APPENDIX

Anonymous authors
Paper under double-blind review

A INDUCING AND EXECUTING OPERATORS

In the main text, we examplify the induction and the execution process using a binary operator. Here, we discuss other details regarding the formulation for all three types of operators, *i.e.*, unary, binary, and ternary.

Unary Operator To induce the unary operator $\mathcal{T}_u^a$ for an attribute a, we solve the following optimization problem

$$\begin{aligned}\mathcal{T}_u^a = \arg\min_{\mathcal{T}} \ell_u^a(\mathcal{T}) = 1/5 \times \Big(&\mathbb{E}\left[\left\|M(b_{o,1}^a)\mathcal{T} - M(b_{o,2}^a)\right\|_F^2\right] + \mathbb{E}\left[\left\|M(b_{o,2}^a)\mathcal{T} - M(b_{o,3}^a)\right\|_F^2\right] + \\ &\mathbb{E}\left[\left\|M(b_{o,4}^a)\mathcal{T} - M(b_{o,5}^a)\right\|_F^2\right] + \mathbb{E}\left[\left\|M(b_{o,5}^a)\mathcal{T} - M(b_{o,6}^a)\right\|_F^2\right] + \\ &\mathbb{E}\left[\left\|M(b_{o,7}^a)\mathcal{T} - M(b_{o,8}^a)\right\|_F^2\right]\Big) + \lambda_u^a \left\|\mathcal{T}\right\|_F^2,\end{aligned} \tag{1}$$

where the indexing follows the row / column major. By taking the derivative with respect to $\mathcal{T}$ and setting it to be $\mathbf{0}$, we have the following solution,

$$\mathcal{T}_u^a = A^{-1}B \tag{2}$$

where, assuming independence,

$$\begin{aligned}A = &\mathbb{E}\left[M(b_{o,1}^a)^T M(b_{o,1}^a)\right] + \mathbb{E}\left[M(b_{o,2}^a)^T M(b_{o,2}^a)\right] + \mathbb{E}\left[M(b_{o,4}^a)^T M(b_{o,4}^a)\right] + \\ &\mathbb{E}\left[M(b_{o,5}^a)^T M(b_{o,5}^a)\right] + \mathbb{E}\left[M(b_{o,7}^a)^T M(b_{o,7}^a)\right] + 5\lambda_u^a I\end{aligned} \tag{3}$$

and

$$\begin{aligned}B = &\mathbb{E}\left[M(b_{o,1}^a)^T\right]\mathbb{E}\left[M(b_{o,2}^a)\right] + \mathbb{E}\left[M(b_{o,2}^a)^T\right]\mathbb{E}\left[M(b_{o,3}^a)\right] + \mathbb{E}\left[M(b_{o,4}^a)^T\right]\mathbb{E}\left[M(b_{o,5}^a)\right] + \\ &\mathbb{E}\left[M(b_{o,5}^a)^T\right]\mathbb{E}\left[M(b_{o,6}^a)\right] + \mathbb{E}\left[M(b_{o,7}^a)^T\right]\mathbb{E}\left[M(b_{o,8}^a)\right].\end{aligned} \tag{4}$$

Note that as long as $\lambda_u^a > 0$, A is a symmetric positive definite matrix and hence is invertible. Compared to the binary case, the unary operator can be regarded as a special binary operator where one of the operand is a constant, absorbed into operator learning, and jointly solved.

To predict the answer representation, we solve another optimization problem, *i.e.*,

$$\widehat{M_u^a} = \arg\min_{M} \ell_u^a(M) = \mathbb{E}\left[\left\|M(b_{o,8}^a)\mathcal{T}_u^a - M\right\|_F^2\right]. \tag{5}$$

Taking its derivative and setting it to $\mathbf{0}$, we have

$$\widehat{M_u^a} = \mathbb{E}\left[M(b_{o,8}^a)\right]\mathcal{T}_u^a. \tag{6}$$

Note that this is exactly the execution of the learned operator.

Binary Operator The optimization problem for the binary case can be expanded as

$$\begin{aligned}\mathcal{T}_b^a = \arg\min_{\mathcal{T}} \ell_b^a(\mathcal{T}) = 1/2 \times \Big(&\mathbb{E}\left[\left\|M(b_{o,1}^a)\mathcal{T}M(b_{o,2}^a) - M(b_{o,3}^a)\right\|_F^2\right] + \\ &\mathbb{E}\left[\left\|M(b_{o,4}^a)\mathcal{T}M(b_{o,5}^a) - M(b_{o,6}^a)\right\|_F^2\right]\Big) + \lambda_b^a \left\|\mathcal{T}\right\|_F^2.\end{aligned} \tag{7}$$

We note that, assuming independence, the solution satisfies

$$\begin{aligned}&\mathbb{E}\left[M(b_{o,1}^a)^T M(b_{o,1}^a)\right]\mathcal{T}\mathbb{E}\left[M(b_{o,2}^a)M(b_{o,2}^a)^T\right] + \\ &\mathbb{E}\left[M(b_{o,4}^a)^T M(b_{o,4}^a)\right]\mathcal{T}\mathbb{E}\left[M(b_{o,5}^a)M(b_{o,5}^a)^T\right] + 2\lambda_b^a\mathcal{T} \\ &= \mathbb{E}\left[M(b_{o,1}^a)^T\right]\mathbb{E}\left[M(b_{o,3}^a)\right]\mathbb{E}\left[M(b_{o,2}^a)^T\right] + \mathbb{E}\left[M(b_{o,4}^a)^T\right]\mathbb{E}\left[M(b_{o,6}^a)\right]\mathbb{E}\left[M(b_{o,5}^a)^T\right].\end{aligned} \tag{8}$$

This is a linear matrix equation and can be turned into a linear equation by vectorization. Using $\mathrm{vec}(A\mathcal{T}B) = A \otimes B\,\mathrm{vec}(\mathcal{T})$ (Lancaster, 1970), where $\otimes$ denotes the Kronecker product, we have

$$\mathrm{vec}(\mathcal{T}_b^a) = A^{-1}B, \tag{9}$$

where

$$\begin{aligned} A =& \mathbb{E}\left[M(b^a_{o,1})^T M(b^a_{o,1})\right] \otimes \mathbb{E}\left[M(b^a_{o,2}) M(b^a_{o,2})^T\right] + \\ & \mathbb{E}\left[M(b^a_{o,4})^T M(b^a_{o,4})\right] \otimes \mathbb{E}\left[M(b^a_{o,5}) M(b^a_{o,5})^T\right] + 2\lambda^a_b I \end{aligned} \tag{10}$$

and

$$\begin{aligned} B =& \text{vec}\left(\mathbb{E}\left[M(b^a_{o,1})^T\right] \mathbb{E}\left[M(b^a_{o,3})\right] \mathbb{E}\left[M(b^a_{o,2})^T\right]\right) + \\ & \text{vec}\left(\mathbb{E}\left[M(b^a_{o,4})^T\right] \mathbb{E}\left[M(b^a_{o,6})\right] \mathbb{E}\left[M(b^a_{o,5})^T\right]\right). \end{aligned} \tag{11}$$

Note that A is also symmetric positive definite given positive λ^a_b and hence invertible.

The predicted answer representation is given by

$$\widehat{M^a_b} = \arg\min_M \ell^a_b(M) = \mathbb{E}\left[\left\|M(b^a_{o,7}) \mathcal{T}^a_b M(b^a_{o,8}) - M\right\|^2_F\right], \tag{12}$$

which can be solved by executing the induced binary operator $\widehat{M^a_b} = \mathbb{E}\left[M(b^a_{o,7})\right] \mathcal{T}^a_b \mathbb{E}\left[M(b^a_{o,8})\right]$.

Ternary Operator A ternary operation can be regarded as an unary operation on elements defined on rows / columns. Specifically, we propose to construct the algebraic representation of a row / column by concatenating the algebraic representation of each panel in it, *i.e.*,

$$M(b^a_{o,i}, b^a_{o,i+1}, b^a_{o,i+2}) = [M(b^a_{o,i}); M(b^a_{o,i+1}); M(b^a_{o,i+2})]. \tag{13}$$

Then the ternary operator can be solved by

$$\mathcal{T}^a_t = \arg\min_{\mathcal{T}} \ell^a_t(\mathcal{T}) = \mathbb{E}\left[\left\|M(b^a_{o,1}, b^a_{o,2}, b^a_{o,3})\mathcal{T} - M(b^a_{o,4}, b^a_{o,5}, b^a_{o,6})\right\|^2_F\right] + \lambda^a_t \left\|\mathcal{T}\right\|^2_F. \tag{14}$$

Similar to the unary case discussed above,

$$\mathcal{T}^a_t = A^{-1} B \tag{15}$$

where

$$A = \mathbb{E}\left[M(b^a_{o,1}, b^a_{o,2}, b^a_{o,3})^T M(b^a_{o,1}, b^a_{o,2}, b^a_{o,3})\right] + \lambda^a_t I \tag{16}$$

and

$$B = \mathbb{E}\left[M(b^a_{o,1}, b^a_{o,2}, b^a_{o,3})^T\right] \mathbb{E}\left[M(b^a_{o,4}, b^a_{o,5}, b^a_{o,6})\right]. \tag{17}$$

Correspondingly, the answer representation can be obtained by first executing the ternary operator $\mathbb{E}\left[M(b^a_{o,4}, b^a_{o,5}, b^a_{o,6})\right] \mathcal{T}^a_t$ and slicing it from the result.

To compute the operator distribution, we model it based on the fitness of each operator type,

$$P(\mathcal{T}^a = \mathcal{T}^a_u \mid \{I_{o,i}\}^8_{i=1}) \propto \exp(-\ell^a_u(\mathcal{T}^a_u)) \tag{18}$$

$$P(\mathcal{T}^a = \mathcal{T}^a_b \mid \{I_{o,i}\}^8_{i=1}) \propto \exp(-\ell^a_b(\mathcal{T}^a_b)) \tag{19}$$

$$P(\mathcal{T}^a = \mathcal{T}^a_t \mid \{I_{o,i}\}^8_{i=1}) \propto \exp(-\ell^a_t(\mathcal{T}^a_t)). \tag{20}$$

B Instances of Operators

In the original work of Zhang et al. (2019) and Hu et al. (2020), there are four operators: Constant, Progression, Arithmetic, and Distribute of Three. Progression is parameterized by its step size ($\pm 1/2$). Arithmetic includes addition and subtraction. And Distribute of Three is implemented as shifting and can be either a left shift or a right one. Note that Constant can be regarded as special Progression with a step size of 0. In this work, we group all four operators into three types: unary (Constant and Progression), binary (Arithmetic), and ternary (Distribute of Three).

To study systematic generalization in abstract relation learning, we use the Raven's Progressive Matrices (RPM) generation method proposed in (Zhang et al., 2019; Hu et al., 2020) and carefully split data into three regimes:

- Systematicity: The training set and the test set contain all three types of operators but disjoint instances. Specifically, the training set has Constant, Progression of ± 1, addition in Arithmetic, and left shift in Distribute of Three, while in the test set there are Progression of ± 2, subtraction in Arithmetic, and right shift in Distribute of Three.
- Productivity: The training set contains only unary operators and the test set only binary operators. Specifically, the training set has Constant and all instances of Progression, while the test set all instances of Arithmetic.
- Localism: The training set contains only binary operators and the test set only unary operators. Specifically, the training set has all instances of Arithmetic and the test set Constant and all instances of Progression.

Table 1: The network architecture used for each branch of the object CNN.

Operator	Parameters
Convolution	$[6, 5, 1]$
BatchNorm	6
SoftPlus	
MaxPool	2
Convolution	$[16, 5, 1]$
BatchNorm	16
SoftPlus	
MaxPool	2
Linear	120
SoftPlus	
Linear	84
SoftPlus	
Linear	m
LogSoftMax	

C Implementation Details

C.1 Network Architecture

We use a LeNet-like architecture (LeCun et al., 1998) for each branch of the object CNN. See Table 1 for the design. Note that the object CNN consists of four branches, including objectiveness, type, size, and color. The parameters for Convolution denote the output channel size, kernel size, and stride, respectively. A BatchNorm layer is parameterized by the number of channels, whereas a MaxPool layer by its stride. An output size is used to specify a Linear layer's parameter. m equals 2, 5, 6, 10 for objectiveness, type, size, and color, respectively. For numerical stability, we use LogSoftMax to turn a probability simplex into its log space.

C.2 Other Hyperparameters

For the inner regularized linear regression, we set different regularization coefficients for different attributes but, for the same attribute, we keep them the same across all three types of operators. For position, $\lambda = 10^{-4}$. For number, $\lambda = 10^{-6}$. For type, $\lambda = 10^{-6}$. For size, $\lambda = 10^{-6}$. For color, $\lambda = 5 \times 10^{-7}$. All of the regularization terms in Eq. (9) in the main text are set to be 1 and $\{M_0^a\}$ and $\{M^a\}$ are initialized as 2×2 square matrices.

For training, we first train for 10 epochs parameters regarding objectiveness, including the objectiveness branch, and the representation matrices on position and number. We then perform 2 rounds of cyclic training on parameters regarding type, size, and color, each of which experiences 10 epochs of updates in a round. Finally, we fine-tune all parameters for another 10 epochs, totaling up to 80 training epochs. The entire system is optimized using ADAM (Kingma & Ba, 2014) with a learning rate of 9.5×10^{-5}.

References

Sheng Hu, Yuqing Ma, Xianglong Liu, Yanlu Wei, and Shihao Bai. Hierarchical rule induction network for abstract visual reasoning. *arXiv preprint arXiv:2002.06838*, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2014.

Peter Lancaster. Explicit solutions of linear matrix equations. *SIAM review*, 12(4):544–566, 1970.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. Raven: A dataset for relational and analogical visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.