#### Computational modelling

We used the original hybrid model by Daw et al. (2011), which assumes that there is an explicit trade-off between the model-based and model-free system. Model-based and model-free algorithms learn the value of the stimuli that appear in the task in three different pairs. There is one first-stage pair (*s*1 ∈ {1*,*2}), where one stimulus represents the colour red and the other stimulus represents the colour blue. These colours were paired with 5 irrelevant shapes, making 10 unique combinations of first stage stimulus pairs. There are two second-stage pairs consisting of two stimuli each (*s*2 ∈ {3*,*4*,*5*,*6}). The indices *s*1 and *s*2 refer to stage 1 and stage 2, respectively. The index *t* refers to the trial number.

At the first-stage, model-free ‘cached’ values were updated using the SARSA (*λ*) temporal difference algorithm. This algorithm learns to maximise the total outcome by strengthening or weakening associations between the first-stage state and the first-stage actions, depending on whether a reward followed that action or not:

*,* (1)

where *α*is the learning rate. The parameter *λ* is a gain parameter for the eligibility traces, which connects the two stages and allows the second-stage reward prediction error to influence first-stage choices. If *λ* = 1, the reward on stage 2 drives the choice on stage 1, and if *λ* = 0, outcomes on stage 2 do not influence stage 1 choices, which is representative of a pure model-based approach. The parameter *λ* also accounts for the main effect of reward as observed in the analysis of first-stage stay-switch behaviour, but not for an interaction of reward and state.

Model-based values were calculated for each first-stage stimulus and every trial in a forward looking manner by multiplying the state value of the best second-stage option with the state transition probabilities:

(2) . (3)

Model-based learning was simplified and the transition probabilities were not updated by explicitly modelling state prediction errors. Simulations by the authors of the original task showed that learning of state transitions quickly converge to stable values (see supplementary materials Daw et al. (2011)).

The hybrid model computes the actual value that is used in determining the stage 1 choice as a weighted combination of the model-based (*Q*MB) and model-free (*Q*MF ) values. The first-stage Q-values were computed in the following way:

(4)

where *ω* is a weighting parameter for the model-based strategy and (1 − *ω*) is the weighting factor for the model-free strategy.

Q-values for the four second-stage stimuli were updated according to the reward prediction errors (Rummery and Niranjan, 1994):

(5)

where *α*is the learning rate. We assume that the learning rate for both stages is the same.

A first-stage choice depends on the relative difference in stimulus values between *Q*1 and *Q*2 and the choice *C* on the previous trial, which takes on the value 1 when the current choice equals the previous choice. The parameter *π* captures first-stage choice perseverance. Using the softmax choice function, the probability of choosing a first-stage stimulus was computed according to:

, (6)

and for the second-stage:

, (7)

where *β*is the softmax temperature that controls the stochasticity of choices, which we assume to be the same for both stages.

**Model fitting procedure**

We used a hierarchical model-fitting strategy that takes into account the likelihood of individual participant choices given the individual participant parameters and also the likelihood of the individual participant parameters given the parameter distribution in the overall population across conditions. This method regularises individual participants’ parameter fits, rendering them more robust toward over-fitting. The parameters were estimated for the explicit learning group and implicit learning group separately.

We transformed [0,1]-bounded parameters (λ,α, ω) into a Gaussian scale using the logistic function:

and the [0, ∞)-bounded parameters, (β,π), were logarithmically scaled using the exponential function:

.

The model parameters are denoted by Greek letters and their respective Gaussian transformations by Latin letters. Normally distributed parameters allow for the use of parametric tests to identify differences between sessions.

Gradient-based optimisation algorithms are not guaranteed to converge to a global maximum and could instead converge to local maxima. To avoid this, the algorithm was initialised with a range of starting parameters and the iteration with the highest likelihood value was chosen to make further inference.

All model fitting procedures were verified on surrogate data generated from a known decision process.

**Latent Variables**

Using the best fitting parameter estimates, the latent variables were obtained from a known decision process for each participant.

To understand which of the computational processes contributed most to the rating values, we computed various prediction errors. We differentiated retrospective from prospective processes. Given that the reward probabilities change every 30 trials, there is always a second-state stimulus that is the best option.

RPE1 = r - stage 1 Qmb

RPE2 = r – stage 1 Qmf

RPE6 = Qmb(t+1) – Qmb(t)

RPE7 = r – Qmb best option to win

RPE8 = r – Qmf best option to win

RPE9 = r – Qmb(stage 1 choice with highest transition probability to current state)

RPE10 = r – Qmf(stage 1 choice with highest transition probability to current state)