A Appendix

A.1 Program induction with dreamcoder

DreamCoder was run in two modes for the reported analyses: with and without library learning. All parameters matched between the two modes except those exclusive to library learning: 4 iterations of DreamCoder, 10 CPUs, 2 minute timeout for program enumeration, 2 minute timeout for recognition model training, and 50% of tasks used to train the recognition model were randomly sampled "dreams". The recognition model was trained to predict a unigram distribution over DSL primitives and library components. No task batching was enabled, so every task was solved at every iteration. Parameters only relevant to library learning were: structure penalty $\lambda = 1.5$, refactoring steps $n = 1$.

A.2 Details of training and hyperparameter tuning of meta-reinforcement learning agents

Meta-learning can be considered as a bi-level optimization problem where there is an outer-loop of learning in which the model learns a useful inductive bias across different tasks of the same task distribution and an inner-loop of learning which takes that inductive bias and rapidly learns or adapts within a specific task [1]. In recurrent-based meta-reinforcement learning architectures like ours, in which tasks are fed sequentially to the model, the outer loop is explicitly implemented as a reinforcement learning algorithm that updates the weights across tasks and the inner loop is implicitly implemented in the activation dynamics of the recurrent network [2, 3] which employs fast adaptation within a specific task. See Ortega et al. [4] for a formal explanation of this adaptation. In our case, the LSTM weights are updated in the outer loop across different grids to give the LSTM a useful prior learned from patterns or abstractions seen across different grid tasks. The activation dynamics within the LSTM utilizes this prior, along with the history of observations within the episode, to implement a fast algorithm to solve the current grid task for the inner loop.

This work's agent's encoder processes the board through a convolutional layer and a fully connected layer and passes this output into the LSTM along with the previous action and reward (see Fig. 3B). To implement the grounding mechanism (Fig.6A), we take the encoder's output, and pass it through two additional fully connected layers to predict the task embedding. All agents were trained using Proximal Policy Optimization (PPO; Schulman et al. 5 using the Stable Baselines package Raffin et al. 6) for one million episodes. We performed a hyperparameter sweep separately for the agents without the grounding loss (i.e. original agents) and with the grounding loss, since we have to tune the new c_{task} weight on the grounding loss jointly. We performed a hyperparameter sweep for: batch size, n_steps (number of steps to run in an environment update), gamma, learning rate, learning rate schedule (constant or linear), clip range, number of epochs, the λ for Generalized Advantage Estimate (GAE λ), max grad norm, activation function, value loss coefficient, entropy coefficient, and grounding loss coefficient for agents with the grounding loss. The hyperparameter sweep was done by sampling from the space of hyperparameter using the Tree-Structured Parzen Estimator [7]. We evaluated 200 samples of hyperparameters from the space for all agents. Both grounding and non-grounding agents used the same hyperparameter spaces to sweep over. We initially did a separate hyperparameter sweep for different grounding agents, but we found in initial experiments that they all reached similar hyperparameter values and training reward after the search. Hyperparameters were evaluated by training on 100,000 episodes and looking at the training reward. The environments used during test time (Fig.3B and Fig. 6) were completely held-out during this process. We trained each grounding agent (and the non-grounding agent) fifteen times and ran their policy fifteen times for each held-out test task distribution (GSP and control tasks). Training was run on a university cluster using NVIDIA P100 GPUs with 16 GB of memory.

A.3 Gibbs Sampling with People Experiment

To generate a task distribution of boards directly from humans we used Gibbs Sampling with People, or GSP (see Harrison et al. 8 and a similar task for binary sequences in Griffiths et al. 9). GSP samples internal prior distributions by putting humans "in the loop" of a Gibbs sampler. In our case, the stimulus space consisted of the space of 4×4 boards, and each of the 16 stimulus dimensions corresponded to the binary color of each tile, namely, red or blue. One of these dimensions was masked out (i.e. "greyed") for the prediction task. Each GSP trial consisted of a prediction

Table 1: Hyperparameters Chosen

Agent	No Grounding Loss	Grounding Loss
batch_size	16	256
n_steps	2048	8
gamma	0.9	0.9
learning_rate	0.000516501	0.000376021
lr_schedule	linear	linear
ent_coef	1.3907E-05	1.45674E-06
clip_range	0.3	0.3
n_epochs	10	5
gae_lambda	0.8	0.95
max_grad_norm	2	0.6
vf_coef	0.000914363	0.016291309
activation_fn	relu	tanh
grounding_coef	0	0.494866282

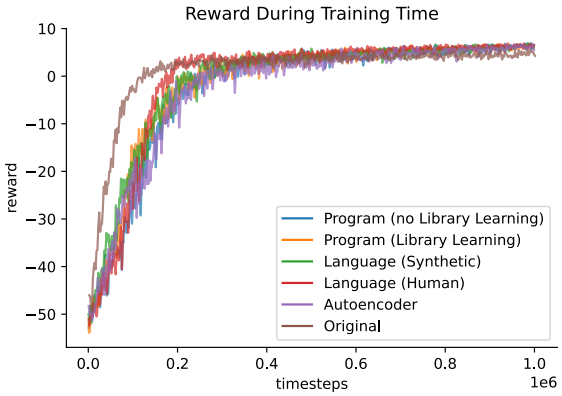

Figure 1: Reward Curves of Different Grounding Agents

task of predicting what color the single masked square is in the grid conditional on the colors of all other squares on the grid. Once a decision is made, the resulting stimulus is passed on to a new participant who repeats the task with another masked square and so on. A sample is generated once a full sweep through the all sixteen squares is completed, similar to the standard procedure of Gibbs sampling. In each trial, participants were presented with a board with one of its tiles covered (indicated by a white tile) as well as the following prompt "what should be the underlying color of the covered white tile such that the board is described by a very simple rule?". They then delivered their answer by clicking on a button that corresponded to their color of choice. Overall, we ran 100 GSP chains in parallel for 15 sweeps each, and chains were initialized with randomly sampled boards. The order in which tiles were masked out within each sweep was also randomized across chains to avoid potential biases.

Participants were recruited on Amazon Mechanical Turk (AMT) and a total of 272 participants completed the study. To ensure that participants did not suffer from any color perception deficiencies, we ran the Ishihara color blindness test [10] as a pre-screening task. This also helped in screening out automated scripts ("bots") that masquerade as participants [11].

A.4 Human Language Description Experiment

We took 500 of the highest probability GSP boards and randomly assigned subjects 25 of the boards such that each subject gets a unique set of boards and each board gets 9-12 descriptions each from different people. The prompt people were given was: "Your goal is to describe this pattern of red squares in words. Be as detailed as possible. Someone should be able to reproduce the entire board given your description. You may be rewarded based on how detailed your description is." We used Prolific (`http://www.prolific.co`) for anonymous subject recruiting. Participants were paid at a rate of approximately $12 per hour (with a total study cost of $1844) and were fully anonymized through the whole process. All text data focused on just the board stimuli and did not include any personally-identifiable content. The experiment contained writing about artificial grid stimuli, so potential risks to subjects were very minimal. Subjects gave their consent through a

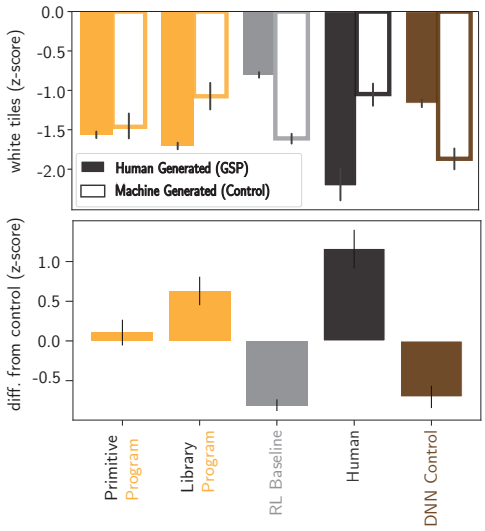

Figure 2: Grounding results for control experiment where we co-train with DNN representations from the hidden layer of a network trained to predict randomly held-out tiles of the GSP boards. Although co-training with this network does improve performance on human and machine generated boards relative to the original agent, it does not selectively improve performance on human-generated boards and impair performance on machine-generated boards like co-training with the DreamCoder recognition model does.

standard consent form at the start of the experiment (where they had to confirm that they agree and give their consent).

A.5 DNN Co-training Control Experiment

For the program results, we co-trained the RL agent with representations from the recognition DNN model used in DreamCoder that proposes programs given a board. The recognition model's architectural modifications, training objective, and data distribution play a critical role in developing useful representations that can be distinguished from a more generic DNN trained on the GSP boards. The DreamCoder DNN's architecture is built to specifically predict a probability distribution over the current routines in the library (this includes high-level routines added to the DSL during library learning). The recognition model is trained to predict the most likely programs for a given grid by balancing a program's description length and its score (its training objective). In addition, the data it trains on comes both from the GSP task dataset, as well as from grids that are randomly sampled from the current DSL/grammar ("dreams": its training data).

As a comparison point, we provide results (Fig. A2) when cotraining with representations from a "generic DNN" that is trained on a more standard task on the GSP boards (predicting randomly held-out tiles). The results show that the selective performance improvement on human-generated tasks vs machine-generated tasks is unique to cotraining representations from the DreamCoder DNN as opposed to a standard DNN. The recognition model's specific architectural modifications, training objective, and data distribution used for program induction within the DreamCoder framework play a critical role in developing representations that can be distinguished from those of a more 'vanilla' DNN and can serve as a distinct and effective cotraining target that can better instill human inductive biases.

A.6 Broader Impact Statement

This work aims to guide artificial agents through human-like behavior. One way in which this work achieves this goal is by co-training with human-generated language descriptions. Although all language data collected in this study was anonymized and devoid of personally-identifiable information, a larger discussion is to be had in scaling this approach up while respecting the privacy

and anonymity of people's data. Should an approach like ours be massively scaled, a conversation about careful checks on data privacy must be had in order to prevent leakage of private information in downstream agent behaviors.

Current artificial systems are not built explicitly to exhibit human-like qualities or behavior. Our work on explicitly training in human-like biases into artificial systems' behavior can be greatly beneficial in terms of being more easily interpretable by humans and being more easily amenable to human-machine collaborations.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes]
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See towards the end of the discussion section.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See section A.4 of the Appendix.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We discuss potential negative societal impacts in section A.4 and human data collection in section A.3.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code will be available in the supplementary material for reviewers and is publicly available here.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See section A.2 of appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See figure captions and section A.2 of appendix.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See section A.2 of appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite the authors responsible for creating the task paradigm that we used.

(b) Did you mention the license of the assets? [N/A]

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Data and code are available here.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See section A.3 on human data collection of appendix.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] ee section A.3 on human data collection of appendix.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] See section A.3 on human data collection of appendix.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [Yes] See section A.3 on human data collection of appendix.

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] See section A.3 on human data collection of appendix.

References

[1] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[2] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[3] Matthew Botvinick, Sam Ritter, Jane X Wang, Zeb Kurth-Nelson, Charles Blundell, and Demis Hassabis. Reinforcement learning, fast and slow. *Trends in cognitive sciences*, 23(5): 408–422, 2019.

[4] Pedro A Ortega, Jane X Wang, Mark Rowland, Tim Genewein, Zeb Kurth-Nelson, Razvan Pascanu, Nicolas Heess, Joel Veness, Alex Pritzel, Pablo Sprechmann, et al. Meta-learning of sequential strategies. *arXiv preprint arXiv:1905.03030*, 2019.

[5] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[6] Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. Stable baselines3, 2019.

[7] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[8] Peter Harrison, Raja Marjieh, Federico Adolfi, Pol van Rijn, Manuel Anglada-Tort, Ofer Tchernichovski, Pauline Larrouy-Maestri, and Nori Jacoby. Gibbs sampling with people. *Advances in Neural Information Processing Systems*, 33:10659–10671, 2020.

[9] Thomas L Griffiths, Dylan Daniels, Joseph L Austerweil, and Joshua B Tenenbaum. Subjective randomness as statistical inference. *Cognitive psychology*, 103:85–109, 2018.

[10] JH Clark. The ishihara test for color blindness. *American Journal of Physiological Optics*, 1924.

[11] Michael Chmielewski and Sarah C Kucker. An mturk crisis? shifts in data quality and the impact on study results. *Social Psychological and Personality Science*, 11(4):464–473, 2020.