Advanced machine learning
Bayesian program induction

Laura Capodicasa, Simon Lorent, Sarah Paquot, Maxime Tasset et Laurent Vanhee

Overview

I - Human-level concept learning through probabilistic program induction

II - Simulation as an engine of physical scene understanding

I - Human-level concept learning through probabilistic Induction

1st paper

Overview

- Goals
- Functions
- Implementation
- Tests and Results
- Discussion

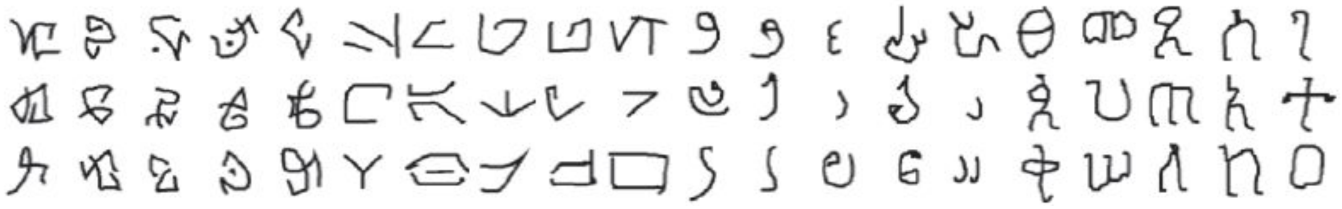
Goals

- Human Level Understanding of Concepts
- "One Shot" Learning
- Rich concepts from limited data

Use Case

We compare people, BPL, and other approaches on a set of five challenging concept learning tasks.

A very popular problem is to guess a letter read from a predefined alphabet.



In this case, the stroke of a pen has an impact on how the information is structured.

Omniglot dataset

1623 characters spanning 50 different alphabets

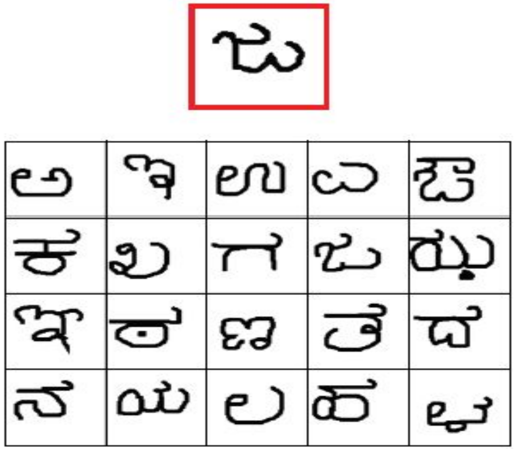
- Scrapped from omniglot.com
- Printed form
- Converted to handwritten using human participants

Resulting data set :

- (Image, Movie [x,y,t])
    - Movie : how the drawing was produced

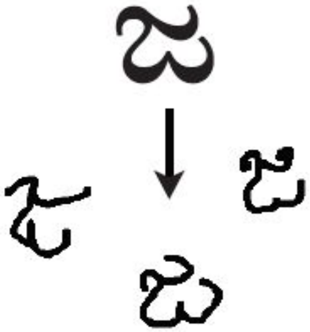
A wide range of functions

1. Classification of new examples



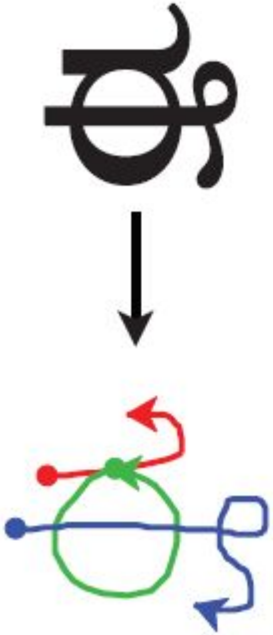
A wide range of functions (2)

2. Generation of new examples



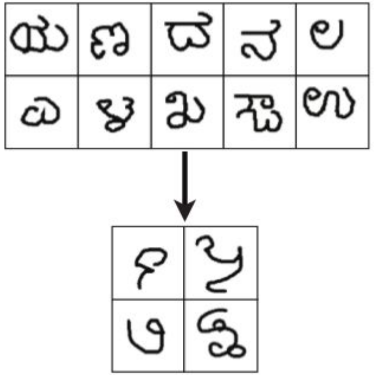
A wide range of functions (3)

3. Parsing an object into parts and relations



A wide range of functions (4)

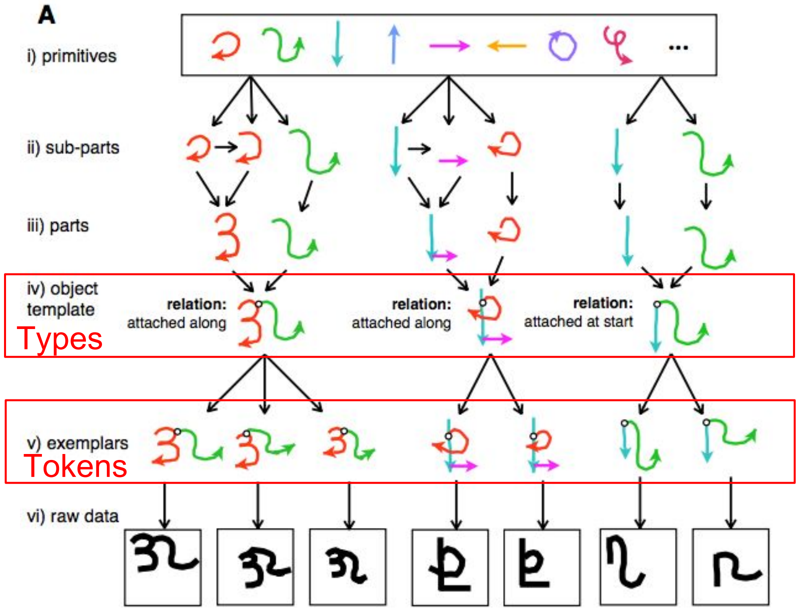
4. Generation of new concepts from related concepts.



Bayesian Program Learning (BPL) - Idea

1. Sample new <u>types</u> of concepts
2. Each new type is a generative model which produces new examples of the concept (<u>tokens</u>)
3. Renders the token-level variables in the format of the raw data.

=> BPL is a generative model for generative models.



BPL - Generating new types

Character type : ψ={κ,S,R}

- k : the number of strokes
- S = {S1,...,Sκ} the strokes
- R={R1,...,Rκ} relations between strokes

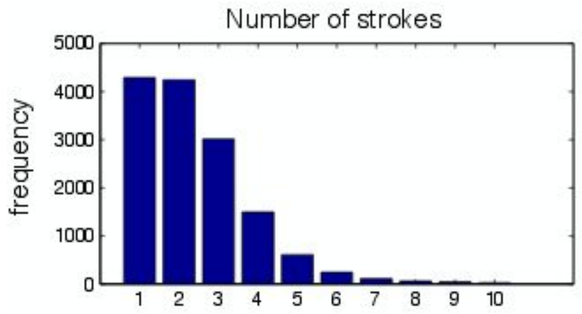
Joint distribution :

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

Strokes && Relations

Strokes : Pen up -> pen down

- Composed of sub-strokes {si1,...,sini} (simple movements separated by brief pauses)
- ni : number of substrokes
    - sampled from empirical frequency : P(ni|κ) (depends of κ)
    - characters with many strokes tend to have simpler strokes (ni decrease with k)

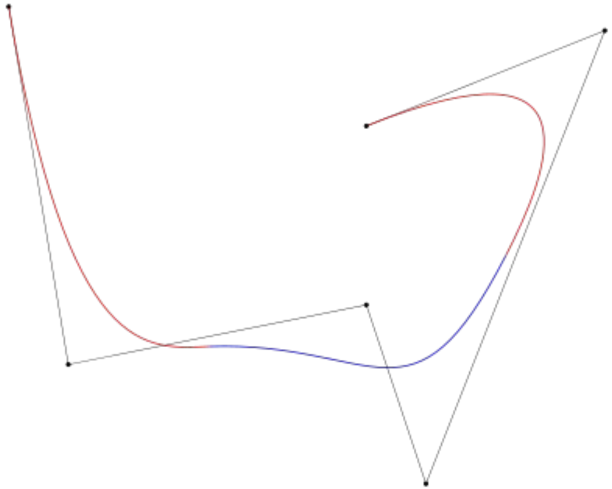


Relations : How the beginning of stroke Si connects to the previous strokes {S1,...,Si−1}

Algorithm to generate types

Each sub-stroke sij : uniform cubic b-spline

procedure GENERATETYPE
  $\kappa \leftarrow P(\kappa)$ ▷ Sample number of parts
  for i = 1 ... κ **do**
    $n_i \leftarrow P(n_i|\kappa)$ ▷ Sample number of sub-parts
    for j = 1 ... n_i **do**
      $s_{ij} \leftarrow P(s_{ij}|s_{i(j-1)})$ ▷ Sample sub-part sequence
    end for
    $R_i \leftarrow P(R_i|S_1, ..., S_{i-1})$ ▷ Sample relation
  end for
  $\psi \leftarrow \{\kappa, R, S\}$
  return @GENERATETOKEN(ψ) ▷ Return program



BPL - Generating character tokens

- motor noise is added to the control points and the scale of the subparts
- trajectory precise start location Li(m) is sampled from the schematic provided by its relation Ri to previous strokes
- global transformations are sampled
- binary image I(m) created by a stochastic rendering function

procedure GENERATETOKEN(ψ)
  for $i = 1...\kappa$ **do**
    $S_i^{(m)} \leftarrow P(S_i^{(m)}|S_i)$ ▷ Add motor variance
    $L_i^{(m)} \leftarrow P(L_i^{(m)}|R_i, T_1^{(m)}, ..., T_{i-1}^{(m)})$ ▷ Sample part's start location
    $T_i^{(m)} \leftarrow f(L_i^{(m)}, S_i^{(m)})$ ▷ Compose a part's trajectory
  end for
  $A^{(m)} \leftarrow P(A^{(m)})$ ▷ Sample affine transform
  $I^{(m)} \leftarrow P(I^{(m)}|T^{(m)}, A^{(m)})$ ▷ Sample image
  return $I^{(m)}$

Keys ideas

- Compositionality
- Causality
- Learning to learn

BPL - Compositionality (1)

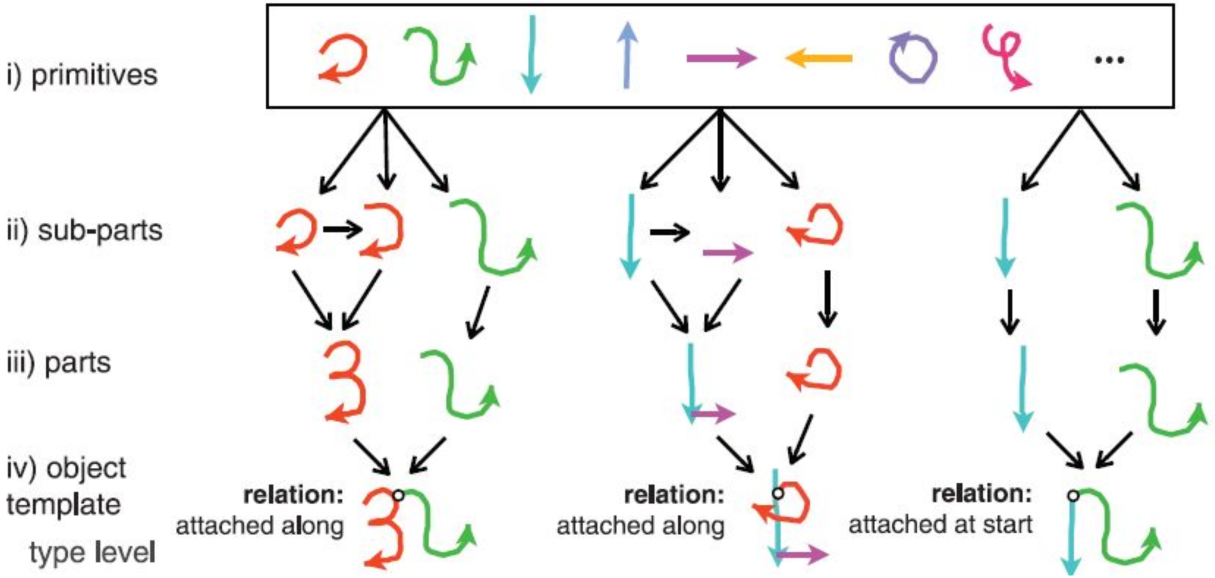
Rich concepts built compositionally from simpler primitives.

Concepts are built from :

- Parts
- Subparts
- Spatial Relations
- primitives

Idea: define a generative model that sample new types of concepts combining parts.

BPL - Compositionality (2)



Type

BPL - Compositionality (3)

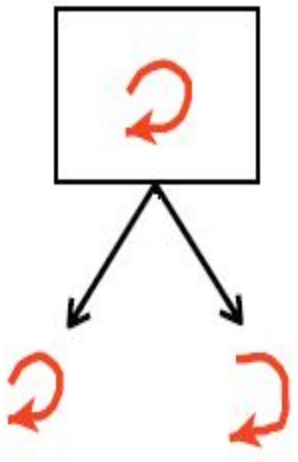
In this example :

- Primitives are simple pen strokes. (b-spline)
- Sub-parts are built based on the shape of the associated primitive.
- Parts are a concatenation of sub-parts where the person writing does not lift the pen
- Object Templates are built from parts that are associated to each other via relations.

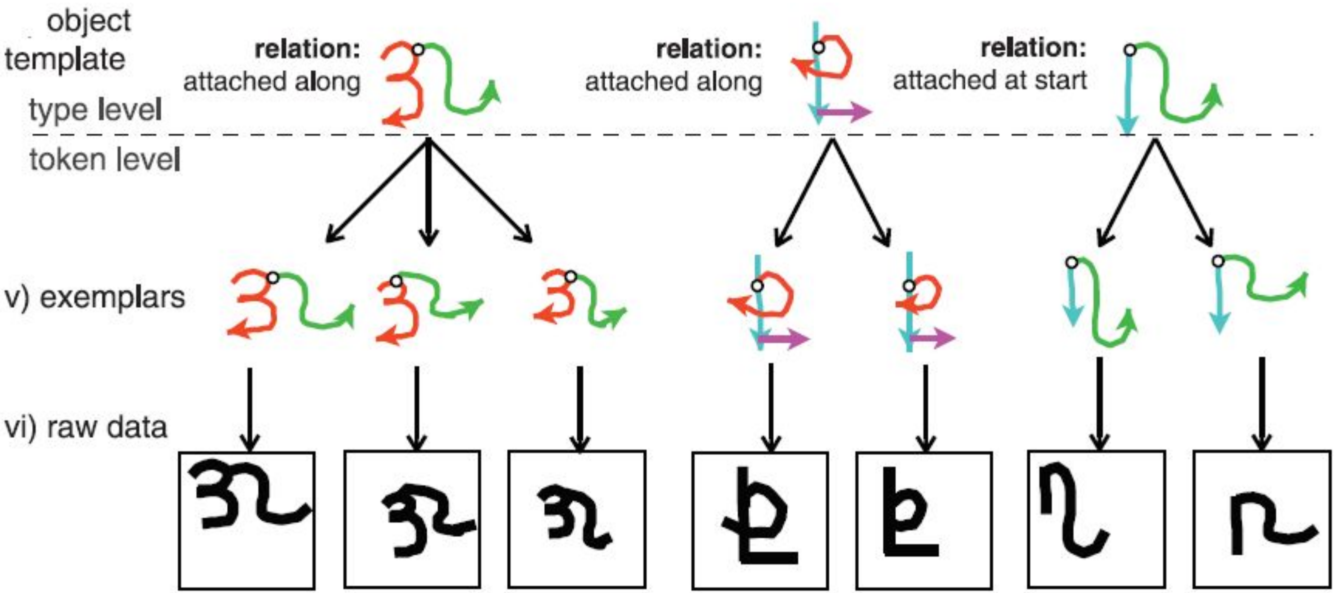
BPL - Causality (1)

Primitives, parts, and relations follow a causal structure. In a sense, the Object template is compatible with noise. This allows the computer to distinguish between examples of the same objects.

The idea is to generate examples of our Object template while simulating real world properties.



BPL - Causality (2)



Token

Binary
image

BPL - Learning to Learn (1)

The construction of programs that best explains the observation under a Bayesian Criterion.

The development of hierarchical priors allows previous experience with related concepts to ease learning of new concepts.

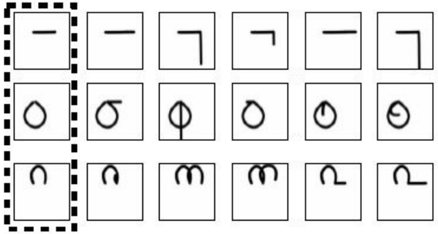
Multiple methods that generate hyperparameters.

BPL - Learning to Learn (2)

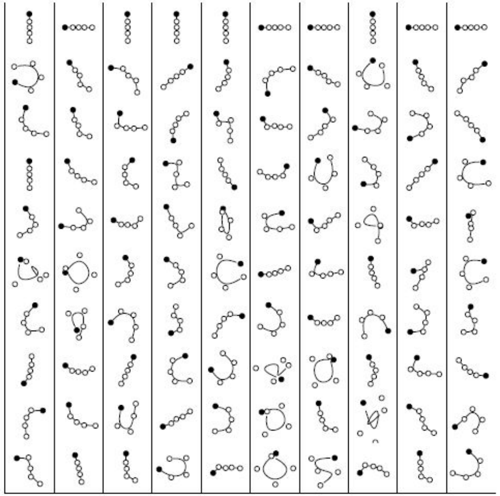
Methods :

- Learning Primitives

Bigrams



Primitives

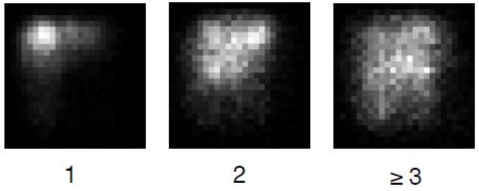


BPL - Learning to Learn (2)

Methods :

- Learning Primitives
- Learning Start Positions

Start position for strokes in each position



BPL - Learning to Learn (2)

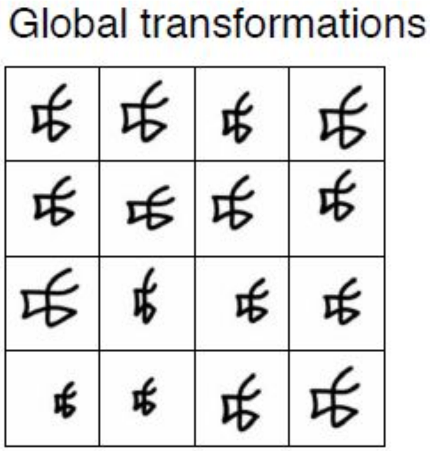
Methods :

- Learning Primitives
- Learning Start Positions
- Learning Relations and Token variability

BPL - Learning to Learn (2)

Methods :

- Learning Primitives
- Learning Start Positions
- Learning Relations and Token variability
- Learning Image Parameters



BPL - Learning to Learn (3)

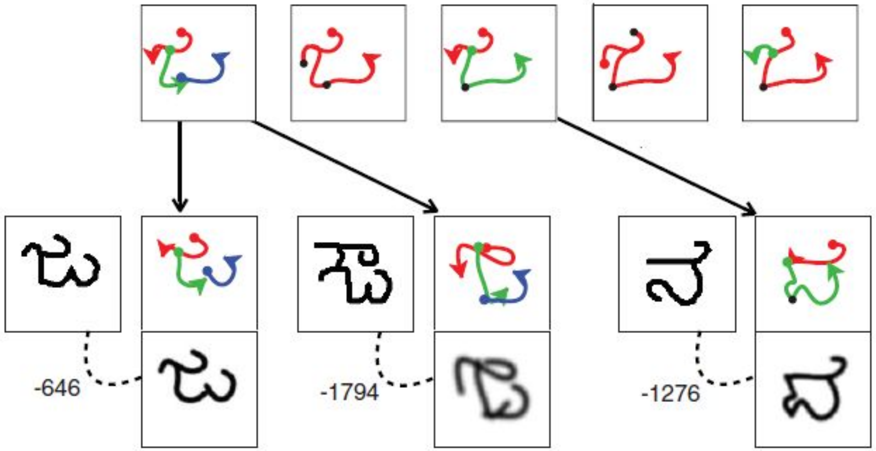
The bayesian Criterion allows for the generation of Object template from image data (learning)

BPL - Learning to Learn (4)

Allows for testing Image data with existing Object Templates.

Bayesian Criterion fits the best template.

It is followed by a reconstruction



BPL - Mathematics

We define

- ψ is a type which correspond to the object template. Ex: 
- θ is a token. There's M tokens, there are of type ψ. Ex: 
- I is the binary image. It correspond to a given token. Ex: 

And :

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

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

Generative process for types
Generative process for tokens
Image model

BPL - Mathematics (2) - example use formula

Using the following formula for spam detection

$$P(\Psi, \Theta^{(1)}, \ldots, \Theta^{(M)}, I^{(1)}, \ldots, I^{(M)}) = P(\Psi) \prod_{m=1}^{M} P(I^{(m)}|\Theta^{(m)}) P(\Theta^{(m)}|\Psi)$$

In this case: Ψ = 1 if the message is a spam, 0 otherwise.
Θ_i = 1 if the word i is in the message, 0 otherwise.

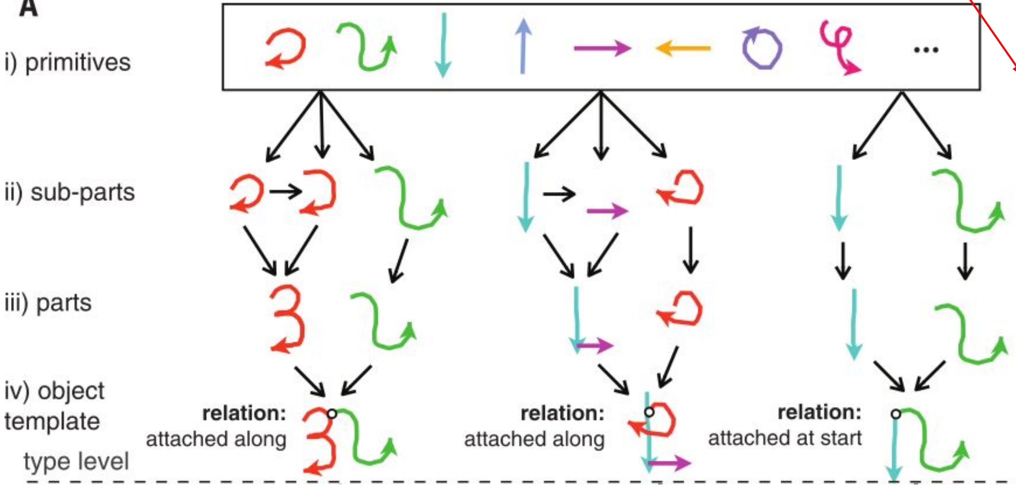
Join probability is the probability of the two events happening together. So here we have :

P(Spam, $\Theta^{(1)}, \ldots, \Theta^{(M)}$) = P(Spam) P($\Theta^{(1)}$ | Spam) * … * P($\Theta^{(M)}$ | Spam)

BPL - Mathematics (3)

$$P(\Psi, \Theta^{(1)}, \ldots, \Theta^{(M)}, I^{(1)}, \ldots, I^{(M)}) = \boxed{P(\Psi)} \prod_{m=1}^{M} P(I^{(m)}|\Theta^{(m)}) P(\Theta^{(m)}|\Psi)$$

A

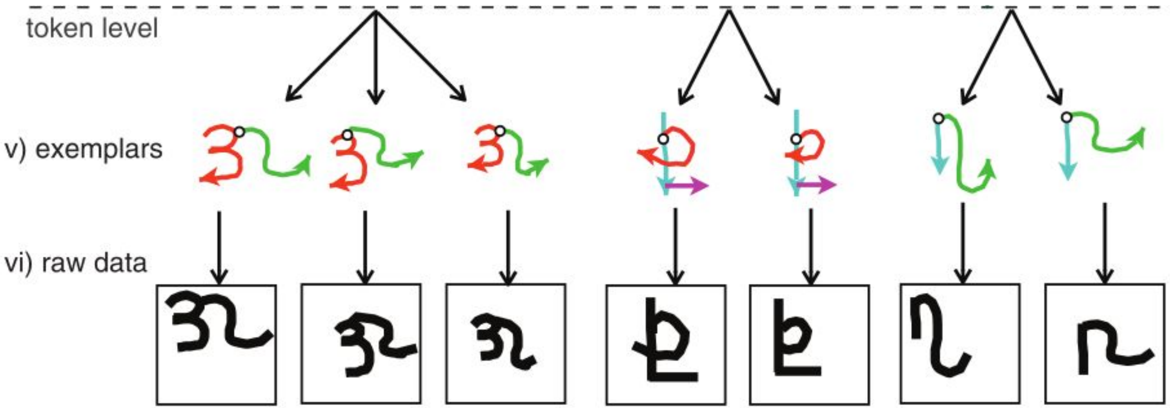


B

procedure GENERATETYPE
  $\kappa \leftarrow P(\kappa)$ ▷ Sample number of parts
  for i = 1 ... κ **do**
    $n_i \leftarrow P(n_i|\kappa)$ ▷ Sample number of sub-parts
    for j = 1 ... n_i **do**
      $s_{ij} \leftarrow P(s_{ij}|s_{i(j-1)})$ ▷ Sample sub-part sequence
    end for
    $R_i \leftarrow P(R_i|S_1, ..., S_{i-1})$ ▷ Sample relation
  end for
  $\psi \leftarrow \{\kappa, R, S\}$
  return @GENERATETOKEN(ψ) ▷ Return program

BPL - Mathematics (4)

$$P(\Psi, \Theta^{(1)}, \ldots, \Theta^{(M)}, I^{(1)}, \ldots, I^{(M)}) = P(\Psi) \prod_{m=1}^{M} \boxed{P(I^{(m)}|\Theta^{(m)})}\boxed{P(\Theta^{(m)}|\Psi)}$$



procedure GENERATETOKEN(ψ)
  for $i = 1 \ldots \kappa$ **do**
    $S_i^{(m)} \leftarrow P(S_i^{(m)}|S_i)$ ▷ Add motor variance
    $L_i^{(m)} \leftarrow P(L_i^{(m)}|R_i, T_1^{(m)}, \ldots, T_{i-1}^{(m)})$ ▷ Sample part's start location
    $\boxed{T_i^{(m)} \leftarrow f(L_i^{(m)}, S_i^{(m)})}$ ▷ Compose a part's trajectory
  end for
  $A^{(m)} \leftarrow P(A^{(m)})$ ▷ Sample affine transform
  $\boxed{I^{(m)} \leftarrow P(I^{(m)}|T^{(m)}, A^{(m)})}$ ▷ Sample image
  return $I^{(m)}$

Testing & Results

5 Learning Tasks:

- One-Shot Classification
- Generating Examples (4 Methods)

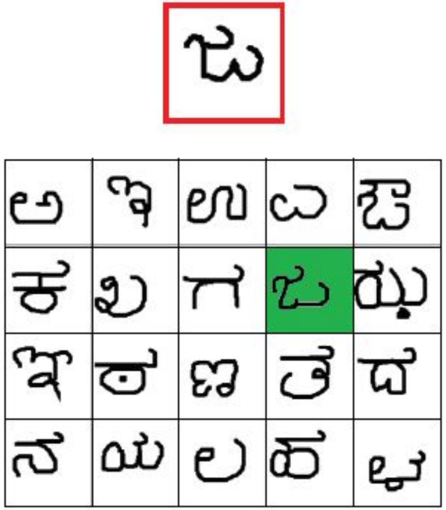
Testing & Results - One Shot Classification (1)

Within alphabet classification tasks (10 different alphabets with 20 characters).

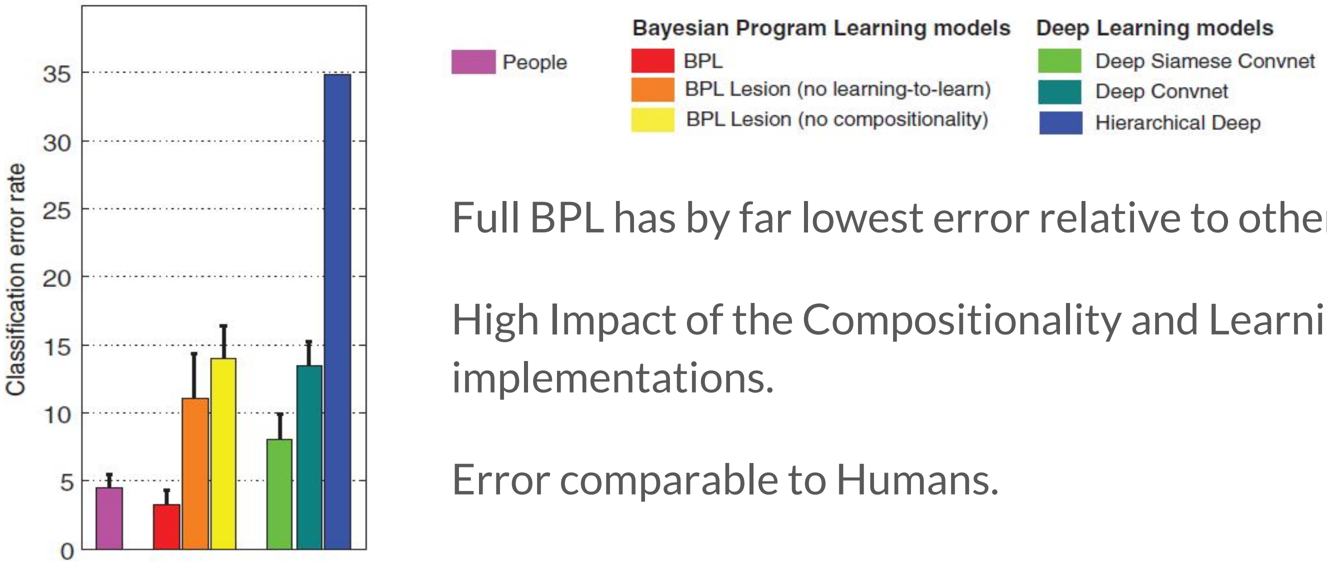
Turing test between BPL, humans, and alternative models.

Participant given a character.

Participant guesses closest match.



Testing & Results - One Shot Classification (2)



Full BPL has by far lowest error relative to other models.

High Impact of the Compositionality and Learning to learn implementations.

Error comparable to Humans.

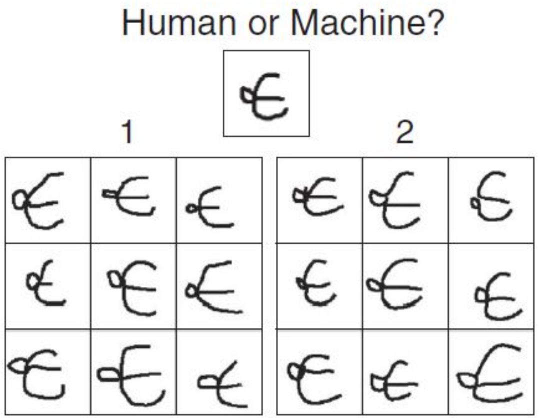
Testing & Results - Generating Examples (1)

In the creative output production tasks, we compare paired examples of human and machine behavior.

Judges are given human generated characters and respective BPL generated characters.

Judges try to guess the human's examples over the machine's examples.

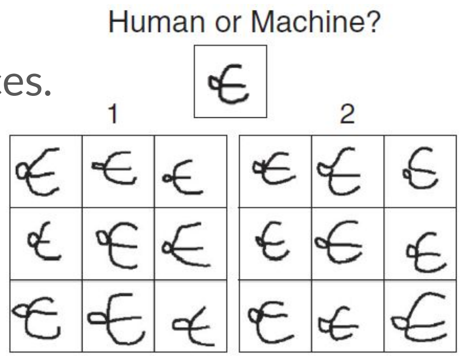
4 Different Tests.



Testing & Results - Generating Examples (2)

- Generating new Exemplars :

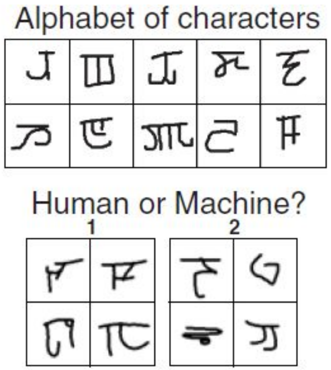
  From a concept, BPL and humans produce 9 different instances.
  Judges given pairs generated by humans and the BPL.



- Generating new Exemplars (Dynamic) :

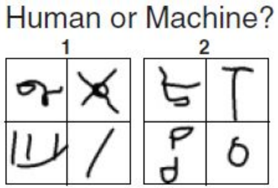
  From a concept, BPL and humans produce 9 different instances.
  Judges given movies of generated instances. Tests the dynamic aspects of writing characters.

Testing & Results - Generating Examples (3)



- Generating new Concepts (from type) :

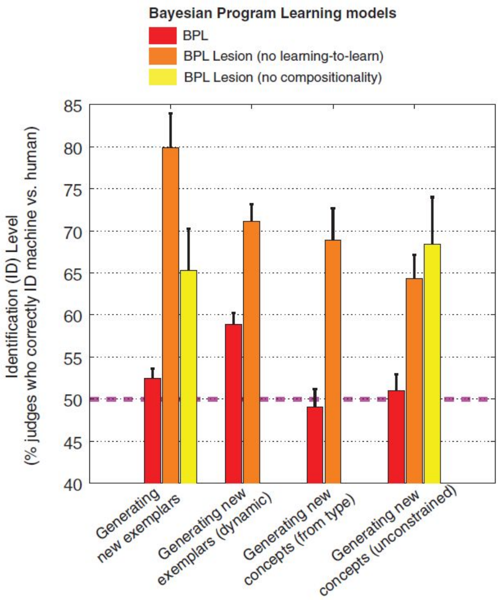
  Show people/BPL 10 characters from an Alphabet.
  They quickly create new characters similar to the Alphabet.
  Judges guess which characters were generated by a human.



- Generating new Concepts (unconstrained):

  No alphabet constraints on people/BPL.
  Judges guess which characters were generated by a human.

Testing & Results - Generating Examples (4)



Full BPL has near perfect results.

High Impact of the Compositionality and Learning to learn implementations.

50% ID level => Judges decide correctly 50% of time .

Discussion (1)

Issues :

- BPL see les structure than humans. In the example : parallel lines, symmetry, and etc.. not represented.
- BPL is less general than humans when it comes using these concepts for other abilities.
- Causality models not easy to implement.

Discussion (2)

Advantages :

- Can be applied to the domain of speech. Construction of words (by sounds)
- Study can be tested on children to understand their way of thinking.
- Compositionality and Learning to Learn models are pretty straightforward.
- Can shed light on the neural representation of objects.

II - Simulation as an engine of physical scene understanding

2nd paper

Overview : Intuitive physics engine (IPE)

- Goals
- Model
- Model illustration
- Computational theory
- Experiments

Goals

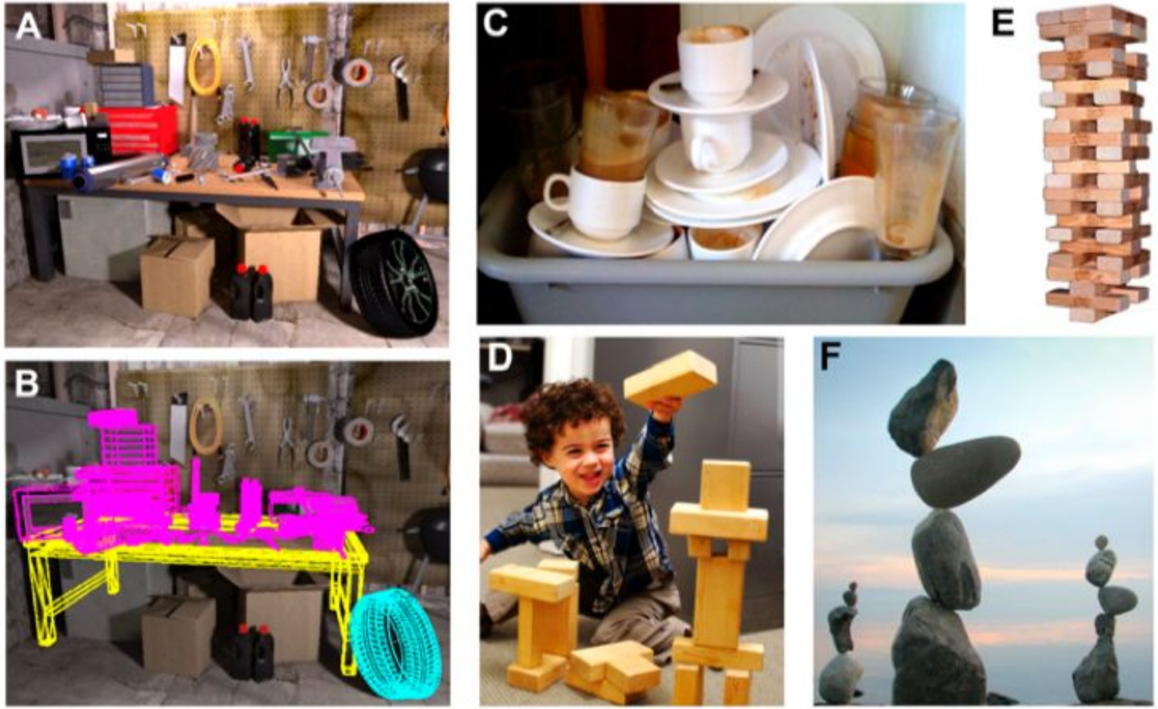
- Understand how human brain works
- Make quick predictions from incomplete informations
- See if people will adopt the cheapest approximations possible

Model

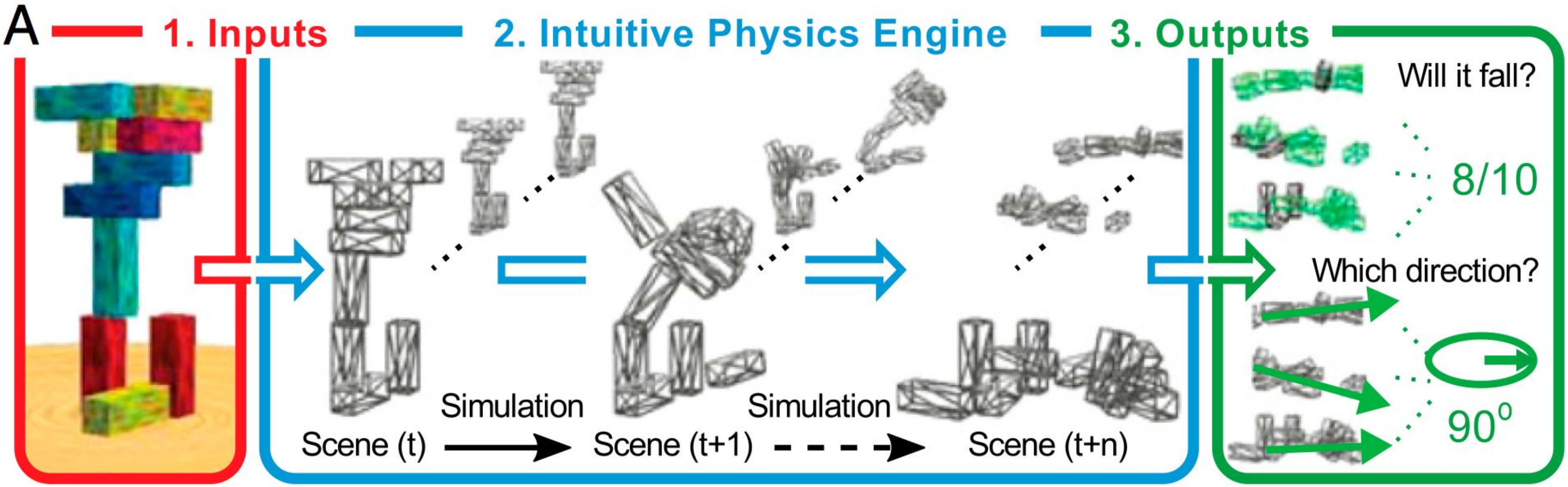
In order to model the physic, a physic engine is used. Parameters are 3 randoms variables.

- σ : captures uncertainty about scene representation
- ϕ : uncertainty about latent force
- μ : uncertainty about physic property

Examples



Model illustration



Computational theory: Definition

- S_t : Scene state at time, t.
- $S_{t_0:t_1} = (S_{t_0}, S_{t_0+1}, \ldots, S_{t_1-1}, S_{t_1})$: Sequence of scene states from time t_0 to t_1.
- f_t : Extrinsic force applied beginning at time t.
- $f_{t_0:t_1}$: Sequence of extrinsic forces applied from time t_0 to t_1.
- I_{S_t} : Observed information about S_t.
- I_f : Observed information about $f_{t_0:t_1}$.
- $\psi(\cdot)$: Deterministic physical dynamics from t_0 to t_1, which maps S_{t_0} to a new state at time S_{t_1} : $S_{t_1} = \psi(S_{t_0}, f_{t_0}, t_1 - t_0)$. The force, f_{t_0}, is applied for a duration $t_1 - t_0$. The dynamics can be applied recursively,

$$S_{t_2} = \psi\left(\psi\left(S_{t_0}, f_{t_0}, t_1 - t_0\right), f_{t_1}, t_2 - t_1\right).$$

Computational theory: Definition (2)

- We denote the repeated application of $\psi(\cdot)$ from $(t_0 : t_n)$ as $\Psi(\cdot)$,

$$
\begin{aligned}
S_{t_n} &= \psi\big(\ldots\psi\big(S_{t_0}, f_{t_0}, t_1 - t_0\big), \ldots, f_{t_{n-1}}, t_n - t_{n-1}\big) \\
&= \Psi\big(S_{t_0}, f_{t_0:t_{n-1}}, t_0 : t_n\big).
\end{aligned}
$$

$\mathcal{Q}_q(\cdot)$: Output predicate corresponding to a query, q, which maps an initial state and a future sequence of scene states to a judgmer $J_q = \mathcal{Q}_q(S_{0:T})$

In the experiments, the queries were sensitive only to the initial and final scene states (i.e., for Will the tower fall?, the query reflected how many blocks dropped from t=0 to t =T),so $J_q = \mathcal{Q}_q(S_0, S_T)$

Some precisions

$\psi(\cdot)$ is deterministic

$$\Rightarrow \Pr(S_{t+1} | S_t, f_t) = 1, \text{ for } S_{t+1} = \psi(S_t, f_t, 1)$$
$$0 \text{ for any other value of } S_{t+1}$$

Then, where comes the uncertainty from?

- We have I_{S_0}, not S_0
- We have I_f , not $f_{0:T-1}$

Proof

$$
\begin{aligned}
P(S_{0:T}|I_{S_0}, I_f) &= \int_{f_{0:T-1}} P(S_{0:T}, f_{0:T-1}|I_{S_0}, I_f)\, df_{0:T-1} \\
&= \int_{f_{0:T-1}} P(S_0|I_{S_0}, {\color{red}I_f}) \times P(S_{1:T}, f_{0:T-1}|S_0, {\color{red}I_{S_0}}, I_f)\, df_{0:T-1} \\
&= \int_{f_{0:T-1}} P(S_0|I_{S_0}) \times P(f_{0:T-1}|S_0, I_f) \times P(S_{1:T}|S_0, {\color{red}I_f}, f_{0:T-1})\, df_{0:T-1} \\
&= \int_{f_{0:T-1}} P(S_0|I_{S_0}) \times P(f_{0:T-1}|S_0, I_f) \times P(S_1|S_0, f_{0:T-1}) \times P(S_{2:T}|{\color{red}S_0}, S_1, f_{0:T-1})\, df_{0:T-1} \\
&= \int_{f_{0:T-1}} P(S_0|I_{S_0}) \times P(f_{0:T-1}|S_0, I_f) \times P(S_1|S_0, f_0) \ldots P(S_T|S_{T-1}, f_{T-1})\, df_{0:T-1}
\end{aligned}
$$

$$\Pr\left(S_{0:T}|I_{S_0},I_f\right) = \int_{f_{0:T-1}} \Pr(S_T|S_{T-1},f_{T-1})\cdots\Pr(S_1|S_0,f_0)$$

$$\Pr(S_0|I_{S_0})\Pr\left(f_{0:T-1}|I_f\right)\mathrm{d}f_{0:T-1}$$

$$= \int_{f_{0:T-1}} \Pr(\psi(S_{T-1},f_{T-1},1)|S_{T-1},f_{T-1})\cdots \qquad \textbf{[S1]}$$

$$\Pr(\psi(S_0,f_0,1)|S_0,f_0)\Pr(S_0|I_{S_0})\cdots$$

$$\Pr\left(f_{0:T-1}|I_f\right)\mathrm{d}f_{0:T-1}.$$

$$\Pr\big(S_T, S_0 | I_{S_0}, I_f\big) = \int\limits_{f_{0:T-1}} \int\limits_{S_{1:T-1}} \Pr(S_T | S_{T-1}, f_{T-1}) \ldots \Pr(S_1 | S_0, f_0)$$

$$\Pr(S_0 | I_{S_0}) \Pr\big(f_{0:T-1} | I_f\big) \mathrm{d}S_{1:T-1} \mathrm{d}f_{0:T-1}$$

$$= \int\limits_{f_{0:T-1}} \Pr(S_T | S_0, f_{0:T-1}) \Pr(S_0 | I_{S_0}) \Pr\big(f_{0:T-1} | I_f\big) \mathrm{d}f_{0:T-1}$$

$$= \int\limits_{f_{0:T-1}} \Pr(\Psi(S_{t_0}, f_{0:T-1}, 0:T) | S_0, f_{0:T-1})$$

$$\Pr(S_0 | I_{S_0}) \Pr\big(f_{0:T-1} | I_f\big) \mathrm{d}f_{0:T-1}.$$

Note that because $\psi(\cdot)$ can be applied recursively,

$$\Pr(S_T | S_0, f_{0:T-1}) = \Pr(\Psi(S_0, f_{0:T-1}, 0:T) | S_0, f_{0:T-1}).$$

Outputs

$$J_q = \mathrm{E}\left[\mathcal{Q}_q(S_{0:T}) | I_{S_0}, I_f\right]$$
$$= \int\limits_{S_{0:T}} \mathcal{Q}_q(S_{0:T}) \Pr\left(S_{0:T} | I_{S_0}, I_f\right) \mathrm{d}S_{0:T}$$

$$J_q = \mathrm{E}\left[\mathcal{Q}_q(S_0, S_T) | I_{S_0}, I_f\right]$$
$$= \int\limits_{S_T} \int\limits_{S_0} \mathcal{Q}_q(S_0, S_T) \Pr\left(S_T, S_0 | I_{S_0}, I_f\right) \mathrm{d}S_0 \ \mathrm{d}S_T,$$

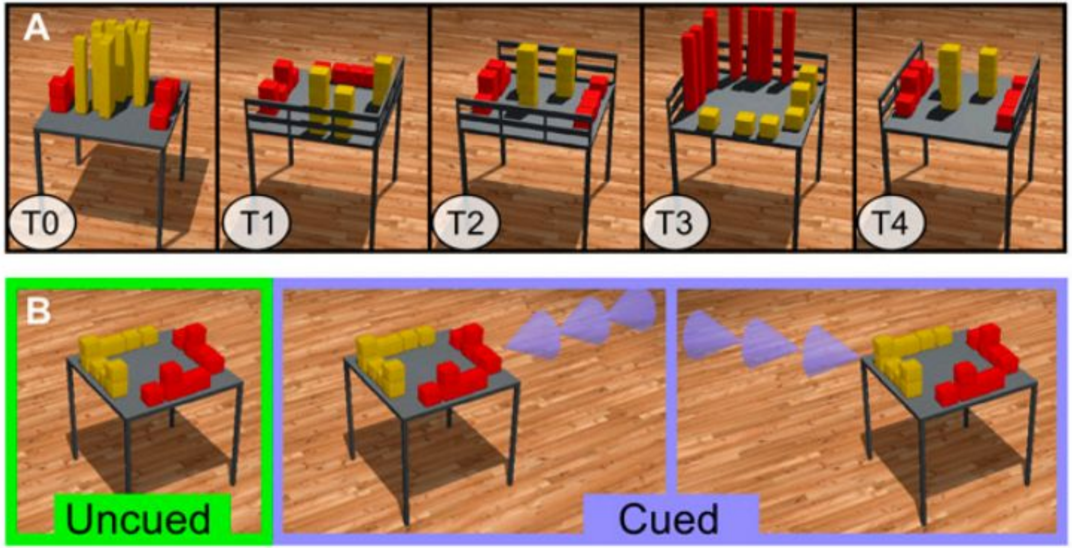
Physical simulation

$$J_q^{MC} \approx \frac{1}{k} \sum_{i=1}^{k} \mathcal{Q}_q\Big((G_0, \mu)^{(i)}, \Psi\Big((G_0, \mu)^{(i)}, f^{(i)}, 0:T \Big) \Big)$$

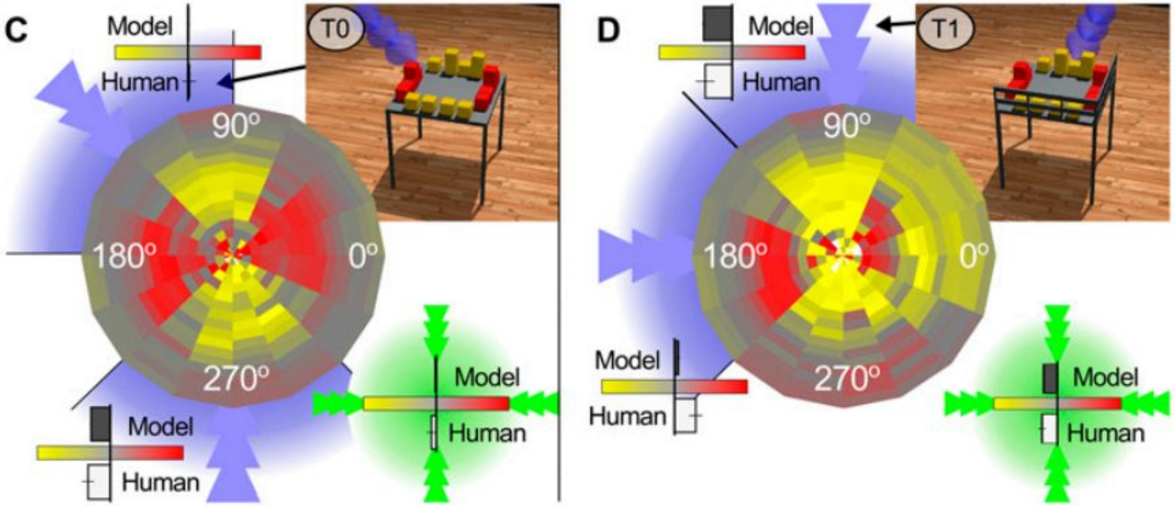
Experiments

1. Will it fall?
2. In which direction?
3. Will it fall? - Varying object masses
4. In which direction? - Varying object masses
5. Which block will fall? - Varying Object Shapes, Physical Obstacles, and Applied Forces.

Experiments



Experiments



Conclusion

Human’s intuitive physical judgments can be viewed as a form of probabilistic inference over the principles of Newtonian mechanic

Thank you!

References

- [Simulation as an engine of physical scene understanding]
- [Human-level concept learning through probabilistic program induction]