Human-level concept learning through probabilistic program induction

Unicode Research

Jash Mehta
Deep Gandhi

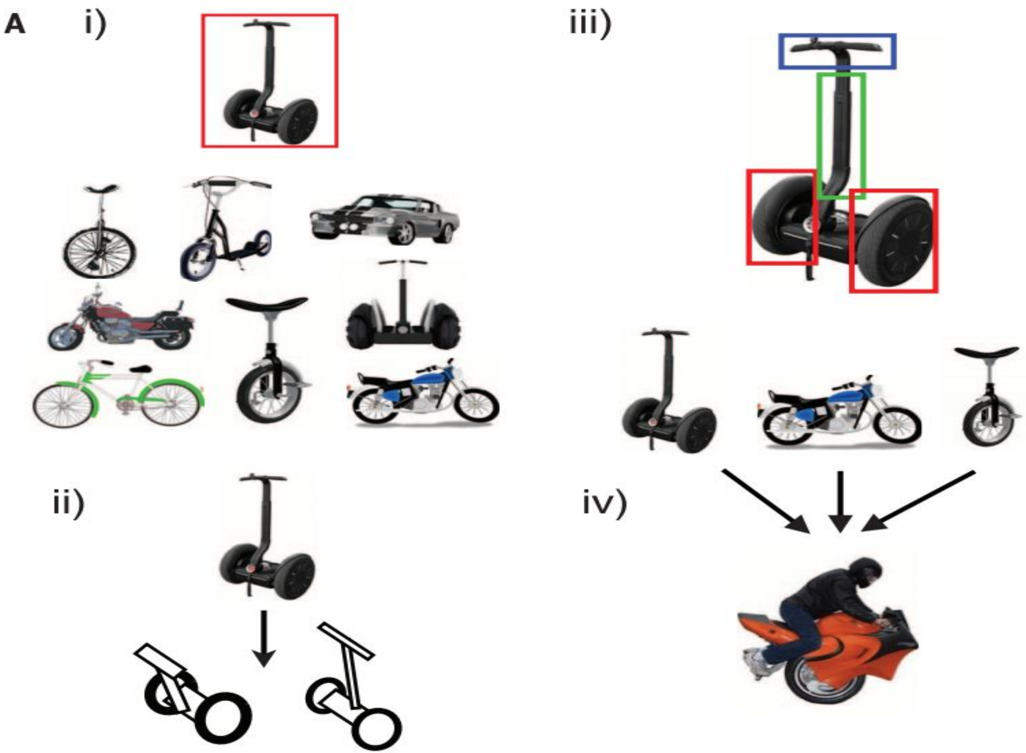
What are the two aspects of human conceptual knowledge which have eluded machine systems?

What are the two aspects of human conceptual knowledge which have eluded machine systems?

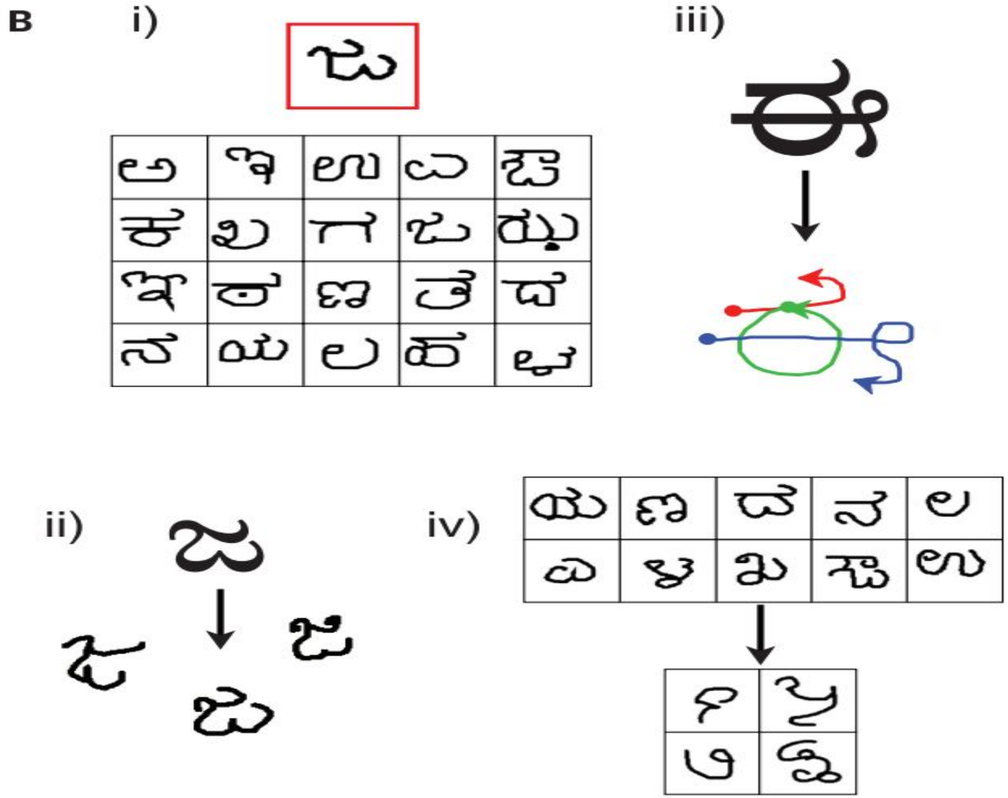
1. People can learn a new concept with just one or a handful of examples. ie making meaning generalizations via “one-shot learning”

2. Humans learn richer representations than machines do; and they use it to:
   (i) create new exemplars
   (ii) parse objects into parts and relations
   (iii) creating new abstract categories

1. People can learn a new concept with just one or a handful of examples.



2. Humans learn richer representations than machines do.



Central Challenge:

1. How to learn concepts with one or a handful of examples?
2. How to learn such abstract, rich, and flexible representations?

Central Challenge:

1. How to learn concepts with one or a handful of examples?
2. How to learn such abstract, rich, and flexible representations?

How can learning succeed from such sparse data yet also produce such rich representations?

Bayesian Program Learning (BPL) framework

- Concepts are represented as probabilistic generative models.
- BPL learns simple stochastic programs to represent concepts, building them compositionally from parts, sub-part and spatial relations,
- BPL defines generative models which sample new types of concepts by combining parts and subparts in new ways. Each new type is also a generative model which produces new examples of concepts.

“BPL is a generative model for generative models”

3 key ideas of BPL

1. **Compositionality**

2. **Causality**

3. **Learning to Learn**

3 key ideas of BPL

1. **Compositionality -**
   Rich concepts can be built from simpler primitives, just like humans tend to break the object down into smaller fundamental parts

2. **Causality -**
   Form generalizations that capture the abstract “causal” structure of real world processes that produce examples of a category.

3. **Learning to Learn -**
   The model develops hierarchical priors that allow previous experience with related concepts to ease the learning of new concepts.

Omniglot Dataset

Fig. 2. Simple visual concepts for comparing human and machine learning. 525 (out of 1623) character concepts, shown with one example each.

$$P(\psi, \theta^{(1)}, ..., \theta^{(M)}, I^{(1)}, ..., I^{(M)})$$

$$= P(\psi)\prod_{m=1}^{M} P(I^{(m)}|\theta^{(m)})P(\theta^{(m)}|\psi)$$

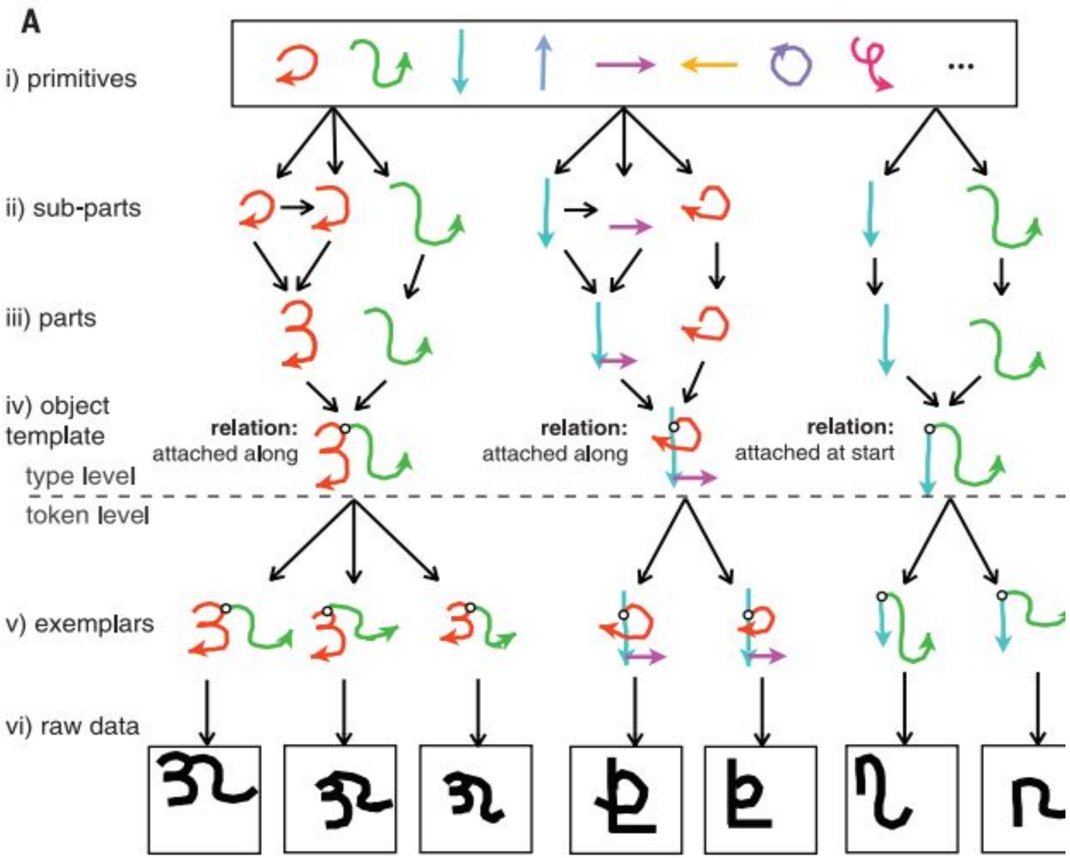
where,

$\boldsymbol{\varphi}$ = character type and $\boldsymbol{\varphi}$ = {κ, S, R}
$\boldsymbol{\theta}$ = token of that type
I = corresponding binary image

A Generative Model of Handwritten Characters

Generating character types

Generating character tokens



1) Generating character types:

$$P(\psi) = P(\kappa) \prod_{i=1}^{\kappa} P(S_i) P(R_i | S_1, ..., S_{i-1}),$$

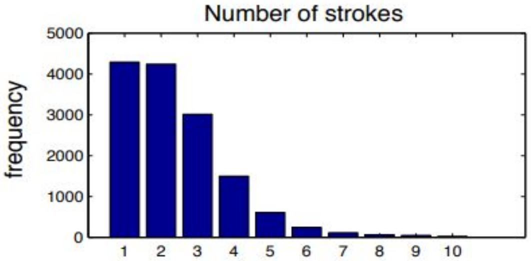
where,

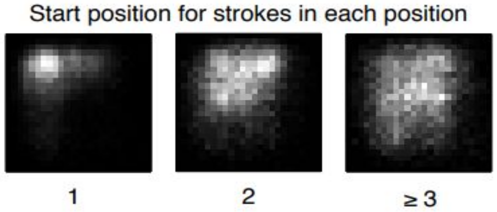
$\boldsymbol{\varphi}$ = character type and $\boldsymbol{\varphi}$ = {κ, S, R}
κ = number of strokes
S = a set of κ strokes
R = relation between strokes

1) Generating character types:

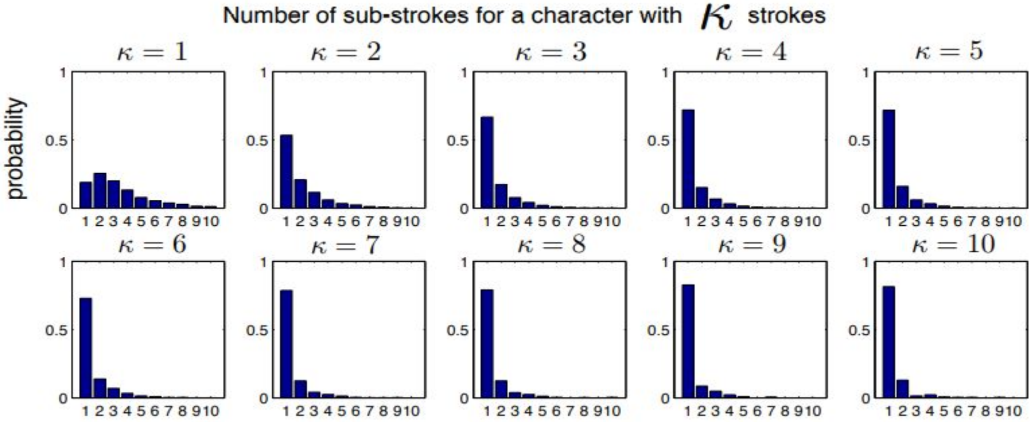
$$P(\psi) = \boxed{P(\kappa)} \prod_{i=1}^{\kappa} P(S_i)P(R_i|S_1, ..., S_{i-1})$$

- The number of strokes κ is sampled from a multinomial P(κ) estimated from the empirical frequencies





- The number of sub-strokes n_i is sampled from the empirical frequency P(n_i | κ) specific to the total number of strokes κ capturing the fact that characters with many strokes tend to have simpler strokes.



1) Generating character types: $P(\psi) = P(\kappa)\prod_{i=1}^{\kappa} \boxed{P(S_i)} P(R_i|S_1, ..., S_{i-1})$

Each sub-stroke s_{ij} is modeled as a uniform cubic b-spline, which can be decomposed into three variables sij = $\{z_{ij}, x_{ij}, y_{ij}\}$ with joint distribution $P(S_i)$

$$P(S_i) = P(z_i)\prod_{j=1}^{n_i} P(x_{ij}|z_{ij})P(y_{ij}|z_{ij})$$

where,

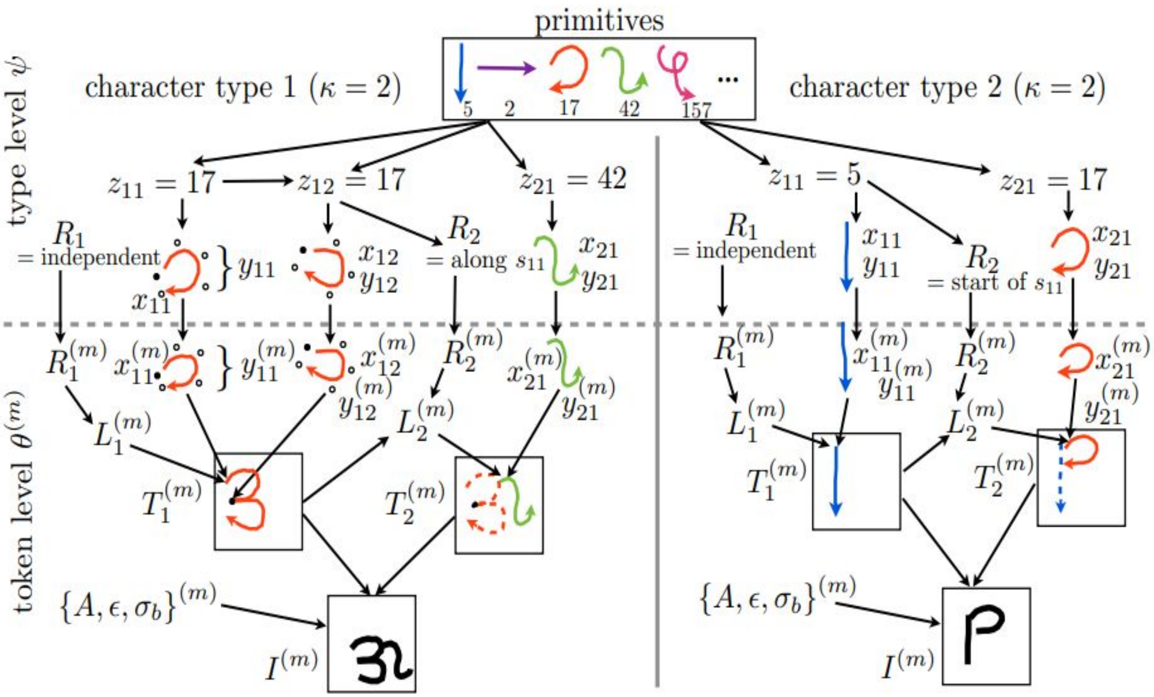
$z_{ij} \in \mathbb{N}$ is an index into the library of primitives

$x_{ij} \in \mathbb{R}^{10}$ small open circles or control points

y_{ij} is type-level scale

1) Generating character types: $P(\psi) = P(\kappa) \prod_{i=1}^{\kappa} \boxed{P(S_i)} P(R_i | S_1, ..., S_{i-1})$

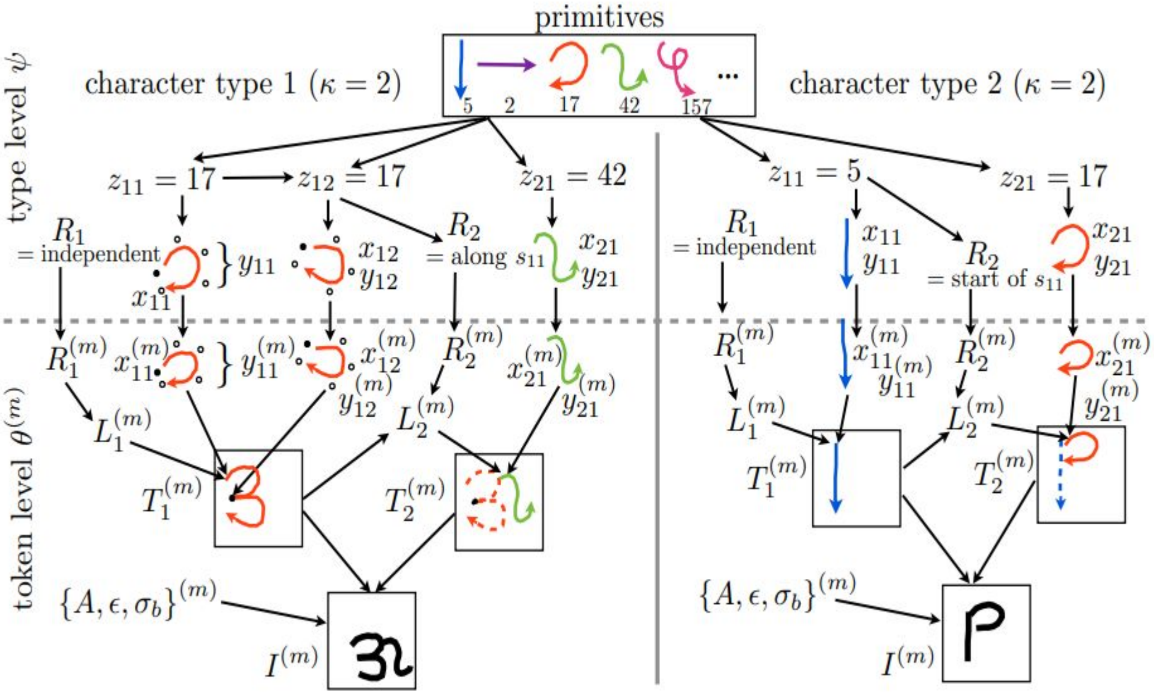
- $P(z_i) = P(z_{i1}) \prod_{j=2}^{n_i} P(z_{ij} | z_{i(j-1)})$
- $P(x_{ij} | z_{ij}) = N(\mu_{z_{ij}}, \Sigma_{z_{ij}})$
- $P(y_{ij} | z_{ij}) = \mathrm{Gamma}(\alpha_{z_{ij}}, \beta_{z_{ij}})$



1) Generating character types: $P(\psi) = P(\kappa) \prod_{i=1}^{\kappa} P(S_i) \boxed{P(R_i|S_1, \ldots, S_{i-1})}$

The spatial relation R_i specifies how the beginning of stroke S_i connects to the previous strokes $\{S_1, \ldots, S_{i-1}\}$

ξi is relation to the previous stoke and
ξi ∈ {Independent, Start, End, Along}



1) Generating character types:

$$\boxed{P(\psi) = P(\kappa) \prod_{i=1}^{\kappa} P(S_i) P(R_i | S_1, ..., S_{i-1})}$$

procedure GENERATETYPE
  $\kappa \leftarrow P(\kappa)$ ▷ Sample the number of strokes
  for $i = 1 ... \kappa$ **do**
    $n_i \leftarrow P(n_i|\kappa)$ ▷ Sample the number of sub-strokes
    $S_i \leftarrow$ GENERATESTROKE(i, n_i) ▷ Sample stroke
    $\xi_i \leftarrow P(\xi_i)$ ▷ Sample relation to previous strokes
    $R_i \leftarrow P(R_i|\xi_i, S_1, ..., S_{i-1})$ ▷ Sample relation details
  end for
  $\psi \leftarrow \{\kappa, R, S\}$
  return @GENERATETOKEN(ψ) ▷ Return program handle
end procedure

procedure GENERATESTROKE(i,n_i)
  $z_{i1} \leftarrow P(z_{i1})$ ▷ Sample the identity of the first sub-stroke
  for $j = 2 \ldots n_i$ **do**
    $z_{ij} \leftarrow P(z_{ij}|z_{i(j-1)})$ ▷ Sample the identities of the other sub-strokes
  end for
  for $j = 1 \ldots n_i$ **do**
    $x_{ij} \leftarrow P(x_{ij}|z_{ij})$ ▷ Sample a sub-stroke's control points
    $y_{ij} \leftarrow P(y_{ij}|z_{ij})$ ▷ Sample a sub-stroke's scale
    $s_{ij} \leftarrow \{x_{ij}, y_{ij}, z_{ij}\}$
  end for
  $S_i \leftarrow \{s_{i1}, \ldots, s_{in_i}\}$ ▷ A complete stroke definition
  return S_i
end procedure

2) Generating character tokens:

$$P(\theta^{(m)}|\psi) = P(L^{(m)}|\theta^{(m)}_{\setminus L^{(m)}}, \psi) \prod_i P(R_i^{(m)}|R_i)P(y_i^{(m)}|y_i)P(x_i^{(m)}|x_i)P(A^{(m)}, \sigma_b^{(m)}, \epsilon^{(m)}).$$

where,

$\theta^{(m)} = \{L^{(m)}, x^{(m)}, y^{(m)}, R^{(m)}, A^{(m)}, \sigma^{(m)}_b, \varepsilon^{(m)}\}$ are token-level variables

$L^{(m)}_i \in R^2$ is starting location

$x_{ij} \in R^{10}$ small open circles or control points

y_{ij} is type-level scale

$A^{(m)} \in R^4$ is image transformation

Blurring is accomplished through a convolution with a Gaussian filter with standard deviation $\sigma^{(m)}_b$

$\varepsilon^{(m)}$ is the probability with which the second noise process stochastically flips pixels

2) Generating character tokens:

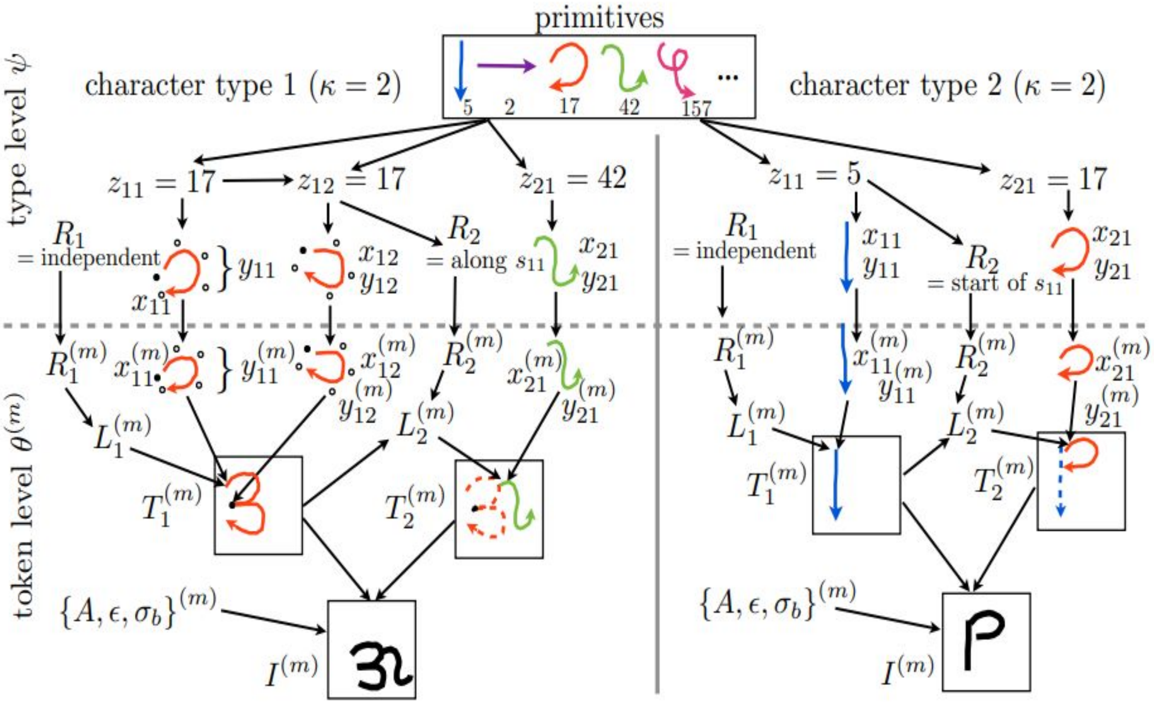
A stroke trajectory $T^{(m)}_i$ (Fig. S1) is a sequence of points in the image plane that represents the path of the pen.

$$T_i^{(m)} = f(L_i^{(m)}, x_i^{(m)}, y_i^{(m)})$$

The control points and scale are noisy versions of their type-level counterparts

$$P(x_{ij}^{(m)} | x_{ij}) = N(x_{ij}, \sigma_x^2 I)$$

$$P(y_{ij}^{(m)} | y_{ij}) \propto N(y_{ij}, \sigma_y^2)$$



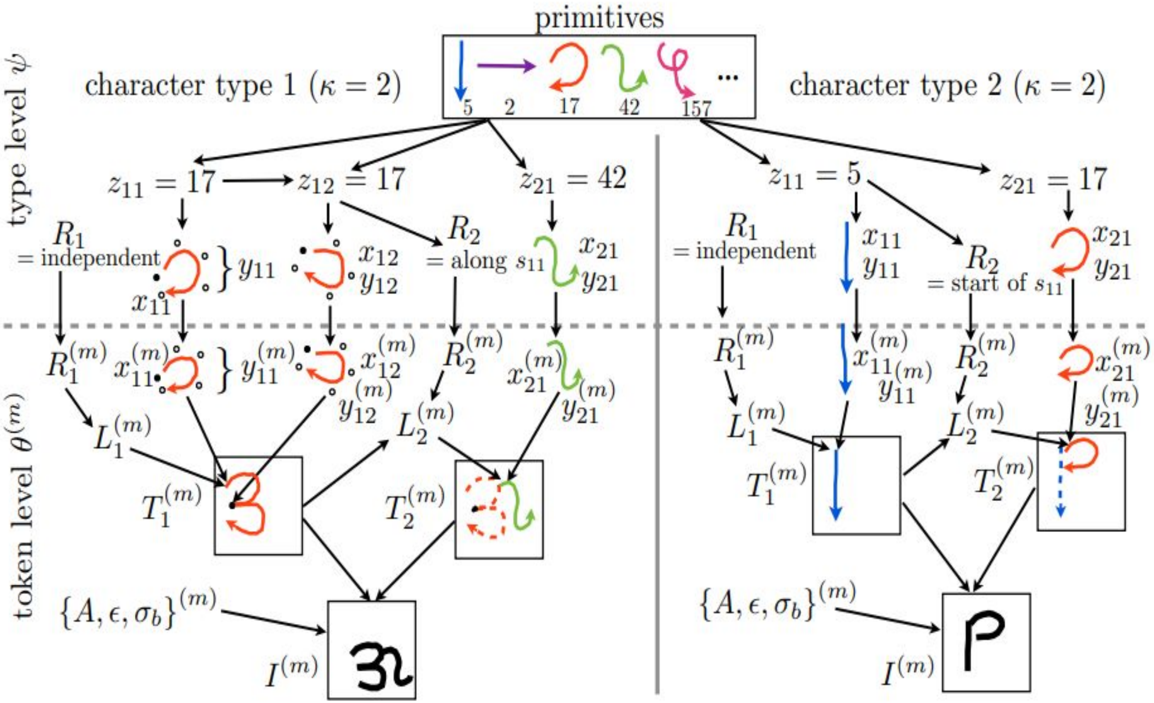
2) Generating character tokens:

Token-level relations must be exactly equal to their type-level counterparts,

$$P(R_i^{(m)}|R_i) = \delta(R_i^{(m)} - R_i).$$

except for the “along” relation which allows for token-level variability for the attachment along the spline

$$P(\tau_i^{(m)}|\tau_i) \propto N(\tau_i, \sigma_\tau^2)$$



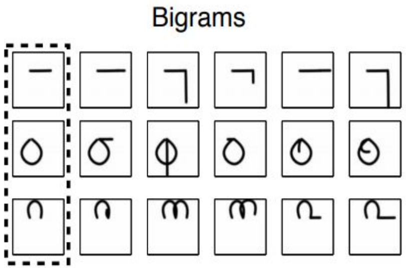
2) Generating character tokens:

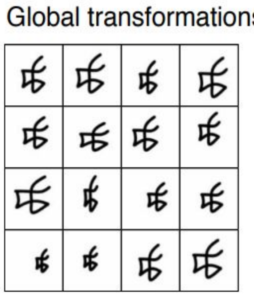
An image transformation $A^{(m)} \in R^4$ is sampled from $P(A^{(m)}) = N([1, 1, 0, 0], \Sigma_A)$, where the first two elements control a global re-scaling and the second two control a global translation of the center of mass of $T^{(m)}$

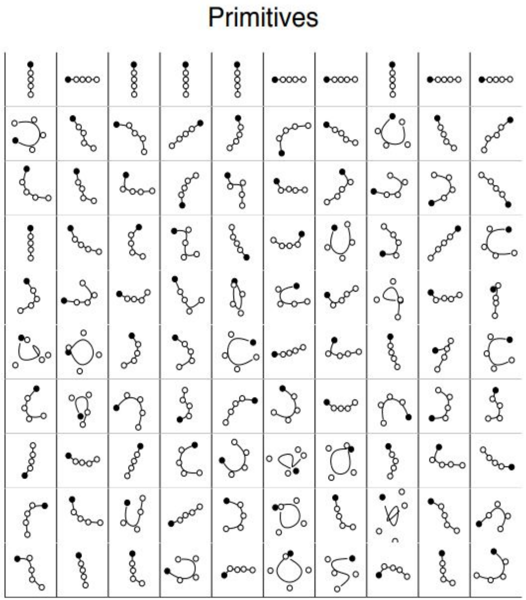
Blurring is accomplished through a convolution with a Gaussian filter with standard deviation $\sigma^{(m)}_b$.

The second noise process stochastically flips pixels with probability $\epsilon^{(m)}$

$$P(I^{(m)}|\theta^{(m)}) = P(I^{(m)}|T^{(m)}, A^{(m)}, \sigma_b^{(m)}, \epsilon^{(m)})$$







2) Generating character tokens:

procedure GENERATETOKEN(ψ)
  for $i = 1 \ldots \kappa$ **do**
    $R_i^{(m)} \leftarrow R_i$ ▷ Directly copy the type-level relation
    if $\xi_i^{(m)} =$ 'along' **then**
      $\tau_i^{(m)} \leftarrow P(\tau_i^{(m)} | \tau_i)$ ▷ Add variability to the attachment along the spline
    end if
    $L_i^{(m)} \leftarrow P(L_i^{(m)} | R_i^{(m)}, T_1^{(m)}, \ldots, T_{i-1}^{(m)})$ ▷ Sample stroke's starting location
    for $j = 1 \ldots n_i$ **do**
      $x_{ij}^{(m)} \leftarrow P(x_{ij}^{(m)} | x_{ij})$ ▷ Add variability to the control points
      $y_{ij}^{(m)} \leftarrow P(y_{ij}^{(m)} | y_{ij})$ ▷ Add variability to the sub-stroke scale
    end for
    $T_i^{(m)} \leftarrow f(L_i^{(m)}, x_i^{(m)}, y_i^{(m)})$ ▷ Compose a stroke's pen trajectory
  end for
  $A^{(m)} \leftarrow P(A^{(m)})$ ▷ Sample global image transformation
  $\epsilon^{(m)} \leftarrow P(\epsilon^{(m)})$ ▷ Sample the amount of pixel noise
  $\sigma_b^{(m)} \leftarrow P(\sigma_b^{(m)})$ ▷ Sample the amount blur
  $I^{(m)} \leftarrow P(I^{(m)} | T^{(m)}, A^{(m)}, \sigma_b^{(m)}, \epsilon^{(m)})$ ▷ Render and sample the binary image
  return $I^{(m)}$
end procedure

Results

- People, BPL and alternative models were compared side-by-side on five concept learning tasks that examine different forms of generalization from just a one or few examples.

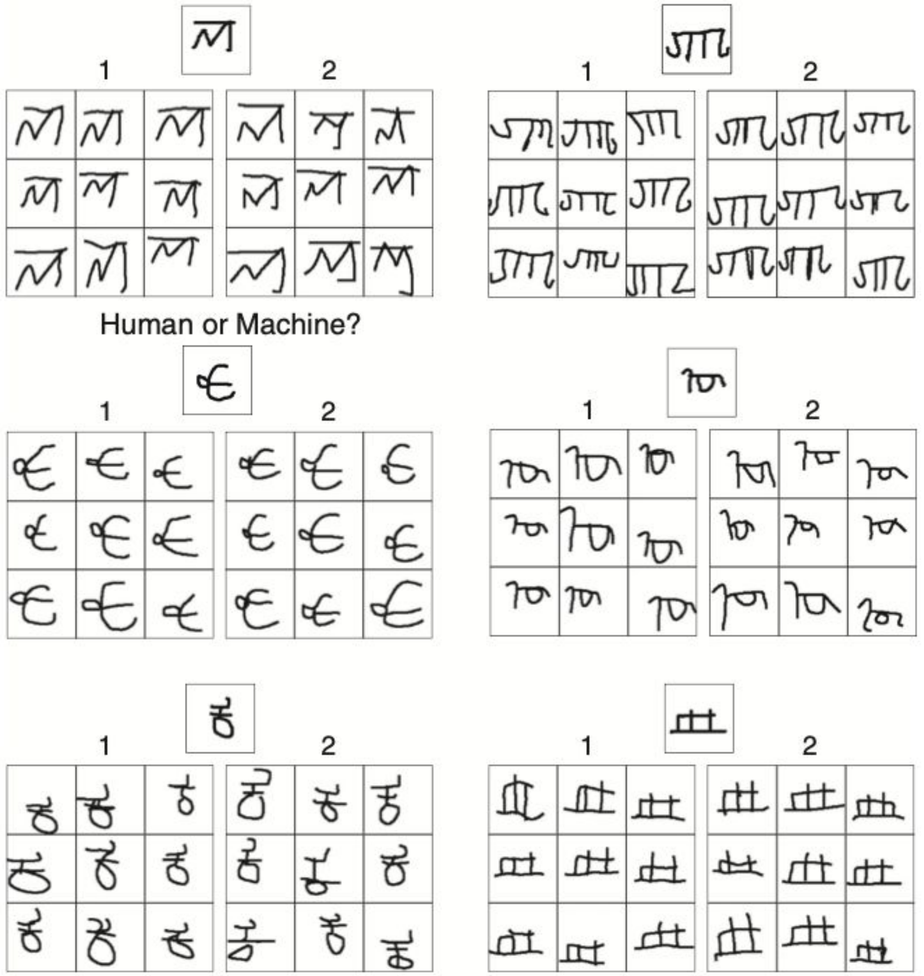
- All of these experiments were run on Amazon Mechanical Turk.

One such task during the same would be:

Results

Generating new exemplars of a given novel character (Fig. 5)

We'll come back to this later!



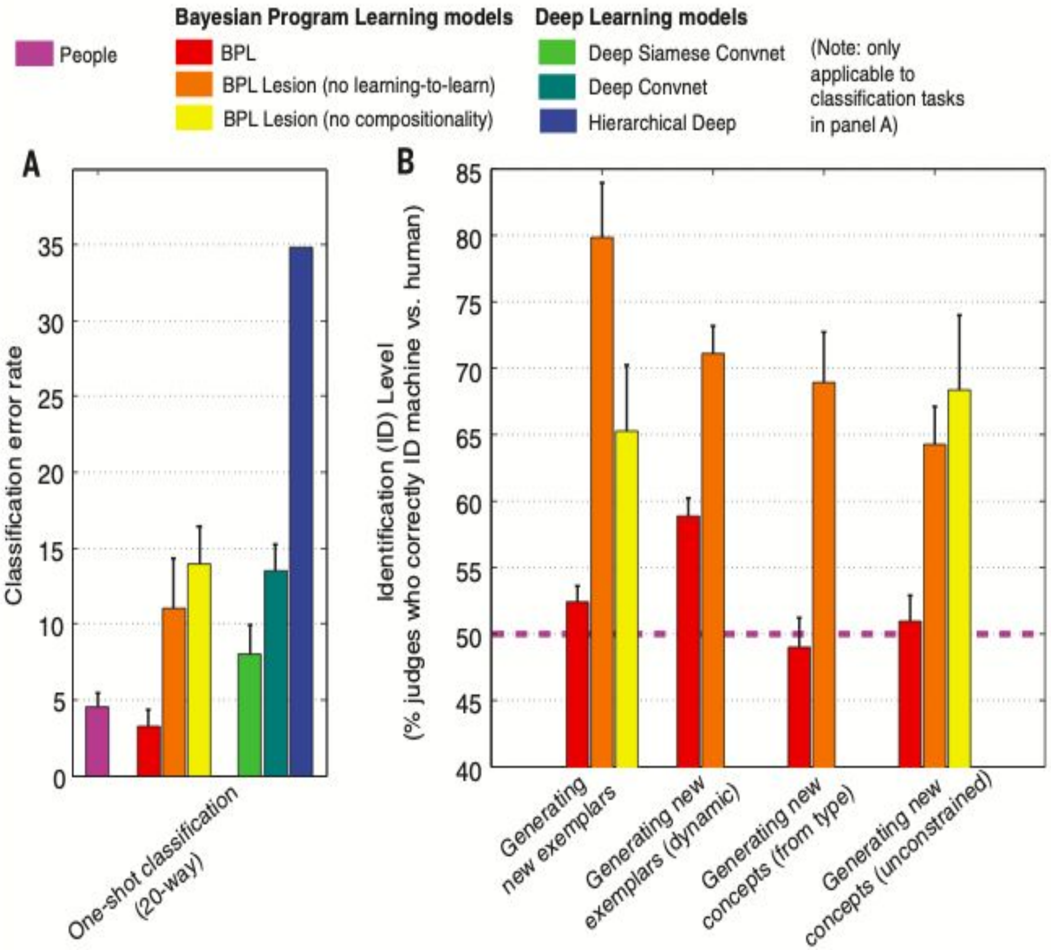
Results

- The main results are summarized in Fig. 6

- Human and machine performance was compared on (A) one-shot classification and (B) four generative tasks. The creative outputs for humans and models were compared by the percent of human judges correctly identify the machine.

- Ideal performance is 50%. This is because at this stage, the machine and humans are perfectly comparable.

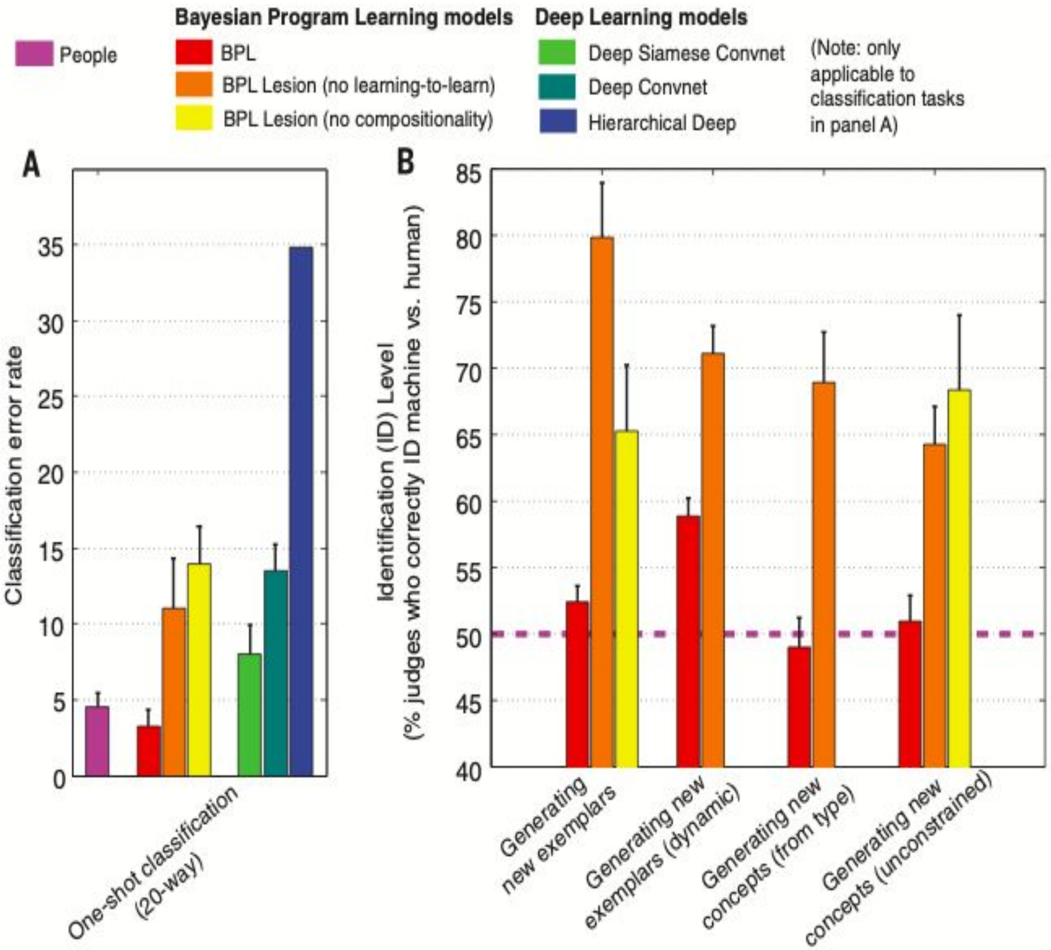
- SoTA what?



Results

- The no learning-to-learn lesion is applied at different levels (bars left to right): (A) token; and (B) token, stroke order, type, and type.

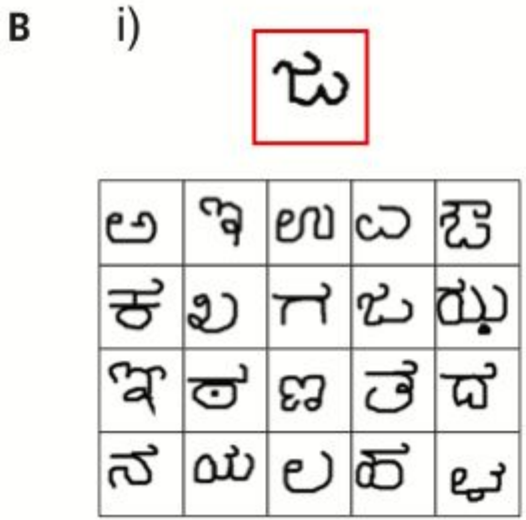
- The bars represent the standard error of mean.



Results

- One shot classification was evaluated through a series of within-alphabet classification tasks for 10 different alphabets.

- As seen in Fig 1B,i, a single image of a new character was presented and participants selected another example of that same character from a set of 20 distinct characters produced by a typical drawer of that alphabet.



Results

- As a baseline, the modified Hausdorff distance was computed between centered images, producing 38.8% errors.

- What is Hausdorff distance?

- Hausdorff distance helps to identify object similarity by extracting a set of edge points and calculating the distance for similarity purposes. As proposed by [M-P Dubuisson et al](), the paper introduces 24 different distance measures based on Hausdorff distance and between two point sets.
- Out of these 24, only D18 was considered to be a metric and all the others violated triangle law of inequality.
- In the end, after further observation using 4 operators (f1: min, f2: max, f3: midpoint, f4: weighted-average),it was found that only D10, D14, D18 and D22 were worth paying attention to.
- D18 cannot handle even a small amount of noise
- Although D10 and D14 handle outliers and occlusions, they remain same even in the presence of significant noise.
- Thus, D22 was considered the modified Hausdorff distance.

$$d_1(\mathcal{A},\mathcal{B}) = \min_{a\in\mathcal{A}} d(a,\mathcal{B}) \tag{1}$$

$$d_2(\mathcal{A},\mathcal{B}) = {}^{50}K^{th}_{a\in\mathcal{A}} d(a,\mathcal{B}) \tag{2}$$

$$d_3(\mathcal{A},\mathcal{B}) = {}^{75}K^{th}_{a\in\mathcal{A}} d(a,\mathcal{B}) \tag{3}$$

$$d_4(\mathcal{A},\mathcal{B}) = {}^{90}K^{th}_{a\in\mathcal{A}} d(a,\mathcal{B}) \tag{4}$$

$$d_5(\mathcal{A},\mathcal{B}) = \max_{a\in\mathcal{A}} d(a,\mathcal{B}) \tag{5}$$

$$d_6(\mathcal{A},\mathcal{B}) = \frac{1}{N_a}\sum_{a\in\mathcal{A}} d(a,\mathcal{B}) \tag{6}$$

where ${}^{x}K^{th}_{a\in\mathcal{A}}$ represents the K^{th} ranked distance such that $K/N_a = x\%$. For example, ${}^{50}K^{th}_{a\in\mathcal{A}}$ corresponds to the median of the distances $d(a,\mathcal{B}), \forall a \in \mathcal{A}$.

directed distance	function			
	f_1	f_2	f_3	f_4
d_1	D_1	D_2	D_3	D_4
d_2	D_5	D_6	D_7	D_8
d_3	D_9	D_{10}	D_{11}	D_{12}
d_4	D_{13}	D_{14}	D_{15}	D_{16}
d_5	D_{17}	D_{18}	D_{19}	D_{20}
d_6	D_{21}	D_{22}	D_{23}	D_{24}

Table 1: 24 distance measures between two point sets.

Results

- Fig (A) observations

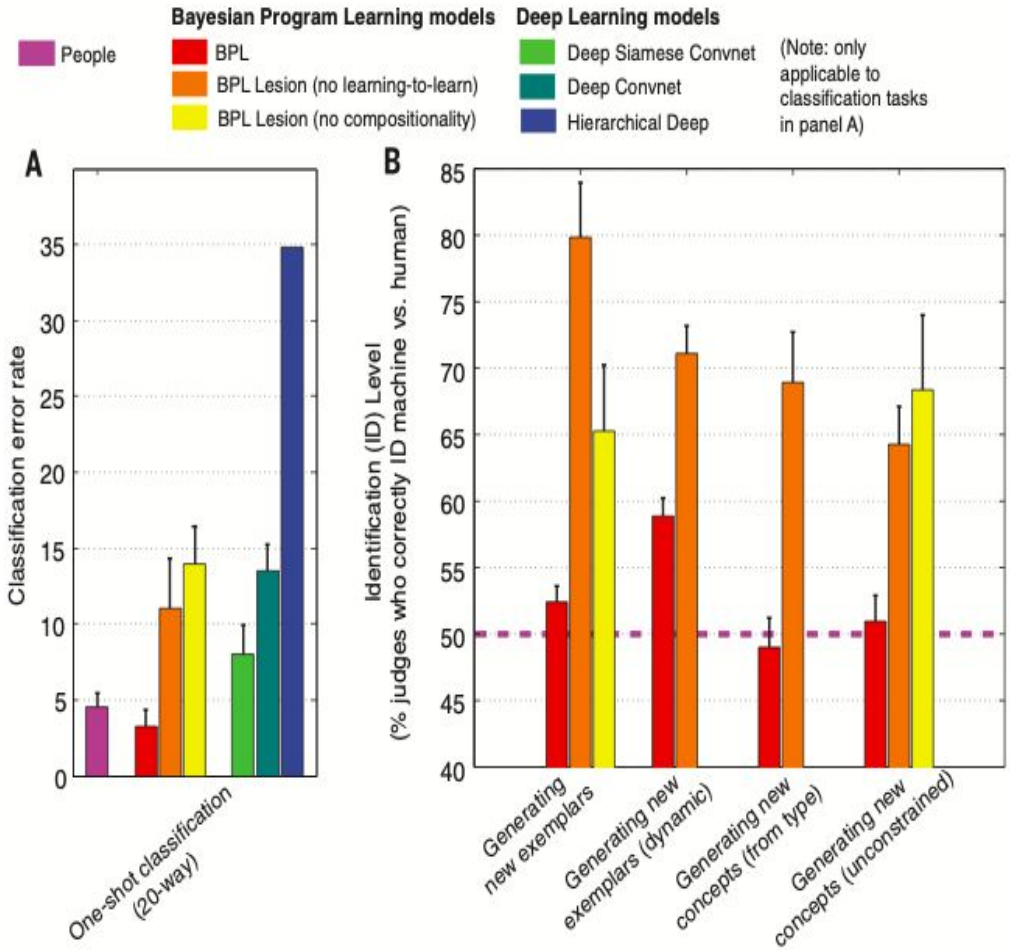
- **People were skilled one shot learners**, achieving an average error rate of 4.5% (N=40)

- BPL showed a similar results with a 3.3% error rate.

- A Deep ConvNet achieved around 13.5% error rate and a hierarchical deep model achieved 34.8% error.

- Deep Siamese network was optimized for one-shot learning tasks and still achieved almost a double error rate (8%) than the humans

- All of these are SoTA, however the goal here is to stay close to the error rate achieved by people and not classify better on the images.

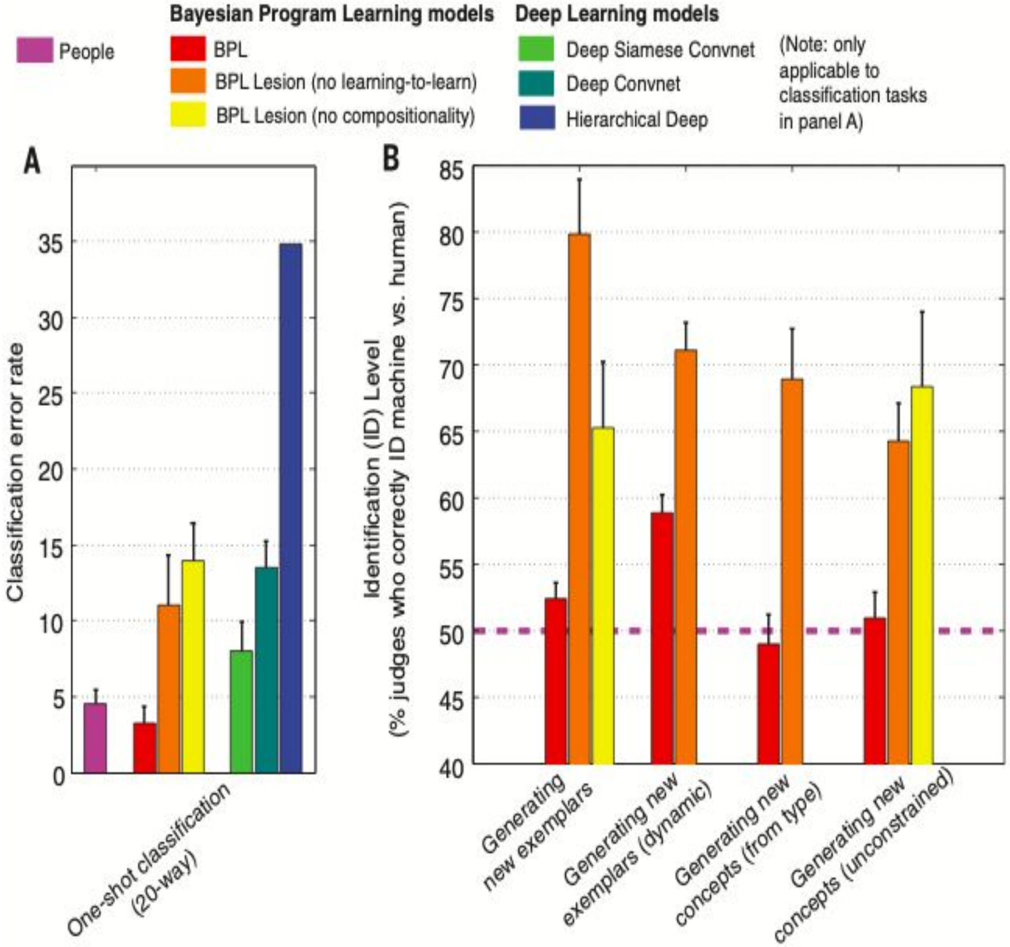


Results

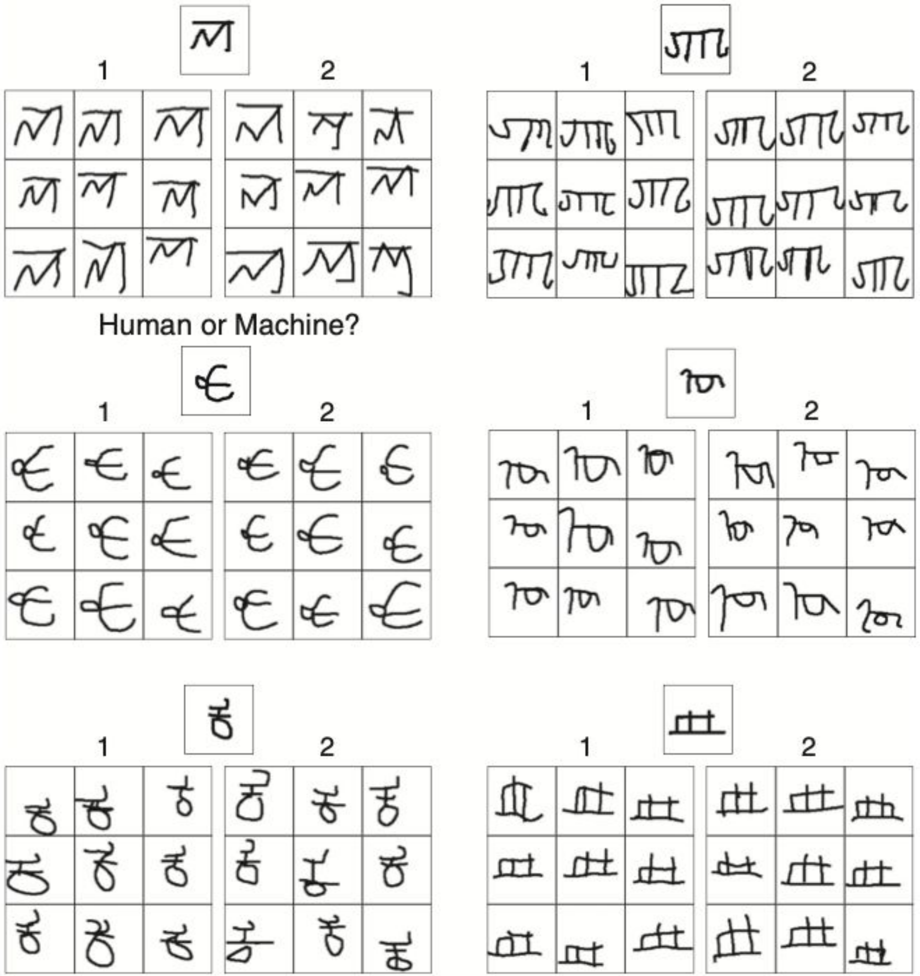
1. BPL’s advantage points to the benefits of modelling due to the underlying causal process in learning concepts, a different strategy than the compared deep learning concepts.

2. The other ingredients also adds up to the better performance.
    - These ingredients include: “learning to learn” and compositionality.

3. However when studied separately, these ingredients don’t perform so well



What is a Visual Turing Test?


1
2
1
2
Human or Machine?
1
2
1
2
1
2
1
2

Answers!

Row wise (Machine generated):

1.1: **1**

1.2: **2**

2.1: **2**

2.2: **1**

3.1: **1**

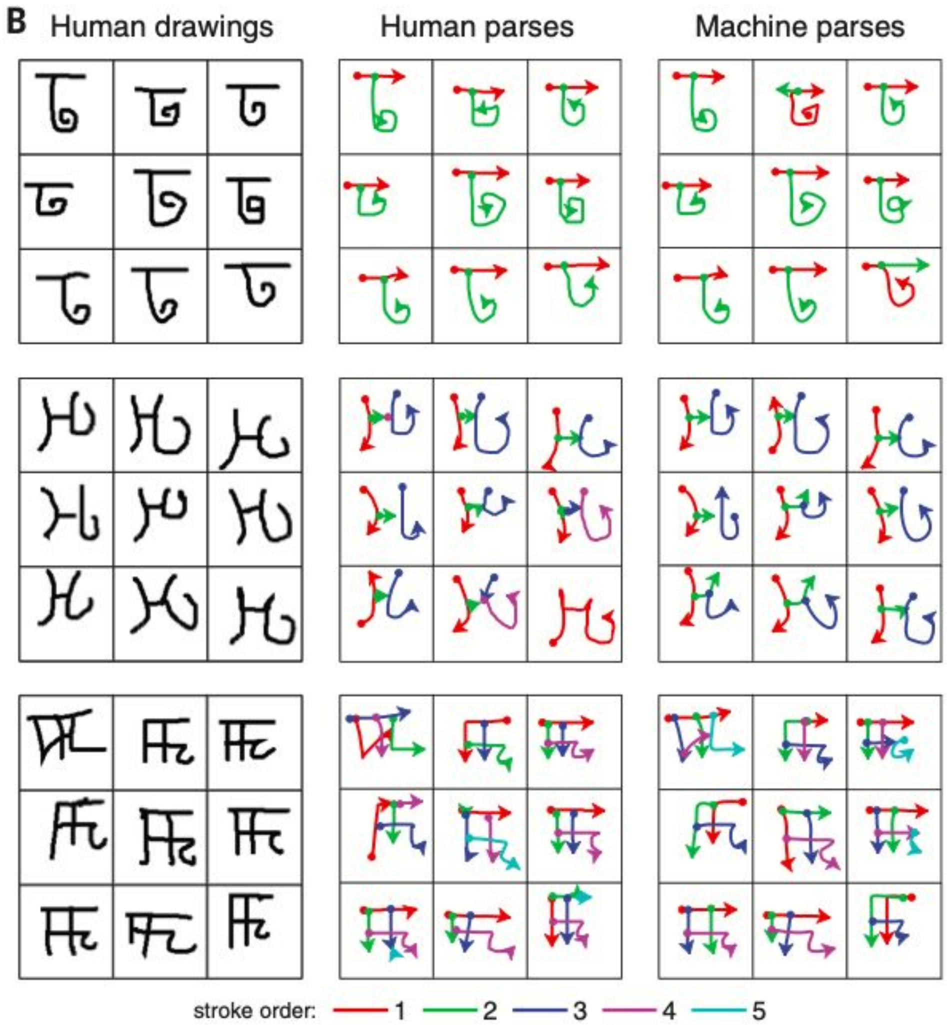
3.2: **1**

This is what a Visual Turing Test is!

- To evaluate parsing directly, movies of BPL and humans drawing were shown to a different set of judges.

- BPL performance on the Visual Turing test of this wasn't perfect (59% average ID level).

- Randomizing learned prior on stroke order and direction significantly raises ID level to 71% indicating the importance of the right causal dynamics for BPL



Results

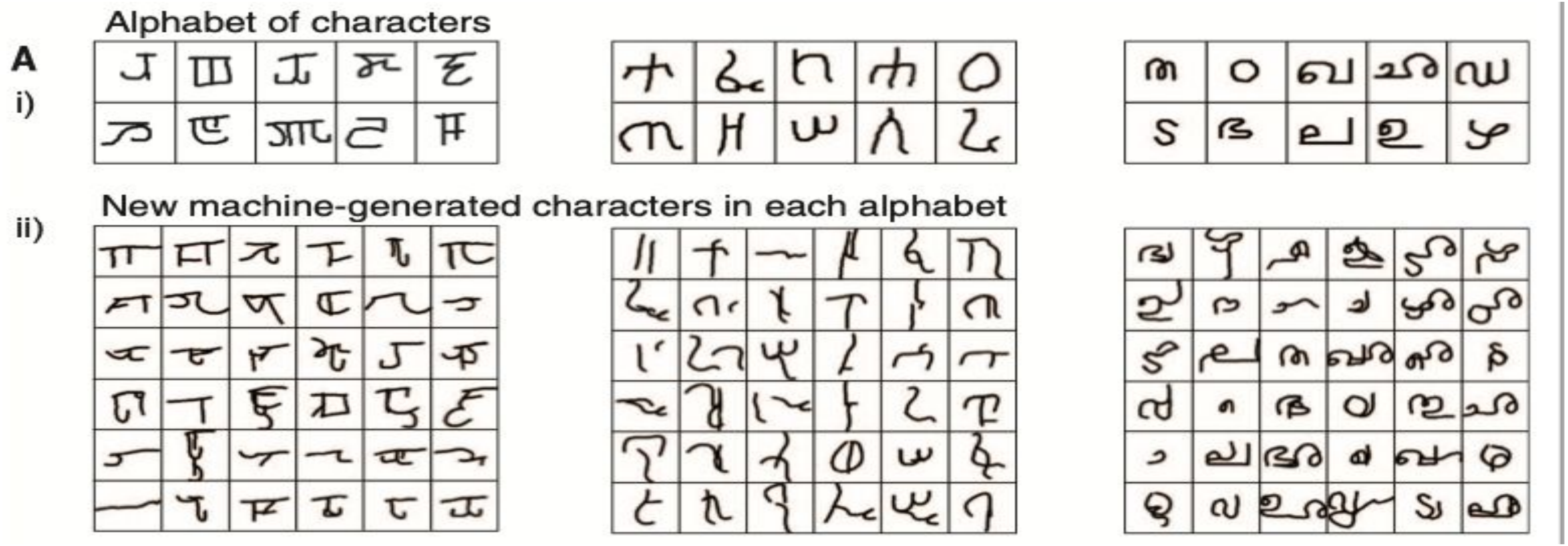
- Although learning to learn on 30 background alphabets proved more effective, humans would be constrained with factors such as less experience as in less familiarity with a few alphabets or the related drawing tasks.

- Thus, the models were retrained on on 2 different subsets of only 5 background alphabets.

- BPL still achieved similar performance (4.3% and 4%) while the Deep ConvNet got worse (24.0% and 23%)

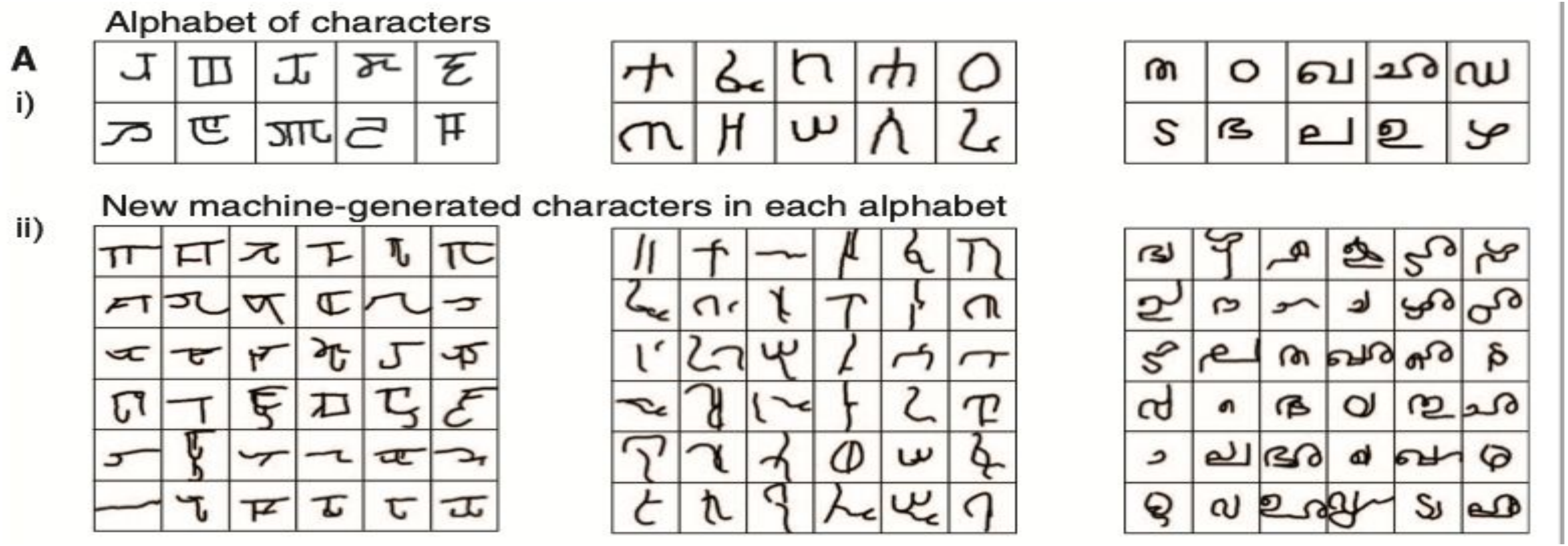
Note: This is where the performance should be near people (4.5%)

Results



- The human productive capacity goes beyond generating new examples of a given concept: People can also generate whole new concepts.

Results



This was tested by showing a few example characters from 1 of 10 foreign alphabets and asking participants to quickly create a new character that appears to belong to the same alphabet. The BPL model can capture this behavior by placing a nonparametric prior on the type level, which favors reusing strokes inferred from the example characters to produce stylistically consistent new characters

Concept Learning

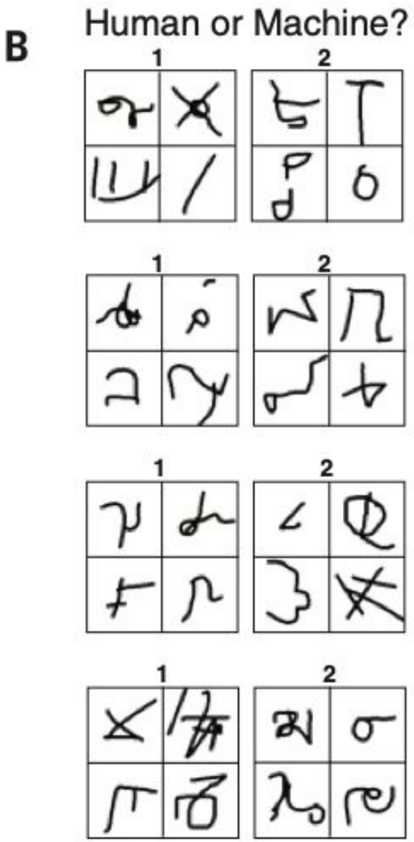
- Human judges compared people versus BPL in a visual Turing test (N = 117)

- The judges had only a 49% ID level on average which is not significantly different from chance.

- Individually, only 8 of 35 judges had an ID level significantly above chance.

- In contrast, a model with a lesion to (type-level) learning to learn was successfully detected by judges on 69% of trials in a separate condition of the visual Turing test, and was significantly easier to detect than BPL (18 of 25 judges above chance).

- Fig 7B shows the model generating alphabets without a reference alphabet



Results

– Further comparisons in section S6 suggested that the model's ability to produce plausible novel characters, rather than stylistic consistency per se, was the crucial factor for passing this test.

- Last, judges (N = 124) compared people and models on an entirely freeform task of generating novel character concepts, unconstrained by a particular alphabet. Sampling from the prior distribution on character types P(Ψ) in BPL led to an average ID level of 57% correct in a visual Turing test (11 of 32 judges above chance); with the nonparametric prior that reuses inferred parts from background characters.

- A lesion analysis revealed that both compositionality (68% and 15 of 22) and learning to learn (64% and 22 of 45) were crucial in passing this test.

* Note: (22 of 45) and (15 of 22) refer to the number of judges predicting above chance (50%)

Discussion

- Despite a changing artificial intelligence landscape, people remain far better than machines at learning new concepts: They require fewer examples and use their concepts in richer ways.

- This work suggests that the principles of **compositionality, causality,** and **learning to learn** will be critical in building machines that narrow this gap.

- The results show that this approach can perform one-shot learning in classification tasks at human-level accuracy and fool most judges in visual Turing tests of its more creative abilities.

- Since for each visual Turing test, fewer than 25% of judges performed significantly better than chance.

Discussion

- Although successful on these tasks, BPL still sees less structure in visual concepts than people do.

- It lacks explicit knowledge of parallel lines, symmetry, optional elements such as cross bars in “7”s, and connections between the ends of strokes and other strokes.

-Moreover, people use their concepts for other abilities that were not studied here, including **planning, explanation, communication** and **conceptual combination.**

- Probabilistic programs could capture these richer aspects of concept learning and use, but only with more abstract and complex structure than the programs studied here.

- More sophisticated programs could also be suitable for learning compositional, causal representations of many concepts beyond simple perceptual categories

Discussion

- Human cultures produce many such symbol systems, including gestures, dance moves, and the words of spoken and signed languages. As with characters, these concepts can be learned to some extent from one or a few examples, even before the symbolic meaning is clear

- (Examples)

- From this limited experience, people can typically recognize new examples and even produce a recognizable semblance of the concept themselves.

- The BPL principles of compositionality, causality, and learning to learn may help to explain how.

Discussion

- Although the work focuses on adult learners, it raises natural developmental questions. If children learning to write acquire an inductive bias similar to what BPL constructs, the model could help explain why children find some characters difficult and which teaching procedures are most effective.

- Comparing children’s parsing and generalization behavior at different stages of learning and BPL models given varying background experience could better evaluate the model’s learning-to-learn mechanisms and suggest improvements.

- By testing these classification tasks on infants who categorize visually before they begin drawing or scribbling, the authors can ask whether children learn to perceive characters more causally and compositionally based on their own proto-writing experience.

- Causal representations are prewired in the current BPL models, but they could conceivably be constructed through learning to learn at an even deeper level of model hierarchy

Discussion

- Recent large-scale brain models and deep recurrent neural networks have also focused on character recognition and production tasks—but typically learning from large training samples with many examples of each concept.

- It is seen that the one-shot learning capacities studied here as a challenge for these neural models: one that is expected that they might rise to by incorporating the principles of compositionality, causality, and learning to learn that BPL instantiates.

Bibliography

- Paper
- Supplementary Material
- Junlong Liu’s slides
- What is probabilistic programming?
- Dr. Adam M. Croom’s Youtube Video

THANKYOU !!!