Using Recurrent Neural Networks to Understand Human Reward Learning

Mingyu Song (mingyus@princeton.edu), Yael Niv (yael@princeton.edu)
Princeton Neuroscience Institute, Princeton University
Princeton, NJ 08540, United States

Ming Bo Cai (mingbo.cai@ircn.jp)
International Research Center for Neurointelligence, (WPI-IRCN), UTIAS, The University of Tokyo
Bunkyo City, Tokyo 113-0033, Japan

Abstract

Computational models are greatly useful in cognitive science in revealing the mechanisms of learning and decision making. However, it is hard to know whether all meaningful variance in behavior has been account for by the best-fit model selected through model comparison. In this work, we propose to use recurrent neural networks (RNNs) to assess the limits of predictability afforded by a model of behavior, and reveal what (if anything) is missing in the cognitive models. We apply this approach in a complex reward-learning task with a large choice space and rich individual variability. The RNN models outperform the best known cognitive model through the entire learning phase. By analyzing and comparing model predictions, we show that the RNN models are more accurate at capturing the temporal dependency between subsequent choices, and better at identifying the subspace in the space of choices where participants' behavior is more likely to reside. The RNNs can also capture individual differences across participants by utilizing an embedding. The usefulness of this approach suggests promising applications of using RNNs to predict human behavior in complex cognitive tasks, in order to reveal cognitive mechanisms and their variability.

Keywords: recurrent neural network; model comparison; probabilistic reward learning; sequential decision making

Introduction

Computational models of human cognition have greatly helped us in understanding the way people learn and make decisions. For example, reinforcement learning (RL) models reveal how people learn to acquire reward from trial-and-error experience; Bayesian inference models demonstrate how people combine prior knowledge and observations to form beliefs about the world. However, despite the great power of these models and what they have taught us about human mind, much variance in the data is often left unexplained even with the best-predicting cognitive models at hand. For example, in a multi-dimensional probabilistic learning task (Song et al., 2020), an average likelihood of $p = 0.22$ per trial was achieved using the best cognitive model. It is unclear whether this seemingly low model likelihood is purely due to the stochastic nature of human behavior, or whether it indicates room for improvement and potentially a better model. This is particularly relevant for complex sequential learning tasks, where decisions depend not only on the current stimulus, but also the history of experience; and is further complicated when there is a large number of candidate choices (e.g. 64 different choices in the above study), making it hard to precisely predict participants' behavior.

Understandably, most cognitive modeling work focus on relative model comparison, which identifies the best-fit model out of a number of alternatives under consideration. Studies do not, however, commonly evaluate the absolute goodness of fit of models, or estimate how far they are from the best possible model. In theory, there exists an upper bound for model log likelihood (i.e., the negative entropy of the data (Cover, 1999)). However, estimating this bound is practically impossible in sequential tasks, because the exact same experience is never repeated (it may be possible, however, in simpler perceptual decision-making tasks where the same choice can be tested repeatedly (Shen & Ma, 2016)). In this work, we propose an alternative empirical approach: to use recurrent neural networks (RNNs) to predict human choices.

RNNs are neural networks with recurrent connections that can pass information from one time point to the next. They are widely used to model temporal dynamics, making them particularly suitable for capturing the sequential dependence of human behavior. Compared to cognitive models that have carefully-crafted assumptions based on knowledge about human cognition, RNNs are agnostic to cognitive processes. They usually have thousands of free parameters (as compared to cognitive models with around 10). The flexibility means that RNNs can potentially capture more variance than do cognitive models, given a sufficient amount of training data (and at the expense of a clearly understood process model of how the participant made their decision). Despite the fact that RNNs have been widely used to solve cognitive tasks (Yang et al., 2019; Wang et al., 2018; Lu et al., 2020), only a handful of works have used RNNs to directly model the way *people* solve these tasks by predicting their behavior on a trial-by-trial basis (Dezfouli, Griffiths, et al., 2019; Dezfouli, Ashtiani, et al., 2019; Fintz et al., 2021). In these works, RNNs have been applied to bandit problems with very sparse reward (Dezfouli, Griffiths, et al., 2019) or no clearly better options (Fintz et al., 2021), and were able to predict either counter-intuitive win-shift-lose-stay behavior, or stereotyped alternation of options, both of which common cognitive models (e.g., reinforcement learning models) failed on. In fact, due to the uninformativeness of reward signals in those specific tasks, heuristics or stereotyped behavior were more likely, which may explain the success of RNNs as compared to reinforcement learning models.

In the current work, we sought to apply this approach to

more standard reward-learning scenarios, in a task with a large enough choice space that will ensure sufficient variance to be explained. We thus considered the probabilistic multi-dimensional reward learning task mentioned in the beginning (Song et al., 2020). In this task, participants tried to learn about multi-dimensional rules through actively configuring three-dimensional stimuli (color, shape and pattern) and receiving probabilistic feedback for their configuration. Prior work found that participants' learning strategy consisted of a combination of reinforcement learning and hypothesis testing with rich individual differences. The best available cognitive model, the "value-based serial hypothesis testing model", out-performed many commonly-used cognitive models, and was able to account for individual differences in belief space and choice policy. However, it still deviated from the data in some important aspects, as we detail below.

We applied RNNs to fit participants' behavior in this task, and found that RNNs predicted choices better than the best cognitive model (average likelihood increased by about 0.04 per trial, from the original $p = 0.22$), suggesting room for improvement for cognitive models. We investigated the underlying reasons for RNNs' superior performance, and found at least two: RNNs were more accurate at capturing the dependency between consecutive choices, and RNNs worked well with the large choice space by identifying the subspace which participants used more accurately than the best cognitive model. We further considered the rich individual difference observed in this task, and incorporated it into the RNN by adding an embedding of individual participants. Participant embedding helped model fits, especially for the first few trials of each independent "game". The embedding space encoded meaningful cognitive variables whose variance across participants was not captured by an RNN without embedding.

Reward learning task and cognitive models

In this section, we briefly introduce the reward learning task and how participants performed in this task. We summarize participants' performance and choice behavior, as well as the results of previous computational cognitive models. For details on the task and modeling, please see (Song et al., 2020).

In the "build-your-own-stimulus" reward-learning task (Figure 1), participants were asked to configure stimuli by choosing their color, shape and/or pattern (each from three options), and received probabilistic binary reward feedback. On each trial, they would select what feature to use for each of the three dimensions, or leave any dimension blank to let the computer randomly decide for that dimension, resulting in 64 possible choices in total (a "choice" is defined as a combination of decisions on all dimensions). In each game, the probability of reward for each possible stimulus was determined by an underlying rule. The rule specified what dimension(s) were reward-relevant in that game (ranging from one dimension to all three dimensions, corresponding to 1D, 2D and 3D conditions); within each relevant dimension, one of the features was designated as the "rewarding feature", and was

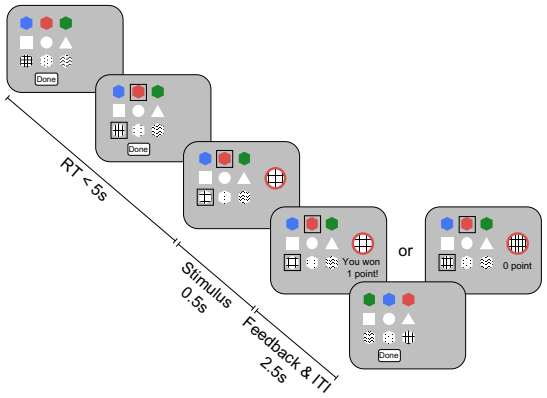

Figure 1: **The build-your-own-stimulus task.** On each trial, participants built a stimulus by selecting a feature (marked by black squares) in each of 0-3 dimensions. After hitting "Done", the stimulus showed up on the screen, with features randomly determined for any dimension without a selection (here, circle was randomly determined). Reward feedback (1 or 0 points) was then shown.

more likely to generate reward than the other two features. Participants' goal was to figure out the underlying rule and build stimuli with rewarding features to earn as many reward points as possible. In half of the games ("known" games), participants were instructed on the number of reward-relevant dimensions; in the other half ("unknown" games), they were not. This resulted in six game types in total.

102 participants performed this task on Amazon Mechanical Turk. Each participant completed 18 games (3 games for each type), with 30 trials in each game.

In all six game types, participants' performance improved over the course of a game (Figure 2A). Games were harder (i.e., participants were less able to learn all the rewarding features) as game complexity (the number of relevant dimensions) increased. Participants showed distinct choice behavior in different game types (Figure 2D): in "known" games, they systematically selected more features on each trial in more complex games; in "unknown" games, the number of selected features was not different across game complexities.

Several cognitive models were tested, including a Bayesian rule learning model, a feature-based reinforcement learning model, and serial hypothesis testing (SHT) models. Through extensive model comparison, the cognitive model found to best predict participants' choices was the value-based SHT model, which combines both value-based reinforcement learning and rule-based hypothesis testing strategies. This model also included parameters specifying each participant's hypothesis space and choice policy, allowing it to capture individual differences. Model performance (simulated with the best-fit parameters for each participant) is shown in Figure 2B and 2E. The model achieved an geometric average trial-wise likelihood of $p = 0.22$, compared to chance level of $p = 0.0156$ (random selection among 64 possible choices). It was also able to capture how performance depended on task difficulty, and correctly predicted the qualitative difference

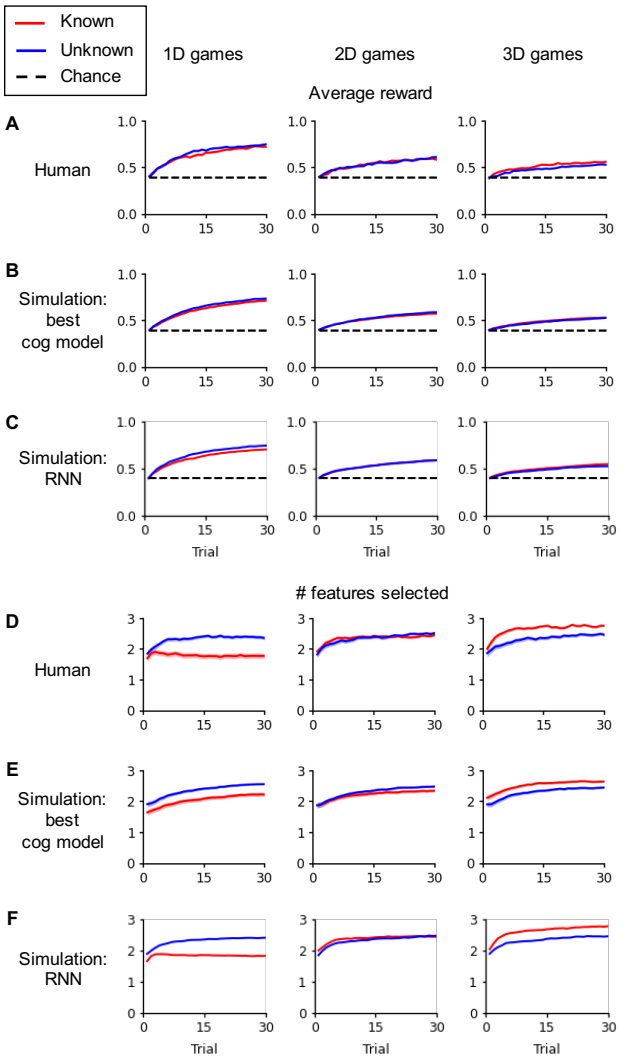

Figure 2: **Performance and choice behavior by game type.** **(A-C)** The average reward, and **(D-F)** the number of features selected per trial over the course of 1D, 2D and 3D games (left, middle and right columns); red and blue curves represent the "known" and "unknown" conditions, respectively. Shaded areas: 1 s.e.m. across participants. Dashed lines: chance level for that type of game. (A,D): participants' behavior; (B,E): simulation of the best cognitive model; (C,F): simulation of the RNN model.

between known and unknown games on the number of features selected. However, the model deviated from human data quantitatively. Cognitive modeling therefore provided important insights for understanding human strategy in this task: it supported the existence of two learning systems (value-based and rule-based), and shed lights on how the integration of the two systems depended on task condition (for more details, see Song et al. (2020)). Unsurprisingly, however, even the best cognition model could not fully account for all the variance in behavior.

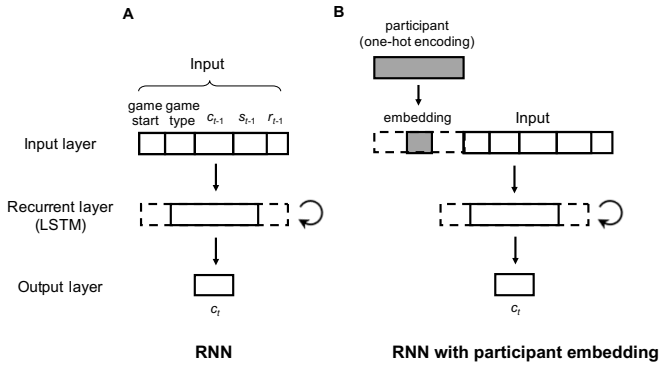

Figure 3: **RNN model structure.** **(A)** The RNN model. **(B)** The RNN with participant embedding model. Circled arrows indicate recurrent connections. Dashed squares indicate varying layer sizes (set by hyper-parameters).

Apply RNNs to fit behavior

In this work, we used recurrent neural networks (RNNs) to fit behavior in this task. The RNN model used here consists of an input layer, a recurrent layer, and an output layer (Figure 3A). On each trial, the input variables consist of a game-start indicator (1 if it is the first trial, 0 otherwise), the game type, the participant's choice (c_{t-1}), the configured stimulus (s_{t-1}), and the outcome (r_{t-1}), with the last three variables taken from the previous trial and are zero on the first trial of a game. Each input variable is one-hot encoded, forming a concatenated binary input vector. The recurrent layer is a long short-term memory (LSTM) layer, followed by a linear feed-forward connection to the output layer, which is then transformed by a Softmax function (with inverse temperature fixed at 1) to determine the probability for choices on the current trial (c_t). We used the cross-entropy loss, i.e., log likelihood of participant's choice, as the cost function. We optimized the network using the Adam optimizer in PyTorch. Hyper-parameters of the models included learning rate, batch size, and the size of the recurrent layer.

We split data into training, validation and test sets. The training set consisted of 16 games from each participant (augmented 1024 times through shuffling the dimensions and features; see Discussion for details), and the validation and test sets each consisted of 1 game per participant. The game type and game index were balanced (to the extent possible) in each set to reduce potential bias or order effect. The weights of the networks were trained on the training set, the hyper-parameter values were selected based on the validation set (the selected values were as follows: learning rate of 0.001, batch size of 10000, recurrent layer size of 50, and early stopping at epoch 28), and all results reported were evaluated on the test set using the best fit network.

Compare RNN with the best cognitive model

The RNN performed better than the best cognitive model, and the advantage persisted through the course of a game (Figure 4A). Because the same network was used to predict all

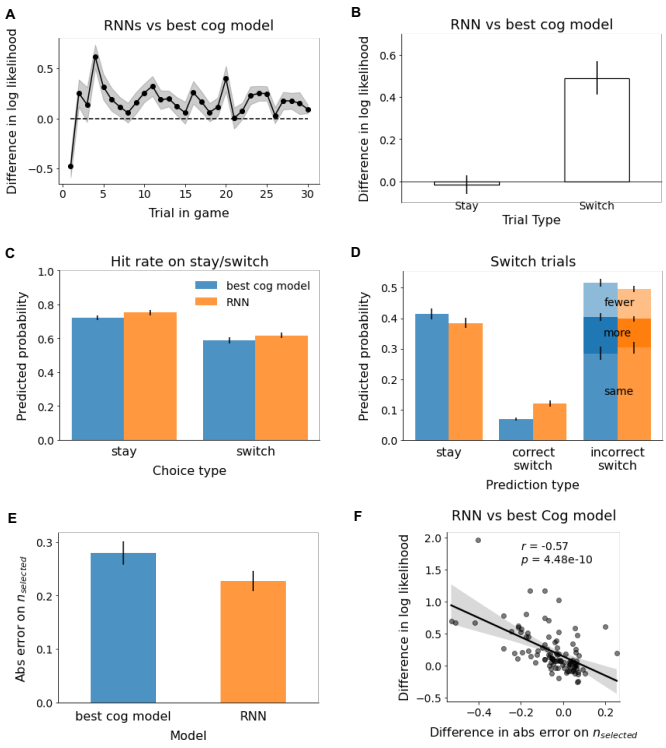

Figure 4: **Comparison between the RNN and the best cognitive model reveals the advantage of RNN in predicting switch trials and the number of features selected per trial.** **(A)** Log likelihood difference between the RNN and the best cognitive model, over the course of a game. Positive values indicate better prediction by RNN. **(B)** Log likelihood differences between the models for stay and switch trials respectively. **(C)** Probability that the models assigned for staying or switching on "stay" or "switch" trials, respectively. For "switch" trials, the probability was calculated as the sum of all possible switch choices. Blue bars: best cognitive model; orange bars: RNN; same in (D,E). **(D)** Model prediction on switch trials only, showing separately the probability that the model predicted for stay, correct switch, and incorrect switch choices (further divided into "fewer", "more" and "same", depending on the number of features selected in each choice, n_{selected}, relative to the true choice). **(E)** Mean absolute error in the number of features selected by the two models, calculated by taking the expectation over all possible choices with respect to the predicted probability of each choice. **(F)** Difference in log likelihood between the RNN and best cognitive model as a function of their difference in absolute error on n_{selected}, for each participant. The better the prediction on the number of features selected, the better the RNN did in fitting the participants' overall behavior in the test game. Shaded area (A) and error bars (B-E): 1 s.e.m. across participants.

participants, the RNN described above could not be personalized for individual participants; this is in contrast to cognitive models that fit a separate set of parameters for each participant. This can explain the worse fit of the RNN compared to the best cognitive model on the first trial of each game.

In the following, we try to identify the reasons for the RNN's good performance, and how they might help us understand what is lacking in the cognitive models. First, we divided all trials into two types: trials where participants repeated the previous trial's choice ("stay trials") and trials where they did not ("switch trials"); this analysis ignored the first trial of each game, which does not fall into either category. The advantage of the RNN over the cognitive model is primarily due to the switch trials (Figure 4B). This result was not due to the RNN being biased towards predicting switches. In fact, when examining how accurate the models are at predicting whether a trial would be a stay or switch trial, the RNN assigned a higher probability for both choice types than did the cognitive model (Figure 4C)[1], suggesting that it was better at correctly identifying how the next trial depended on the previous one (stay or switch).

We focused on switch trials to further investigate the RNN's better performance (Figure 4D). We categorized all possible choices into mistakenly predicting stay, predicting the correct switch, or predicting a switch to an incorrect choice. In the latter case, we further categorized the choices based on whether they involved selecting fewer, the same number, or more features than did the participant. Consistent with the results above, the RNN made fewer mistakes on switch versus stay (lower predicted probability for stay, and higher for correct switch). When it did make a mistake, the RNN was more likely to at least correctly predict the number of features selected. This was confirmed by an overall lower absolute error on predicting the number of features selected (Figure 4E). Finally, across participants, a lower absolute error in predicting the number of features selected by the RNN as compared to the cognitive model was correlated with a greater log likelihood advantage (i.e., a better fit) for the RNN model (Figure 4F).

Taken together, these analyses showed that the RNN was better at correctly predicting the number of features selected. This is particularly useful when the choice space is very large (as in this task), as it helps the RNN identify the correct subspace of the true choice. This is, however, not the complete story: the RNN was more likely to predict the true choice even within the correct subspace (predicted probability ratio between the true choice and all switch choices with the correct n_{selected} is 0.304 for RNN and 0.241 for the cognitive model), the reason behind which remains to be investigated.

Embedding captures individual differences

So far, we used the same RNN, referred to as **the RNN model**, to predict data from all participants. This works well if all participants use the same strategy. By fitting cogni-

[1]Note that (1) predicting "switch" means correctly identifying the "switch" choice type, but not necessarily correctly predicting what the participant switched to; (2) stay trials were relatively easy to predict for both models, thus their likelihood was much higher than switch trials; additional improvement when the likelihood is high contributes less to the total *log* likelihood (cross-entropy loss), explaining the small difference between the models in Figure 4B.

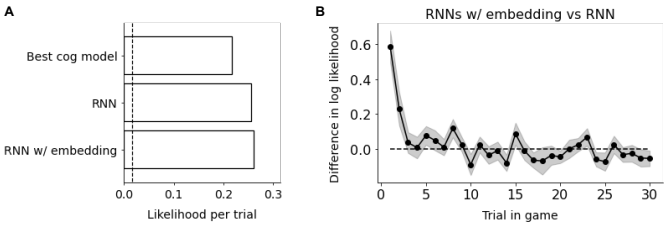

Figure 5: **RNN model comparison shows the effect of embedding**. **(A)** Model likelihood comparison between the best cognitive model, RNN and RNN with participant embedding. **(B)** Log likelihood difference between RNN with and without embedding, over the course of a game. Shaded area: 1 s.e.m. across participants.

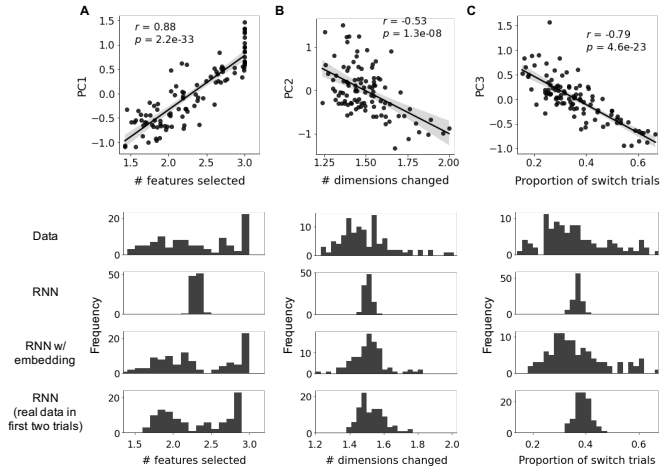

Figure 6: **Participant embedding encodes individual differences.** Top row: the first three principle components of the embedding layer are correlated with **(A)** average number of features selected per trial; **(B)** average number of dimensions changed on switch trials; **(C)** the proportion of switch trials. Bottom rows: histograms of the corresponding variables in participant data and model simulations of the RNN model, the RNN with participant embedding model, and the RNN model that used participants' data in first two trials of the game.

tive models to individual participants, however, we obtained different parameter estimates, suggesting that there might be strategy differences across individuals. Thus, we considered **the RNN with participant embedding model** (Figure 3B), by adding an embedding of individual participants to the input layer of the original RNN. The embedding was trained end-to-end together with the rest of the network. The size of the embedding layer was an additional hyper-parameter. As with the other hyper-parameters, its value was selected based on model performance on a validation set. The best-fit RNN with participant embedding model used the following values: learning rate of 0.001, batch size of 10000, recurrent layer size of 50, participant embedding size of 3, early stopping at epoch 23.

Adding the participant embedding improved the performance of the network (Figure 5A), most notably in the first two trials of each game (Figure 5B).

Despite the limited improvement in prediction, the embedding was crucial for reproducing individual differences. To investigate what information is encoded in the embedding, we performed a PCA analysis on the embedding activity. The three principal components (PCs) were correlated with the following cognitive variables of individual participants, respectively (Figure 6 top row): (1) PC1: average number of features selected per trial; (2) PC2: average number of dimensions changed on switch trials; (3) PC3: the overall proportion of switch trials.

We tested how well the network models captured individual differences by comparing the histograms of these variables in data with those obtained from model simulations. The RNN model failed to capture individual differences; in fact, it could only predict the mean of these variables. In contrast, the RNN with participant embedding model, was able to capture the distribution of all three variables, demonstrating the usefulness of the embedding. Inspired by the finding that adding embedding improved model fits primarily in the first two trials (Figure 5B), we further tested the RNN model by providing it with participants' data (choices and outcomes) for the first two trials in the simulation. With such information, the RNN model *without* embedding was able to capture the variance of some variables (e.g., number of features selected), but still failed on others (e.g., proportion of switch trials). This suggests that the embedding layer encodes information beyond what can be extracted from first two trials of data.

Taken together, these results show the usefulness of an embedding layer in RNNs in capturing the characteristics of individual participants. The embedding layer can then be used to measure similarity between people and serve as the basis of finding sub-groups in a population. In fact, similar work has been done in (Dezfouli, Ashtiani, et al., 2019), where an RNN with embedding model was fit to two-armed bandit task data. One of the two dimensions of the embedding space was found to differentiate between healthy, depressed and bipolar populations, showing the usefulness of embedding in diagnosing patients with cognitive tasks.

Moreover, it is worth noting that the embedding activity was found to encode cognitively-meaningful information in the current task. This is promising if generalizable to other tasks. Neural networks are notoriously hard to interpret despite being powerful prediction tools. The participant embedding, however, helps to make RNNs more interpretable, and can be useful in identifying important task variables that determine participants' strategies. Related, if the embedding vector is found to be correlated with certain cognitive functions (e.g., working memory capacity), the measurement of such cognitive functions can then be used to predict a participant's behavior on the task, and vice versa.

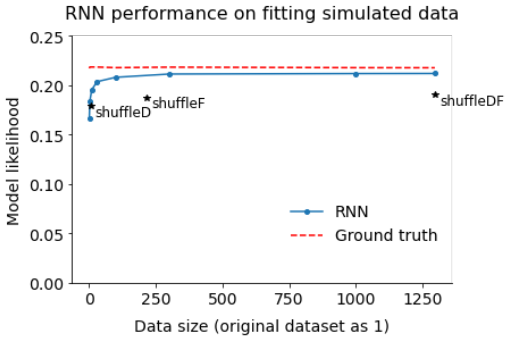

Figure 7: **Dependency of RNN performance on training data size, and effect of data augmentation.** We simulated "data" of varying sizes (1 to 1296 times of the original dataset) with the best cognitive model using best-fit parameters, and used the RNN model to fit these datasets. Red dashed line: ground-truth likelihood of the generative model; blue dots: likelihood of the RNN model. RNN performance improved with data size, asymptotically approaching the ground truth. We augmented the simulated data of size 1 by 6, 216, and 1296 times through shuffling dimensions (shuffleD), features (shuffleF) or both (shuffleDF; as we did for the true data), and fit the RNN model to the augmented datasets; results shown in black asterisk. Data augmentation helped increase the effective data size, but only by roughly 10 times.

Discussion

We showed that RNN models fit human behavior better than cognitive models in a complex reward-learning task. The best cognitive model previously developed for this task used a hybrid strategy that combines reinforcement learning and serial hypothesis-testing, and considered individual differences in how participants understand instructions, their hypothesis-testing policy, and choice policy. However, such a sophisticated cognitive model still failed to fully explain participants' choices. RNN models, in contrast, made no assumption about cognitive processes, but were able to generate more precise predictions both at the group level and for individual participants. Analyses of model predictions revealed that the advantage of RNNs persisted throughout the course of learning. In particular, RNNs were better at predicting "stay" versus "switch" trials, i.e., how the current choice related to the previous one, as well as predicting what subset of the large choice space was used by participants (i.e., the number of features selected). As a next step, we can utilize the insights gained from RNNs to improve cognitive models, in order to achieve a better account of human behavior in this task.

We hope to demonstrate with this work the general value in training RNNs to predict human behavior in complex cognitive tasks. This approach has so far been under-utilized, with only a handful of endeavors (Dezfouli, Ashtiani, et al., 2019; Dezfouli, Griffiths, et al., 2019; Fintz et al., 2021). This is potentially due to the large amount of data required for training neural network models. In the current work, we were able to augment the training set by utilizing the symmetric task structure. Assuming that participants' strategies did not depend on specific dimensions or features, we generated auxiliary data by shuffling the dimensions and features in each game (for both choices and stimuli), which effectively increased the training data size by about 10 times (although still less than the ideal size, and we ran the risk of under-fitting; Figure 7). Such data augmentation may not be possible for every task. A more general solution for using limited amount of data more efficiently may be to use cognitive models as priors for neural network models (Bourgin et al., 2019). As big data becomes more and more common in cognitive science (Suchow et al., 2020), this approach would become more powerful.

The RNN approach has been successfully applied in scenarios where the available cognitive models (most often RL models) are not good candidates for explaining observed behavior that is largely heuristic or stereotyped (Dezfouli, Ashtiani, et al., 2019; Fintz et al., 2021). The current work, in contrast, showcases another use case for this approach: complex cognitive tasks with rich individual variability. In tasks such as the current "build-your-own-stimuli" task, where the best available cognitive model cannot fully capture the richness in behavior, RNNs can be useful in (1) finding the empirical upper bound for goodness of fit; (2) revealing what is missing in the cognitive models; (3) capturing the richness of individual behavioral differences. In this work, we achieved these goals by conducting model comparison, analyzing the winning RNN models, and developing RNN models with participant embedding. Future work can seek to apply RNNs in other similarly complex cognitive tasks to improve our understanding of human cognition and its variability.

Acknowledgment

This work was supported by the National Institute of Drug Abuse (R01DA042065) and Army Research Office (W911NF-14-1-0101). MBC was supported by World Premier International Research Center Initiative, MEXT, Japan.

References

Bourgin, D. D., Peterson, J. C., Reichman, D., Russell, S. J., & Griffiths, T. L. (2019). Cognitive model priors for predicting human decisions. In *International conference on machine learning* (pp. 5133–5141).

Cover, T. M. (1999). *Elements of information theory*. John Wiley & Sons.

Dezfouli, A., Ashtiani, H., Ghattas, O., Nock, R., Dayan, P., & Ong, C. S. (2019). Disentangled behavioural representations. In *Advances in neural information processing systems* (pp. 2254–2263).

Dezfouli, A., Griffiths, K., Ramos, F., Dayan, P., & Balleine, B. W. (2019). Models that learn how humans learn: the case of decision-making and its disorders. *PLoS computational biology*, *15*(6), e1006903.

Fintz, M., Osadchy, M., & Hertz, U. (2021). Using deep learning to predict human decisions and cognitive

models to explain deep learning models. *bioRxiv*. doi: 10.1101/2021.01.13.426629

Lu, Q., Hasson, U., & Norman, K. A. (2020). Learning to use episodic memory for event prediction. *bioRxiv*.

Shen, S., & Ma, W. J. (2016). A detailed comparison of optimality and simplicity in perceptual decision making. *Psychological review*, *123*(4), 452.

Song, M., Niv, Y., & Cai, M. B. (2020). Learning what is relevant for rewards via value-based serial hypothesis testing. In *42nd annual meeting of the cognitive science society.*

Suchow, J. W., Griffiths, T. L., & Hartshorne, J. K. (2020). Workshop on scaling cognitive science. In *the 42nd annual virtual meeting of the cognitive science society.*

Wang, J. X., Kurth-Nelson, Z., Kumaran, D., Tirumala, D., Soyer, H., Leibo, J. Z., ... Botvinick, M. (2018). Prefrontal cortex as a meta-reinforcement learning system. *Nature neuroscience*, *21*(6), 860–868.

Yang, G. R., Joglekar, M. R., Song, H. F., Newsome, W. T., & Wang, X.-J. (2019). Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, *22*(2), 297–306.