APPENDIX

A Transformations Details

We build IQ tasks with geometric transformation of 3 categories: affine, non-linear and syntactic transformations.

Affine transformations

We consider 5 types of affine transformations: translation, rotation, reflection, shear, and scaling.

- *Translation*: with translation vector (i, j), where $i, j \in \{-9, -6, -3, 0, 3, 6, 9\}$.
- *Rotation*: with rotation angle $\alpha \in \{k \cdot 15^\circ : k \in \{0, 1, \ldots, 23\}\}$.
- *Reflection*: horizontal or vertical reflection.
- *Shear*: with shear angles (α, β) where $\alpha, \beta \in \{-60^\circ, -45^\circ, -30^\circ, -15^\circ, 0^\circ, 15^\circ, 30^\circ, 45^\circ, 60^\circ\}$.
- *Scaling*: with scale coefficient $s \in \{0.5, 0.75, 1, 1.25\}$.

Non-linear transformations

We consider 2 types of non-linear transformations: fisheye and horizontal wave.

- *Fisheye*: given pixel (x, y), the transformed pixel $(T(x), T(y))$ is given by $T(x) = x + (x - c_x) \cdot d \cdot \sqrt{(x - c_x)^2 + (y - c_y)^2}$ and $T(y) = y + (y - c_y) \cdot d \cdot \sqrt{(x - c_x)^2 + (y - c_y)^2}$, where (c_x, c_y) is the transformation center and d is the distortion factor.
- *Horizontal wave*: given pixel (x, y), the transformed pixel $(T(x), T(y))$ is given by $T(x) = x$ and $T(y) = y + a\cos(fy)$, where a is the amplitude of cosine wave and f is the frequency.

Syntactic transformations

We consider 2 types of syntactic transformations: black-white and swap.

- *Black-white*: the image is horizontally or vertically splitted into 2 subimages (not necessarily of equal size). One subimage is kept fixed, while the other one will be transformed $x \mapsto 1 - x$, where x is the pixel value.
- *Swap*: the image is splitted into 4 equal subimages, which are then permuted to achive the transformed image.

B Principles for Designing the Hypothesis Space $\mathcal{F}$ and the Function Composer ϕ

We aim to determine general principles for designing $\mathcal{F}$ and ϕ. Suppose $\mu \colon \mathcal{X} \times \mathcal{Y} \to \mathcal{C}^0(\mathcal{X}, \mathcal{Y})$, where $\mathcal{C}^0(\mathcal{X}, \mathcal{Y})$ is the space of all continuous functions from $\mathcal{X}$ to $\mathcal{Y}$, be the mapping that maps each input-output pair (x, y) to the correct function transforming x to y. We further define a norm $\|.\|_{\mathcal{C}^0}$ on $\mathcal{C}^0(\mathcal{X}, \mathcal{Y})$ determined by $\|f\|_{\mathcal{C}^0} = \sup_{x \in \mathcal{X}} \|f(x)\|_{\mathcal{Y}}$, where $\|.\|_{\mathcal{Y}}$ is an arbitrary norm on $\mathcal{Y}$. Our goal is to find ϕ as the solution of the optimization problem:

$$\text{Minimize} \quad \sum_{(x,y)} \|\mu_{x,y} - \phi_{x,y}\|_{\mathcal{C}^0}. \tag{1}$$

We hypothesize that the cardinality of the range $\mathcal{R}(\mu)$ of μ is much less than the number of data points (i.e. the number of relations within the dataset is limited), and further supppose $\mathcal{R}(\mu) = \{\mu_1, \mu_2, \ldots, \mu_k\}$, where μ_i's are functions in $\mathcal{C}^0(\mathcal{X}, \mathcal{Y})$. The optimization problem in Eq. ((1)) can be rewritten as:

$$\text{Minimize} \quad \sum_{i=1}^{k} \sum_{(x,y):\mu_i(x)=y} \|\mu_i - \phi_{x,y}\|_{\mathcal{C}^0}. \tag{2}$$

The optimization problem in Eq. ((2)) can be deduced to multiple optimization subproblems:

$$\text{Minimize} \sum_{(x,y):\mu_i(x)=y} \|\mu_i - \phi_{x,y}\|_{\mathcal{C}^0}, \qquad \forall i = 1, 2, \ldots, k. \tag{3}$$

For each $i = 1, 2, \ldots, k$, let $\phi_i^* \in \mathcal{F}$ be the function that best approximates μ_i. By the triangle inequality, we obtain

$$\|\mu_i - \phi_{x,y}\|_{\mathcal{C}^0} \leq \|\mu_i - \phi_i^*\|_{\mathcal{C}^0} + \|\phi_i^* - \phi_{x,y}\|_{\mathcal{C}^0}, \qquad \forall i = 1, 2, \ldots, k.$$

Solving optimization problem Eq. ((3)) might be difficult, so we instead consider an alternative optimization problem

$$\text{Minimize} \sum_{(x,y):\mu_i(x)=y} (\|\mu_i - \phi_i^*\|_{\mathcal{C}^0} + \|\phi_i^* - \phi_{x,y}\|_{\mathcal{C}^0}), \qquad \forall i = 1, 2, \ldots, k. \tag{4}$$

We deduce following analysis after observing Eq. ((4)):

- The term $\|\phi_i^* - \phi_{x,y}\|_{\mathcal{C}^0}$ suggests $\mathcal{R}(\phi)$ (the range of ϕ) should not be too small or too large, otherwise there may exist some (x, y) such that $\phi_{x,y}$ is far away from ϕ_i^*.
- Since $\mathcal{R}(\phi)$ is contrained, so should be $\mathcal{F}$. If $\mathcal{F}$ is too small, $\|\mu_i - \phi_i^*\|_{\mathcal{C}^0}$ may be large for some i; if $\mathcal{F}$ is too large, ϕ_i^* may be far away from $\mathcal{R}(\phi)$, which leads to large $\|\phi_i^* - \phi_{x,y}\|_{\mathcal{C}^0}$.

With the above arguments, we suggest the following principles for designing $\mathcal{F}$ and ϕ:

1. $\mathcal{F}$ should be constrained by some prior knowledge of μ. For example, if we know μ is invertible, then $\mathcal{F}$ should also contain invertible functions only.
2. ϕ should be determined on the fly in a meta-learning fashion (associated with each input-output pair (x, y)) so that we can control its complexity.

C Training setup

For ESBN, Transformer, RelationNet and PrediNet, we follow the same settings as **?**, where all given images (including examples and answer candidates) are treated as a sequence and passed through a context normalization layer before being fed to the model. For HyperNetwork, we also use the NICE backbone for fair comparisons and maintain the key memories (but not the value memories) to compute the weights; at each layer of the backbone, the attention weights are computed as the output of an LSTM cell, where the input for LSTM is the concatenation of the input and (pseudo-)output of current layer, and the hidden states are taken from the LSTM cell of previous layer. For FINE with NICE backbone, we use 4 NICE layers while using 2-layer MLP for FINE with MLP backbone. We use 8-32 basis weight matrices in the experiments.

We use the Adam optimizer with no weight decay along with gradient clipping with threshold 10 in all experiments. All tasks are trained with 200-300 epochs. The training and testing batch sizes are 32 and 100, respectively. Feature vectors of images are of size 128.

D More Results

Table 1 reports the full result table with mean & std on Omniglot dataset of single-transformation and multi-affine-transformation task.

	Single-transformation									Multi affine
	Affine					Non-linear		Syntactic		
	Trans.	Rot.	Refl.	Shear	Scale	Fish.	H.Wave	B&W	Swap	
RelationNet	27.1±0.4	26.2±0.3	25.5±0.4	27±0.5	27.5±0.4	26.1±0.7	30.2±6.9	49.7±29.1	26.0±2.2	25.3±0.2
PrediNet	68.5±4.0	43.9±6.8	32.9±1.9	62.4±3.7	65.7±2.8	36.2±2.4	46.1±7.9	60.5±8.0	57.5±3.6	34.9±1.1
HyperNet	88.9±1.0	62.0±2.9	94.0±1.9	74.5±1.4	81.8±1.1	63.2±2.0	80.4±1.0	88.6±1.4	90.1±1.0	54.0±4.1
Transformer	89.5±1.0	64.8±1.5	44.3±0.9	86.3±4.1	84±0.9	41.4±11.6	91.0±11.8	97.6±0.4	49.9±18.5	59.4±6.0
ESBN	79.8±0.6	58.6±1.0	50.1±0.3	83.4±1.6	84.5±1.2	67.1±0.8	86.4±1.0	90.5±4.1	71.6±2.7	63.1±0.5
FINE	**94.3±0.4**	**77.6±0.7**	**95.1±1.0**	**87.2±0.3**	**86.6±0.4**	**78.5±0.7**	**95.9±0.4**	**98.4±0.3**	**96.2±0.2**	**69.1±0.6**

Table 1: Test accuracy (mean & std) (%) on Omniglot dataset.

Table 2 reports the full result table with mean & std on CIFAR100 dataset of single-transformation tasks.

	Affine					Non-linear		Syntactic	
	Trans.	Rot.	Refl.	Shear	Scale	Fish.	H.Wave	B&W	Swap
RelationNet	59.9±11.2	49.6±5.8	29.9±1.0	45.3±5.1	66.2±1.3	28.7±1.0	39.5±6.9	26.2±1.4	29.7±0.9
PrediNet	72.4±4.6	65.6±5.0	40.6±2.0	74.3±4.8	76.1±3.6	37.1±1.2	53.9±8.1	32.7±1.8	39.6±1.3
HyperNet	94.8±1.1	86.8±1.3	46.6±0.5	91.3±0.9	85.2±1.2	46.8±0.7	80.5±4.6	47.8±0.9	46±0.8
Transformer	98.4±1.1	86.3±3.8	47.5±1.1	95.4±1.4	84.9±1.2	47.2±1.0	95.1±1.8	51.6±14.3	47.6±0.8
ESBN	96.6±0.7	81.9±1.1	50.6±0.4	90.1±0.7	81.5±0.9	57.7±1.3	95.7±0.7	68.8±6.0	50.5±0.5
FINE	**99.2±0.1**	**91.3±0.2**	**80.6±14.5**	**95.6±0.5**	**87±0.2**	**76.8±1.3**	**98.3±0.4**	**89.1±0.7**	**51.6±1.7**

Table 2: Test accuracy (%) on CIFAR100 dataset of single-transformation tasks.

E Codes

We use codes from the public repository https://github.com/taylorwwebb/emergent_symbols for baseline models, including RelationNet, PrediNet, Transformer and ESBN.