Supplemental Results

Supplemental simulation 1: Pong to breakout generalization results

As described in the main text, but the model learned representations from Pong games, and learned to play Pong first, and then generalized to Breakout. Results in Fig. S2.

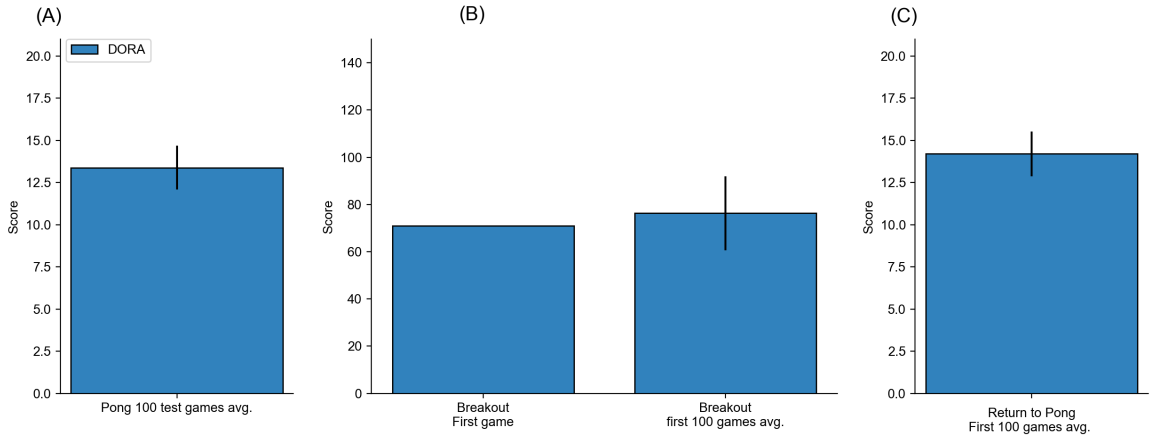

Fig. S2. Results of simulations with DORA trained on Pong and generalizing to Breakout, with DORA learning representations from Pong and DORA learning representations from CLEVR. Error bars represent 2 stderrors. (A) Performance of DORA on Breakout as an average of 100 test games. (B) Results of DORA and humans playing Pong after training on Breakout as the score of the first game played and an average score of the first 100 games played. (C) Results of DORA and humans when returning to play Breakout after playing or learning to play Pong, as an average score for the first 100 games played.

Supplemental data 1:

Two human novices were trained on Breakout for 300 games, then transferred to playing Pong for 100 games, followed by moving back to Breakout for 100 games (these games were played in 2 hours session spread across 6 days; the last 50 games of Breakout and first 20 games of Pong were completed in the same session). Human players, of course, come into playing these games with a life of experience with the world, spatial relations, and other video games, and bring this experience to bear on playing both games. As humans regularly engage in cross-domain generalization, we expect the participants to generalize between games. A comparison of these highly trained humans and DORA and the various DNNs tested in the main text appears in Figure S2.

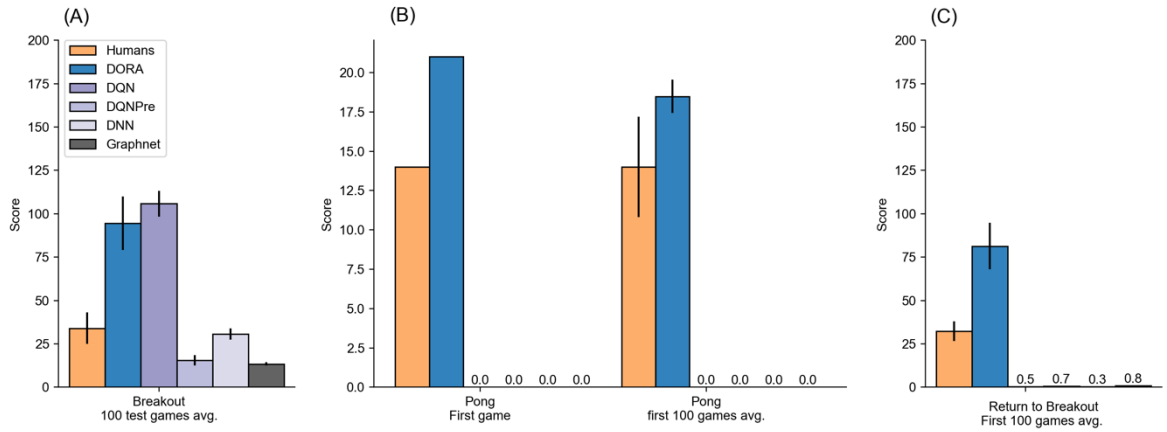

Figure S2. Results of game play simulations with Humans, DORA, the DQNs and the DNNs. Error bars represent 2 stderrors. (A) Performance humans and networks on Breakout as an average of 100 test games. (B) Results of

humans and networks playing Pong after training on Breakout as score on the first game played and mean score over the first 100 games played. (C) Results of humans and networks when returning to play Breakout after playing or learning to play Pong as an average of the first 100 games played.

S2.

In addition to the two participants who played several hundred games of Breakout and Pong, we ran 8 additional participants in a simple transfer task. Participants either played Breakout for 50 minutes followed by playing Pong for 10 minutes (4 participants) or played Pong for 50 minutes followed by playing Breakout for 10 minutes (4 participants). We had players play to a time limit rather than a number of games, as a game of Pong takes roughly 4 times as long as a game of Breakout. The average score on the first game of Pong when played first was 6.25 vs. 14.0 when played after Breakout. The average score on the first game of Breakout when played first was 9.5 vs. when played 19.0 when played after Pong. We analyzed the effects of order (whether a game was played first or after another game) on performance use a simple linear mixed effects model with Score predicted by order with participants as a random variable. Because the scores in Breakout and Pong are on different scales (Pong goes to 21, Breakout is (theoretically) unbounded) we normalized all scores by subtracting each score from the grand mean of scores on that game (e.g., each Pong score had the mean of score of all Pong games subtracted from it). We then compared the model using order to predict score (with participants as a random variable) to the null model. The full model explained significantly more variance than the null model (chi-squre(1) = 4.07, $p < 0.05$), with scores in the transfer condition significantly higher than scores in the initial game condition.

Participants were run using the online javatari system (https://javatari.org/).

One noteworthy limitation of DORA's gameplay is that it was slower to learn Breakout than the human players we tested. We suspect the reason for this limitation is that in most simulations, DORA, like the DNNs we ran for comparison, began as a *tabula rasa* with no understanding of anything at the beginning of learning. As a result, DORA spent much of its early experience in these simulations simply acquiring basic relational concepts such as *left-of* (). By contrast, most people start playing video games long after they have acquired such basic concepts. In other words, our ability to play a game such as Breakout, even for the first time, is already facilitated by an enormous amount of cross-domain transfer: People know what to look for (e.g., "where is the paddle relative to the ball?") and how to represent the answer ("to the left of the ball") even before starting to learn how to play the game. DORA, by contrast, had to learn these basic concepts while learning to play the game.

We argue that the reason DORA learned so much faster than the DNNs is that DORA was biased from the beginning to look for the right thing. Whereas DNNs search for representations that minimize the error in the input-output mapping of the task at hand, DORA looks for systematic relations that allow it to build a model of the task it is learning. Once it has learned this model, DORA is off and running, prepared to transfer its learning to new tasks, such as Pong. By contrast, the DNN is trying to be the best it can at *exactly this one task*; it is trying to memorize exactly what to do in response to every possible situation. In the end, the DNN will be a better Breakout player than DORA. But DORA, unlike the DNN, will be able to transfer its learning to other tasks, including but not limited to Pong.

We argue that people are more like DORA than a DNN. You and I will never beat a well-trained DNN at chess, or go, or probably any other task on which a DNN has been adequately trained. But at the end of the day, we will be able to drive home, make dinner, put our children

to bed, and have a glass of wine. All the DNN will know how to do is beat the next competitor. And more importantly, the DNN will never be able to learn how to perform these other tasks without forgetting how to play chess. A human is a general-purpose learning machine that exceeds at using what it already knows to bootstrap its learning of things it doesn't already know. A deep net is a one-act pony, the best in the world at its one over-trained task, good for absolutely nothing else.

Supplemental simulation 2: Inverse Breakout

We ran a simple simulation of this capacity using a modified version of Breakout. In this version, the rules were adjusted such that missing the ball was rewarded and hitting the ball was punished (i.e., points were scored when the ball went past the paddle, and a life was lost when the ball struck the paddle; essentially the reverse of the regular Breakout rules). We ran tested a version of the DORA model that had previously learned to play Breakout successfully (see simulation 2, main text). Unsurprisingly, initially the model followed the previously successful strategy of following the ball to contact it and send it towards the point-scoring bricks. However, upon contact with the paddle, the model was punished with a lost life. As noted in the main text, the model had previously learned that following the ball predicted reward (points), and that moving away from the ball predicted punishment (lost life). After three lost lives, DORA attempted to compare the representation of the current game to the representation it had previously learned from Breakout.

The current representation was that moving the paddle toward the ball resulted in punishment, or *left-of* (ball, paddle1) then *left-of* (paddle2, paddle1) → punishment signal. The previous representation of the game was that moving toward the ball resulted in reward and away from the ball resulted in punishment, or *left-of* (ball, paddle1) then *left-of* (paddle2, paddle1) → reward signal, and *left-of* (ball, paddle1) then *right-of* (paddle2, paddle1) → punishment signal. As described in the main text, DORA performed mapping and relational inference with these two representations. With P1: *left-of* (paddle2, paddle1) → punishment signal in the driver, and P2: *left-of* (paddle2, paddle1) → reward signal and P3: *right-of* (paddle2, paddle1) → punishment signal in the recipient, DORA mapped *left-of* (paddle2, paddle1) in P1 to *left-of* (paddle2, paddle1) in P2, and punishment signal in P1 to punishment signal in P3. DORA then flipped the driver and recipient (P2 and P3 now in the driver and P1 in the recipient) and performed relational inference. During relational inference, it copied the unmapped reward-signal from P2 and *right-of* (paddle2, paddle1) from P3 into the recipient, thus inferring that *right-of* (paddle2, paddle1) predicts a reward signal. When adopting this strategy, moving away from the ball, DORA started scoring points (because the new task was so easy, we had to decide a point total to stop the game).