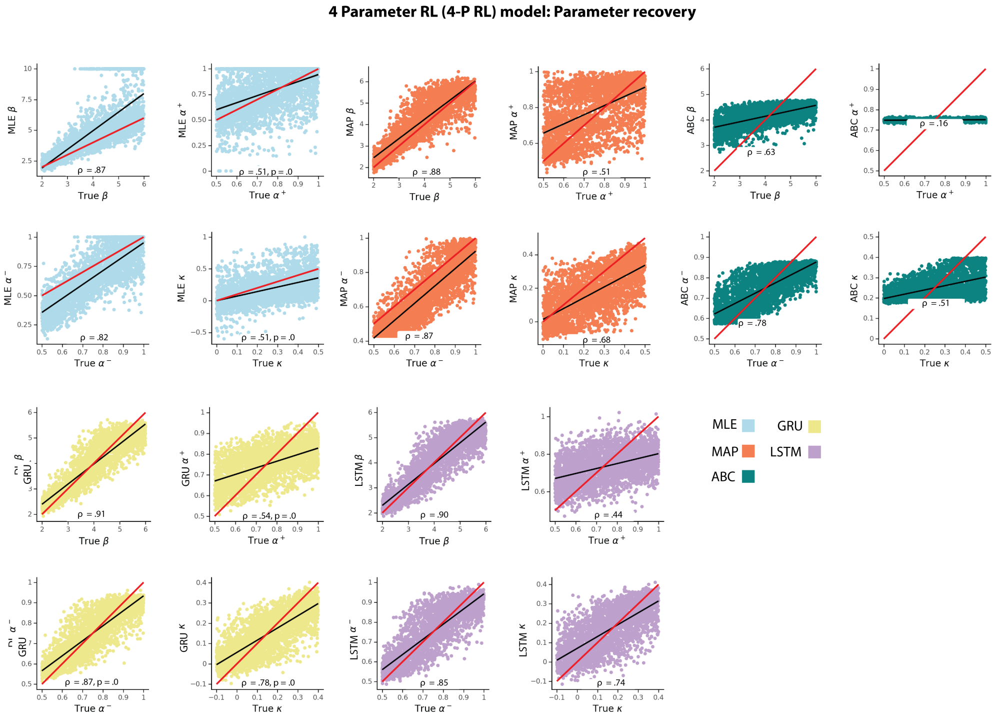


Figure S3: 4 Parameter RL model (4P-RL) parameter recovery using different fitting methods. ρ corresponds to Spearman correlation coefficient, red line represents a unity line (x=x), and black line represents a least squares regression line.

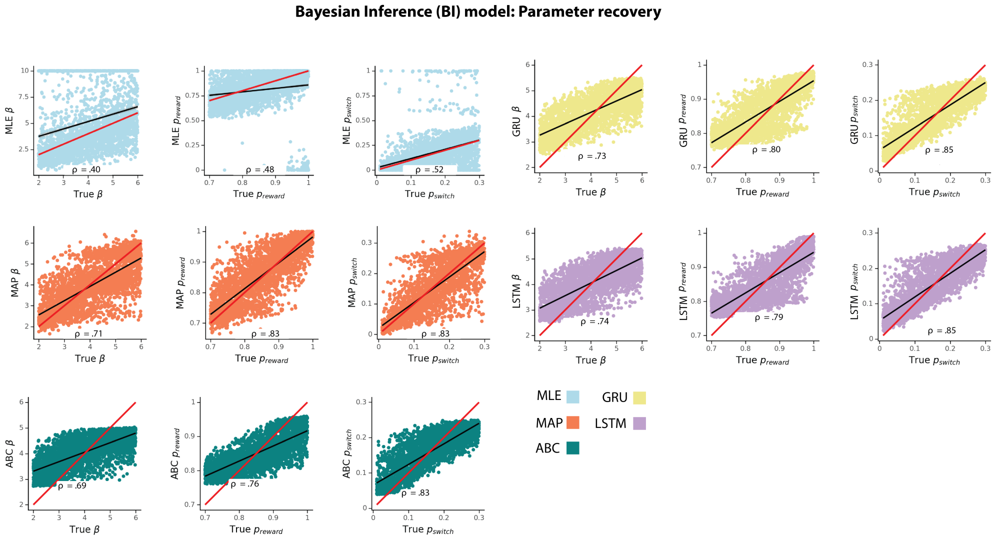
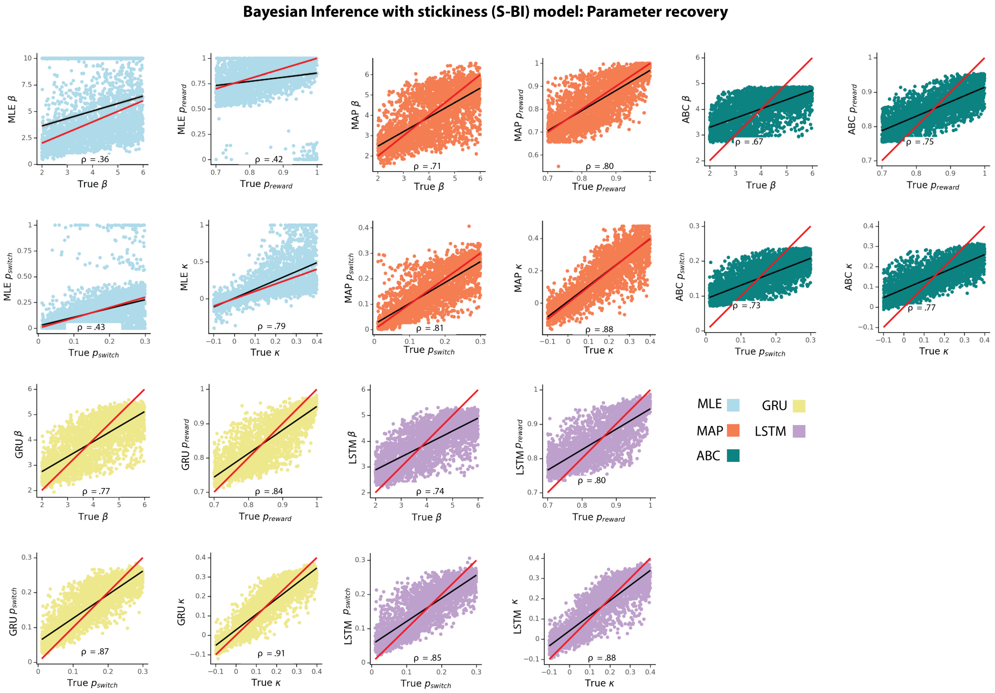


Figure S4: Bayesian Inference (BI) model parameter recovery using different fitting methods. ρ corresponds to Spearman correlation coefficient, red line represents a unity line (x=x), and black line represents a least squares regression line.



Bayesian Inference with stickiness (S-BI) model: Parameter recovery

Figure S5: Bayesian Inference with stickiness (S-BI) model parameter recovery using different fitting methods. ρ corresponds to Spearman correlation coefficient, red line represents a unity line (x=x), and black line represents a least squares regression line.

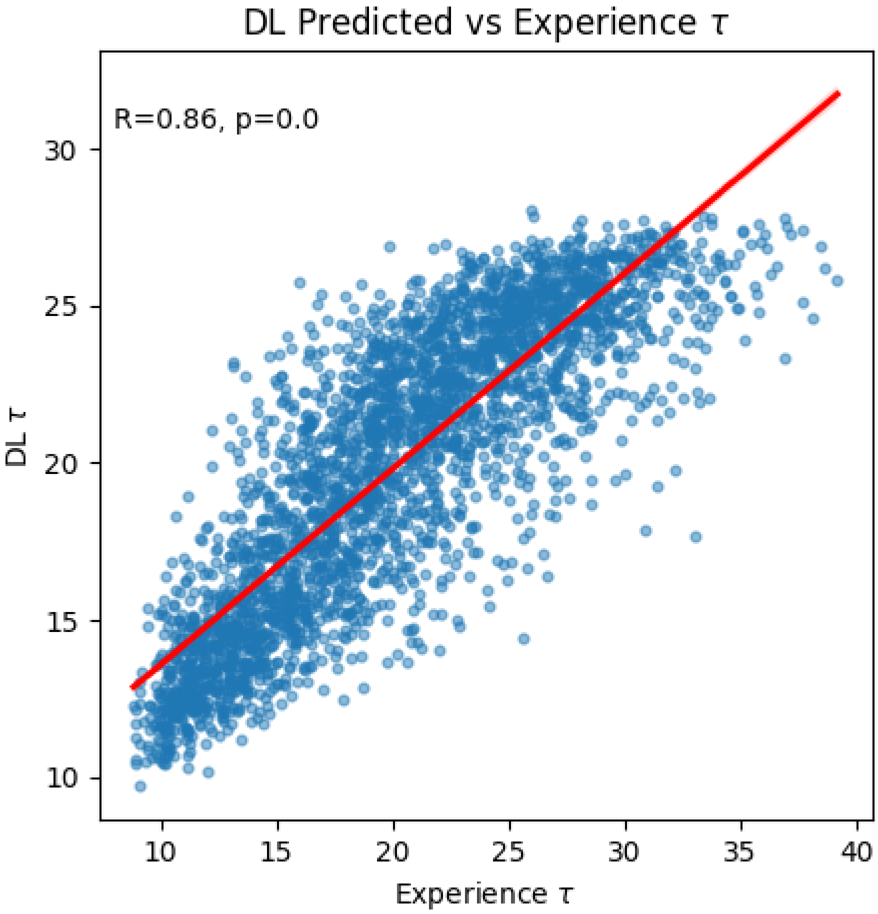


Figure S6: Correlation between the average experienced time intervals in attentive state and the τ parameter in RL-LAS model that captures transition between disengaged/engaged attention states estimated by the ANN.

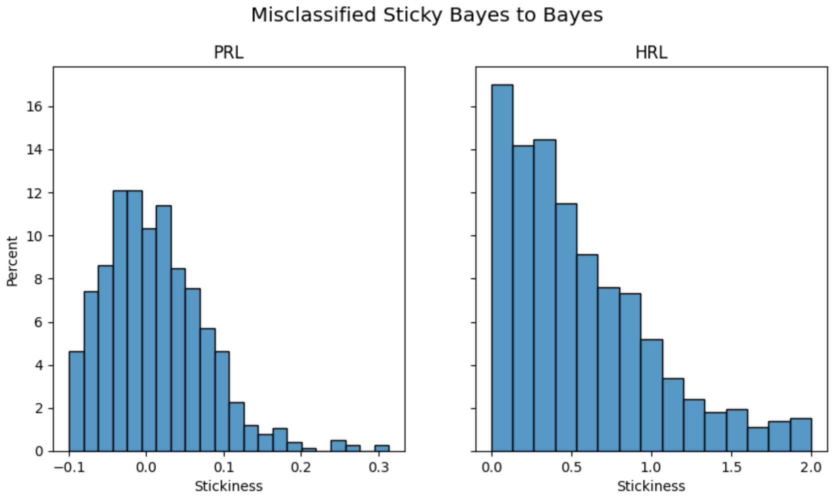


Figure S7: Misclassification of Bayes and sticky Bayes model is contingent on the value of the stickiness parameter κ. The misclassification percentage is higher at κ values closer to 0.

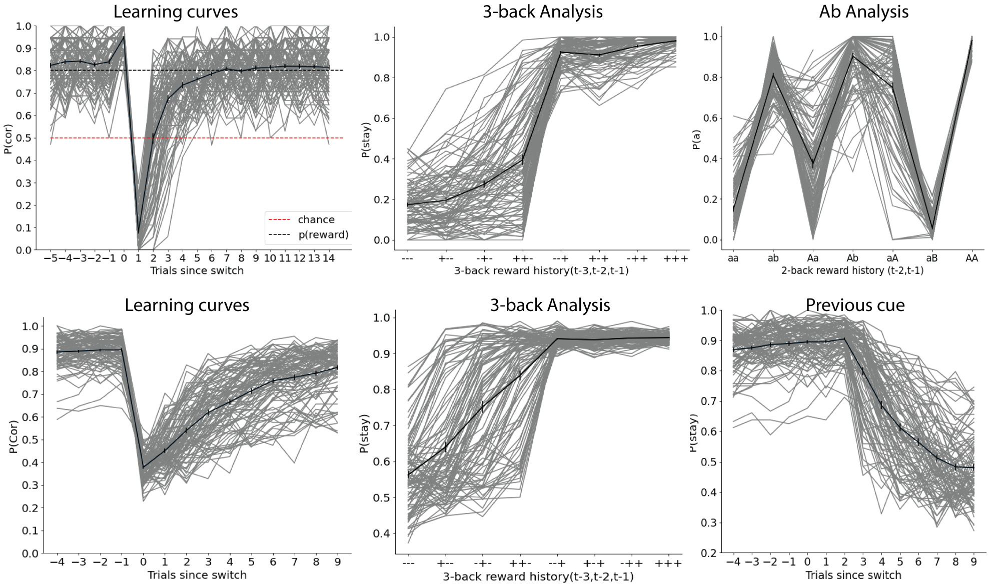


Figure S8: Summary statistics for Approximate Bayesian Computation (ABC). Top row shows summary statistics computed for all models simulated on a probabilistic reversal learning task; the figure only shows agents simulated using a 4-parameter RL model. Bottom row shows summary statistics computed for all models simulated on a hierarchical reversal learning task; the figure only shows performance of HRL model agents. Both rows depict 200 out of 3000 test set agents. Gray lines represent individual agents; black line represents an average across the agents.

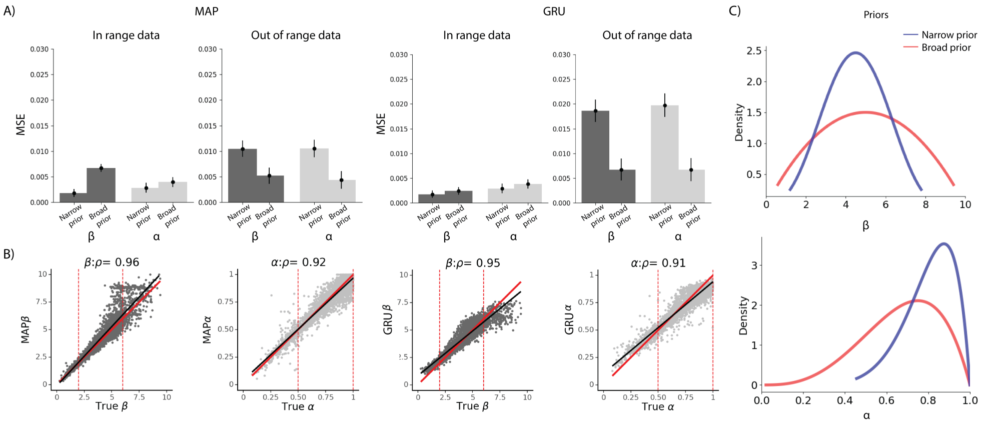


Figure S9: Effect of prior misspecification on parameter estimation in MAP and our ANN approach. A) Applying too narrow prior specification to the fitting procedure (prior in MAP, training samples in ANN) results in difficulty estimating out-of-range parameters for both MAP and ANN. Broader prior specification addresses this issue, with only a slight loss of precision in specific target ranges. Training the network with a broad range of parameters while oversampling parameters from regions of interest yields most robust results. B) Visualization of fitting with MAP and ANN with a wide prior, tested on a full range/wide range data set - training the network with broader range while oversampling from the most plausible range yields less noisy performance in the range compared to MAP. Red lines delineate the range of the narrow prior, which corresponds to the main text results. C) The broad prior was designed by sampling from the full broader range ($\beta \in [0,10]$, $\alpha \in [0,1]$), with the constraint that 70% of samples are in the expected narrow range ($\beta \in [2,6]$, $\alpha \in [0.5,1]$, and 30% outside.)

Tasks	Parameter Recovery				Model Identification		
Cognitive Models	2 PRL	4 PRL	2 PRL intractable	HRL	PRL tractable	2 PRL tractable & intractable	HRL
Batch size	256	256	256	256	128	128	128
GRU units	128	90	256	256	151	151	151
1st dense units	64	45	128	128	75	75	75
2nd dense units	32	22	64	64	37	37	37
Dropout after GRU layer	0.2	0.3	0.2	0.2	0.187	0.187	0.187
Dropout after 1st dense layer	0.1	0.2	0.1	0.1	0.04	0.04	0.04
Dropout after 2nd dense layer	0.01	0.05	0.02	0.01	0.02	0.02	0.02

Table S1: Summary of hyper-parameter values selected from Bayesian optimization algorithms.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). {Tensorflow}: A system for {large-scale} machine learning. *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 265–283.

Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. *Selected papers of hirotugu akaike*, 199–213.

Ashwood, Z. C., Roy, N. A., Stone, I. R., Laboratory, I. B., Urai, A. E., Churchland, A. K., Pouget, A., & Pillow, J. W. (2022). Mice alternate between discrete strategies during perceptual decision-making. *Nature Neuroscience*, *25*(2), 201–212.

Baribault, B., & Collins, A. G. (2023). Troubleshooting bayesian cognitive models. *Psychological Methods*.

Bergstra, J., Yamins, D., & Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. *International conference on machine learning*, 115–123.

Beron, C., Neufeld, S., Linderman, S., & Sabatini, B. (2021). Efficient and stochastic mouse action switching during probabilistic decision making. *Neuroscience*, *10*(2021.05), 13–444094.

Boelts, J., Lueckmann, J.-M., Gao, R., & Macke, J. H. (2022). Flexible and efficient simulation-based inference for models of decision-making. *Elife*, *11*, e77220.

Chen, C., Takahashi, T., Nakagawa, S., Inoue, T., & Kusumi, I. (2015). Reinforcement learning in depression: A review of computational research. *Neuroscience & Biobehavioral Reviews*, *55*, 247–267.

Collins, A. G., Brown, J. K., Gold, J. M., Waltz, J. A., & Frank, M. J. (2014). Working memory contributions to reinforcement learning impairments in schizophrenia. *Journal of Neuroscience*, *34*(41), 13747–13756.

Cools, R., Barker, R. A., Sahakian, B. J., & Robbins, T. W. (2001). Enhanced or impaired cognitive function in parkinson's disease as a function of dopaminergic medication and task demands. *Cerebral cortex*, *11*(12), 1136–1143.

Cools, R., Clark, L., Owen, A. M., & Robbins, T. W. (2002). Defining the neural mechanisms of probabilistic reversal learning using event-related functional magnetic resonance imaging. *Journal of Neuroscience*, *22*(11), 4563–4567.

Costa, V. D., Tran, V. L., Turchi, J., & Averbeck, B. B. (2015). Reversal learning and dopamine: A bayesian perspective. *Journal of Neuroscience*, *35*(6), 2407–2416.

Cousineau, D., & Helie, S. (2013). Improving maximum likelihood estimation using prior probabilities: A tutorial on maximum a posteriori estimation and an examination of the weibull distribution. *Tutorials in Quantitative Methods for Psychology*, *9*(2), 61–71.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dezfouli, A., Ashtiani, H., Ghattas, O., Nock, R., Dayan, P., & Ong, C. S. (2019). Disentangled behavioural representations. *Advances in neural information processing systems*, *32*.

Djuric, P. M., Kotecha, J. H., Zhang, J., Huang, Y., Ghirmai, T., Bugallo, M. F., & Miguez, J. (2003). Particle filtering. *IEEE signal processing magazine*, *20*(5), 19–38.

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.

Eckstein, M. K., Master, S. L., Dahl, R. E., Wilbrecht, L., & Collins, A. G. (2022). Reinforcement learning and bayesian inference provide complementary models for the unique advantage of adolescents in stochastic reversal. *Developmental Cognitive Neuroscience*, *55*, 101106.

Eckstein, M. K., Summerfield, C., Daw, N. D., & Miller, K. J. (2023). Predictive and interpretable: Combining artificial neural networks and classic cognitive models to understand human learning and decision making. *bioRxiv*, 2023–05.

Eppinger, B., Walter, M., Heekeren, H. R., & Li, S.-C. (2013). Of goals and habits: Age-related and individual differences in goal-directed decision-making. *Frontiers in neuroscience*, *7*, 253.

Fengler, A., Govindarajan, L. N., Chen, T., & Frank, M. J. (2021). Likelihood approximation networks (lans) for fast inference of simulation models in cognitive neuroscience. *Elife*, *10*, e65074.

Findling, C., Chopin, N., & Koechlin, E. (2021). Imprecise neural computations as a source of adaptive behaviour in volatile environments. *Nature Human Behaviour*, *5*(1), 99–112.

Frank, M. J., & Badre, D. (2012). Mechanisms of hierarchical reinforcement learning in corticostriatal circuits 1: Computational analysis. *Cerebral cortex*, *22*(3), 509–526.

Ger, Y., Nachmani, E., Wolf, L., & Shahar, N. (2023). Harnessing the flexibility of neural networks to predict dynamic theoretical parameters underlying human choice behavior. *bioRxiv*, 2023–04.

Gillan, C. M., Kosinski, M., Whelan, R., Phelps, E. A., & Daw, N. D. (2016). Characterizing a psychiatric symptom dimension related to deficits in goal-directed control. *elife*, *5*, e11305.

Gläscher, J., Hampton, A. N., & O'Doherty, J. P. (2009). Determining a role for ventromedial prefrontal cortex in encoding action-based value signals during reward-related decision making. *Cerebral cortex*, *19*(2), 483–495.

Hampton, A. N., Bossaerts, P., & O'doherty, J. P. (2006). The role of the ventromedial prefrontal cortex in abstract state-based inference during decision making in humans. *Journal of Neuroscience*, *26*(32), 8360–8367.

Hauser, T. U., Iannaccone, R., Walitza, S., Brandeis, D., & Brem, S. (2015). Cognitive flexibility in adolescence: Neural and behavioral mechanisms of reward prediction error processing in adaptive decision making during development. *NeuroImage*, *104*, 347–354.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

Iwana, B. K., & Uchida, S. (2021). An empirical survey of data augmentation for time series classification with neural networks. *PLOS ONE*, *16*(7), 1–32. https://doi.org/10.1371/journal.pone.0254841

Ji-An, L., Benna, M. K., & Mattar, M. G. (2023). Automatic discovery of cognitive strategies with tiny recurrent neural networks. *bioRxiv*, 2023–04.

Lawrence, A. D., Sahakian, B., Rogers, R., Hodges, J., & Robbins, T. (1999). Discrimination, reversal, and shift learning in huntington's disease: Mechanisms of impaired response selection. *Neuropsychologia*, *37*(12), 1359–1374.

Lee, M. D. (2011). How cognitive modeling can benefit from hierarchical bayesian models. *Journal of Mathematical Psychology*, *55*(1), 1–7.

Lee, M. D., & Webb, M. R. (2005). Modeling individual differences in cognition. *Psychonomic Bulletin & Review*, *12*(4), 605–621.

Lenzi, A., Bessac, J., Rudi, J., & Stein, M. L. (2023). Neural networks for parameter estimation in intractable models. *Computational Statistics & Data Analysis*, *185*, 107762.

Li, J.-J., Shi, C., Li, L., & Collins, A. (2023). Dynamic noise estimation: A generalized method for modeling noise in sequential decision-making behavior. *bioRxiv*, 2023–06.

Li, J.-J., Shi, C., Li, L., & Collins, A. G. (2023). A generalized method for dynamic noise inference in modeling sequential decision-making. *Proceedings of the Annual Meeting of the Cognitive Science Society*, *45*(45).

Liang, S., Li, Y., & Srikant, R. (2017). Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*.

Lintusaari, J., Gutmann, M. U., Dutta, R., Kaski, S., & Corander, J. (2017). Fundamentals and recent developments in approximate bayesian computation. *Systematic biology*, *66*(1), e66–e82.

Minka, T. P. (2013). Expectation propagation for approximate bayesian inference. *arXiv preprint arXiv:1301.2294*.

Montague, P. R., Dolan, R. J., Friston, K. J., & Dayan, P. (2012). Computational psychiatry. *Trends in cognitive sciences*, *16*(1), 72–80.

Moosavi-Dezfooli, S.-M., & Alhussein Fawzi, O. F. (2017). Pascal frossard." *Universal adversarial perturbations." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Myung, I. J. (2003). Tutorial on maximum likelihood estimation. *Journal of mathematical Psychology*, *47*(1), 90–100.

Nguyen, A., Yosinski, J., & Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 427–436.

Niv, Y., Edlund, J. A., Dayan, P., & O'Doherty, J. P. (2012). Neural prediction errors reveal a risk-sensitive reinforcement-learning process in the human brain. *Journal of Neuroscience*, *32*(2), 551–562.

Nussenbaum, K., Velez, J. A., Washington, B. T., Hamling, H. E., & Hartley, C. A. (2022). Flexibility in valenced reinforcement learning computations across development. *Child development*, *93*(5), 1601–1615.

Palestro, J. J., Sederberg, P. B., Osth, A. F., Van Zandt, T., & Turner, B. M. (2018). *Likelihood-free methods for cognitive science*. Springer.

Palminteri, S., Wyart, V., & Koechlin, E. (2017). The importance of falsification in computational cognitive modeling. *Trends in cognitive sciences*, *21*(6), 425–433.

Perfors, A., Tenenbaum, J. B., Griffiths, T. L., & Xu, F. (2011). A tutorial introduction to bayesian models of cognitive development. *Cognition*, *120*(3), 302–321.

Peterson, D. A., Elliott, C., Song, D. D., Makeig, S., Sejnowski, T. J., & Poizner, H. (2009). Probabilistic reversal learning is impaired in parkinson's disease. *Neuroscience*, *163*(4), 1092–1101.

Piray, P., Dezfouli, A., Heskes, T., Frank, M. J., & Daw, N. D. (2019). Hierarchical bayesian inference for concurrent model fitting and comparison for group studies. *PLoS computational biology*, *15*(6), e1007043.

Rigoux, L., Stephan, K. E., Friston, K. J., & Daunizeau, J. (2014). Bayesian model selection for group studies—revisited. *Neuroimage*, *84*, 971–985.

Rmus, M., He, M., Baribault, B., Walsh, E. G., Festa, E. K., Collins, A. G., & Nassar, M. R. (2023). Age-related differences in prefrontal glutamate are associated with increased working memory decay that gives the appearance of learning deficits. *Elife*, *12*, e85243.

Särkkä, S., & Svensson, L. (2023). *Bayesian filtering and smoothing* (Vol. 17). Cambridge university press.

Schwarz, G. (1978). Estimating the dimension of a model. *The annals of statistics*, 461–464.

Sheynin, J., Moustafa, A. A., Beck, K. D., Servatius, R. J., & Myers, C. E. (2015). Testing the role of reward and punishment sensitivity in avoidance behavior: A computational modeling approach. *Behavioural Brain Research*, *283*, 121–138.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1–48.

Shultz, T. R. (2003). *Computational developmental psychology*. Mit Press.

Sisson, S. A., Fan, Y., & Beaumont, M. (2018). *Handbook of approximate bayesian computation*. CRC Press.

Solway, A., Diuk, C., Córdova, N., Yee, D., Barto, A. G., Niv, Y., & Botvinick, M. M. (2014). Optimal behavioral hierarchy. *PLoS computational biology*, *10*(8), e1003779.

Sunnåker, M., Busetto, A. G., Numminen, E., Corander, J., Foll, M., & Dessimoz, C. (2013). Approximate bayesian computation. *PLoS computational biology*, *9*(1), e1002803.

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

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Thompson, B., Van Opheusden, B., Sumers, T., & Griffiths, T. (2022). Complex cognitive algorithms preserved by selective social learning in experimental populations. *Science*, *376*(6588), 95–98.

Turner, B. M., Forstmann, B. U., Wagenmakers, E.-J., Brown, S. D., Sederberg, P. B., & Steyvers, M. (2013). A bayesian framework for simultaneously modeling neural and behavioral data. *NeuroImage*, *72*, 193–206.

Turner, B. M., & Sederberg, P. B. (2014). A generalized, likelihood-free method for posterior estimation. *Psychonomic bulletin & review*, *21*, 227–250.

van Opheusden, B., Acerbi, L., & Ma, W. J. (2020). Unbiased and efficient log-likelihood estimation with inverse binomial sampling. *PLoS computational biology*, *16*(12), e1008483.

Weber, I., Zorowitz, S., Niv, Y., & Bennett, D. (2022). The effects of induced positive and negative affect on pavlovian-instrumental interactions. *Cognition and Emotion*, *36*(7), 1343–1360.

Wei, Y., & Jiang, Z. (2022). Estimating parameters of structural models using neural networks. *USC Marshall School of Business Research Paper*.

Zou, A. R., Muñoz Lopez, D. E., Johnson, S. L., & Collins, A. G. (2022). Impulsivity relates to multi-trial choice strategy in probabilistic reversal learning. *Frontiers in Psychiatry*, *13*, 800290.