EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Probabilistic program induction of symbols

Or: More complex models!

- *The Language of Thought: computational cognitive science approaches to category learning*
- Who: Fausto Carcassi
- When: Sommer semester 2022

Where are we?

- Last week we have seen a very simple application of the LoT idea to categorization with a logical language.
- We have also learned more about the LOTlib3 library
- However, we have not seen the full power of LoT yet compared to e.g., deep learning methods.
- Today we'll have a look at what we can do with serious LoT models!
- The paper we'll look at (Lake et al (2015), *Human-level concept learning through probabilistic program induction*) is a bit old now, but nonetheless very nice.
- If there is time left, we'll also have a brief look at the paper on acquisition of kinship terms.

Few-shot learning

- Humans can learn a huge amount from a single instance. For instance, consider the following object:

  - Based on just this one instance, we can do loads.
  - E.g., classify new examples:

Few-shot learning

- Humans can learn a huge amount from a single instance. For instance, consider the following object:

  - Based on just this one instance, we can do loads.
  - E.g., generate new examples:

Few-shot learning

- Humans can learn a huge amount from a single instance. For instance, consider the following object:

  - Based on just this one instance, we can do loads.
  - E.g., parse the object into parts:

Few-shot learning

- Humans can learn a huge amount from a single instance. For instance, consider the following object:

- Based on just this one instance, we can do loads.
- E.g., generate new concepts:

Cognitive problem

- There is a *cognitive* question of how humans are capable of learning such powerful generalizations from such sparse data.
- Typical machine learning algorithms only do one of these, and they usually require more data!
- In the past weeks we've been learning a new learning algorithm, so let's see how it performs with respect to these challenges.

Program induction

- In order to see what the paper is going, we need a slightly different framing from what we've seen this far.
- Consider the following problem:
    - We have some input-output combos from an unknown computer program
    - We want to infer a computer program that gives those input-outputs.
- In principle, there's many ways of doing this, e.g., genetic algorithms
- You can probably see how this is related to the stuff we've seen this far: we can interpret inference in an LoT as a case of program induction.

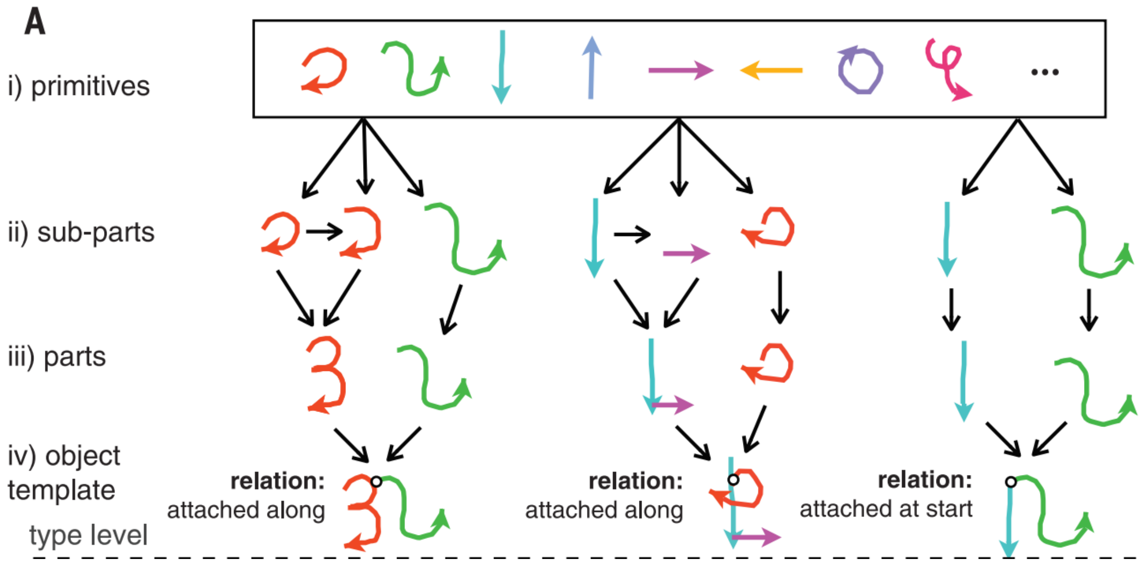
Some fundamental ideas

The paper has very many ideas, but the main ones are to bring together:

- Compositionality
  - Programs are build compositionally like we have seen with pLoT
- Causality
  - The programs capture the causal structure of how the images are generated
- Learning-to-learn
  - In addition to the things we've already learned, this model build a *hierarchical* prior, where experiences with previous concepts change the probability of new ones

Bayesian Program Learning

- Bayesian Program Learning is introduced in Lake et al
- It can learn visual concepts from a single example and generalize in a way very similar to how humans do it
- In the model, concepts are represented by little computer programs that define procedures for drawing images, generated as follows in an LoT:

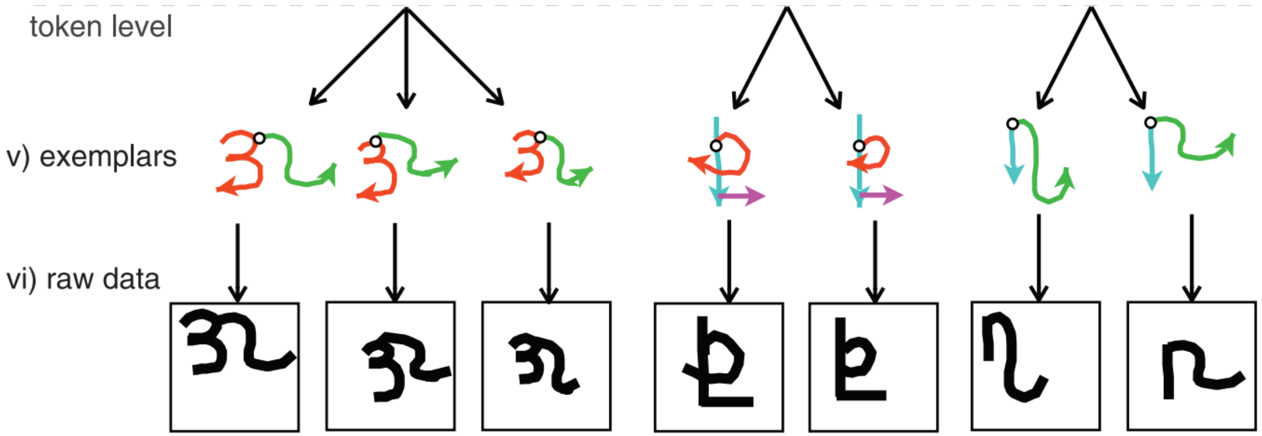


B

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

Bayesian Program Learning

- Once some character types are generated in the LoT, a specific instance of a *drawing* of the characters can also be generated as follows:



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 $I^{(m)}$

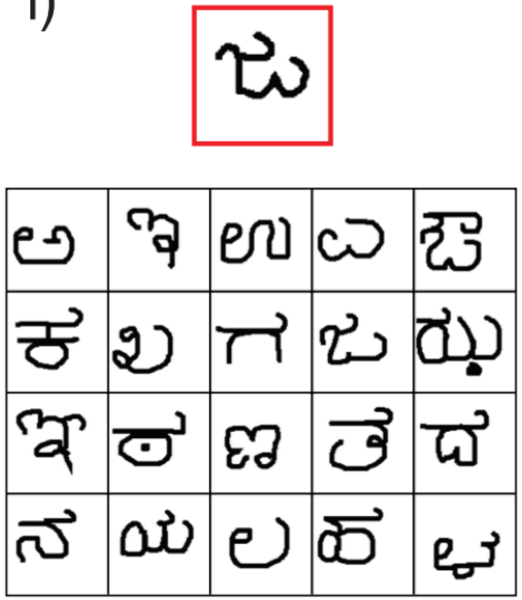
The plan

- The basic idea of the paper is to test Bayesian Program Learning with 5 tasks, and compare its performance with other algorithms as well as humans.
- The tasks involve the Omniglot dataset, which collects multiple examples of 1625 written characters from 50 different writing systems.
  - This includes both images and *pen strokes*

The five tasks

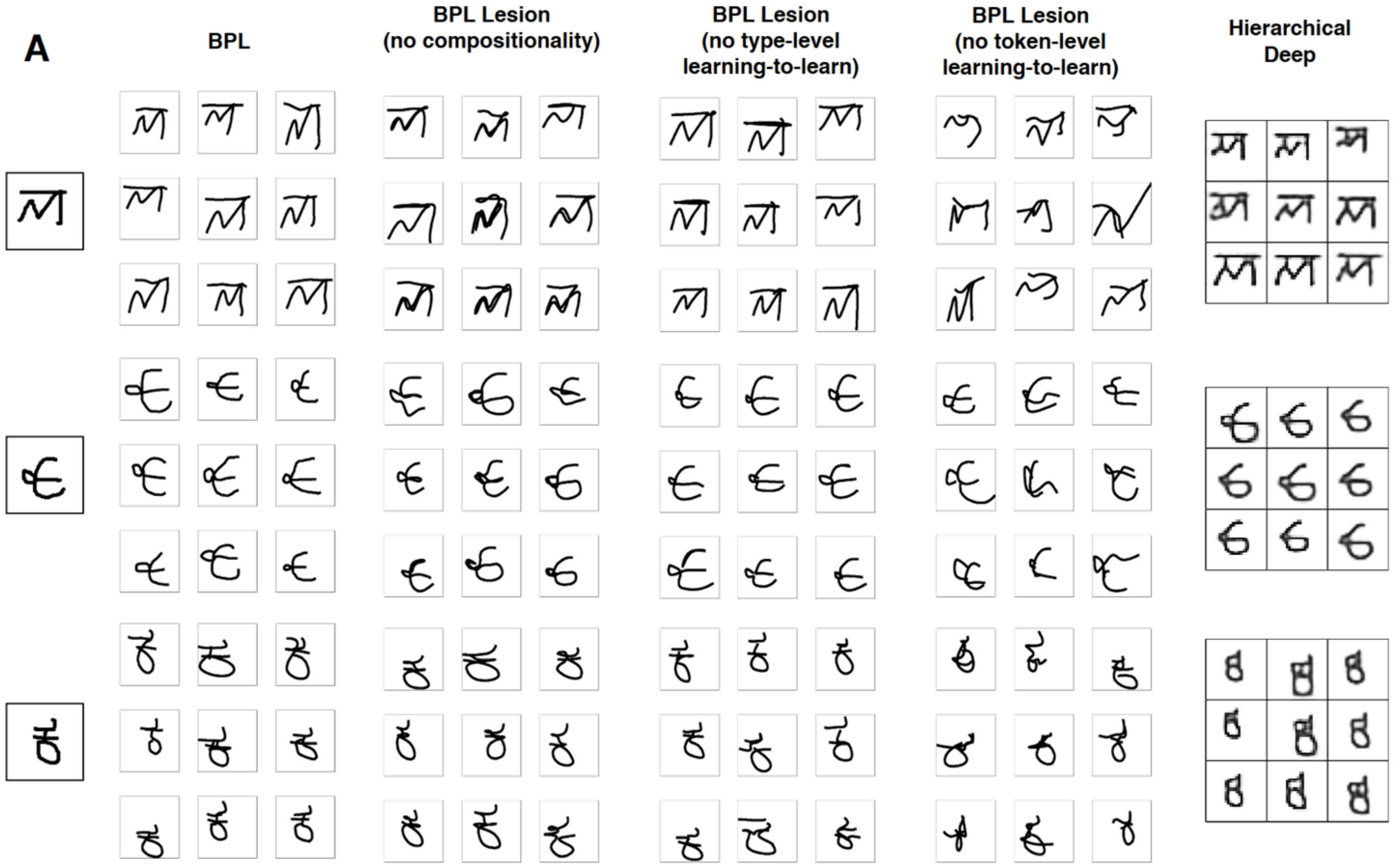
1. One-shot classification of characters:

i)



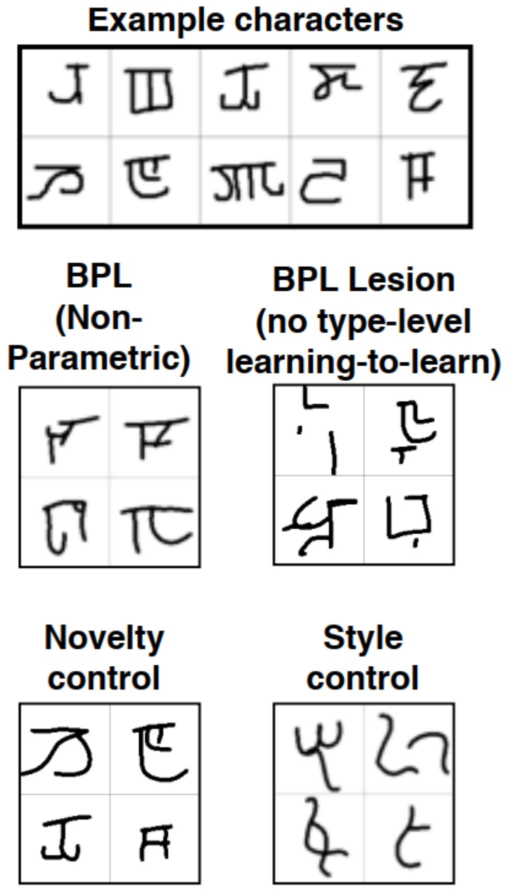
The five tasks

2. Generating new examples:



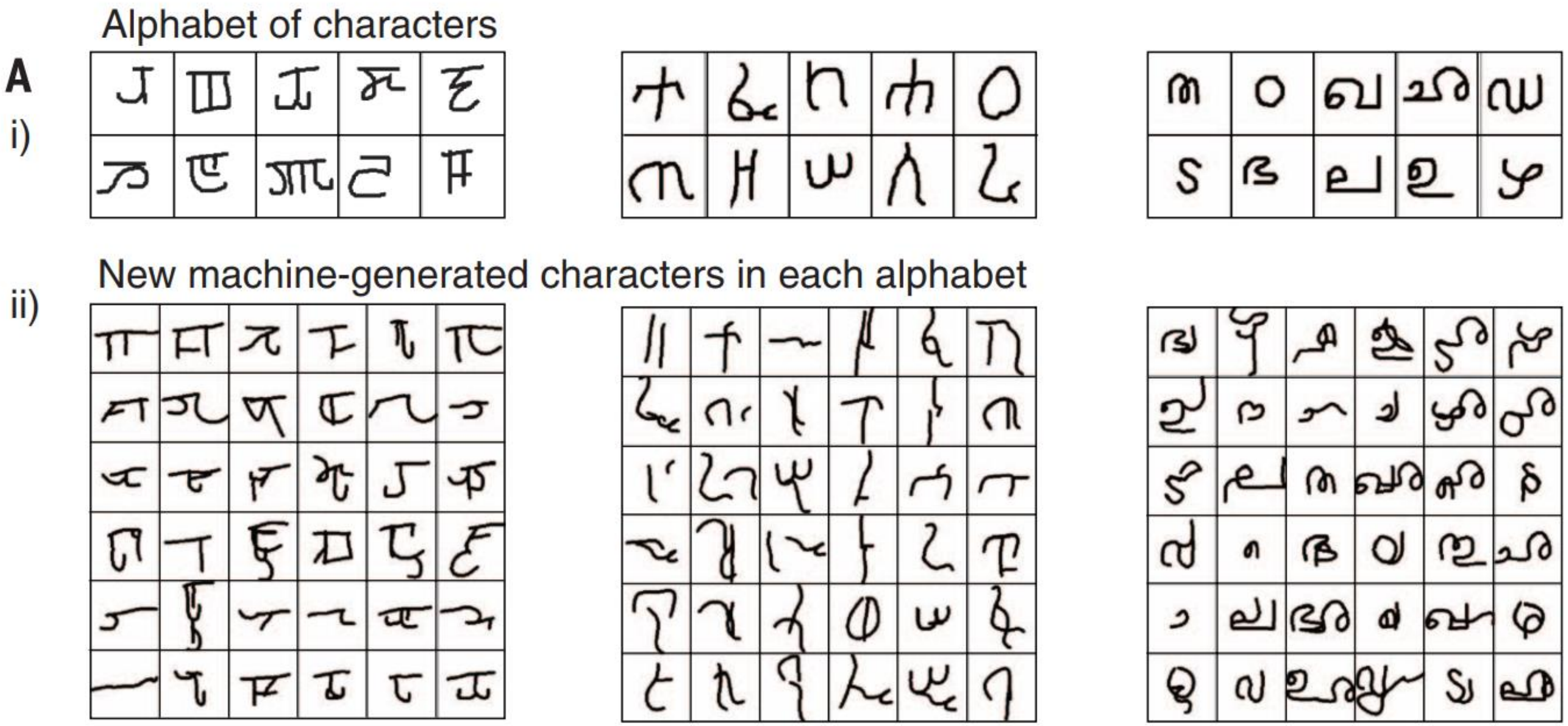
The five tasks

3. Generating new concepts:



The five tasks

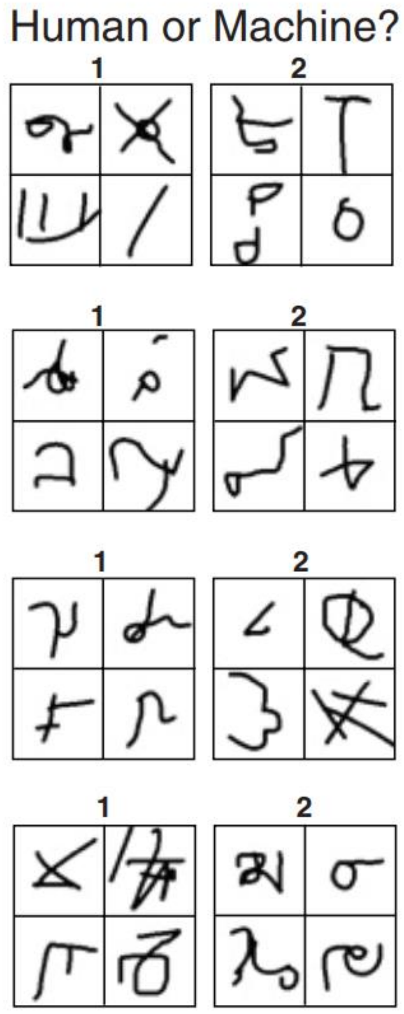
4. Generating new concepts (from type):



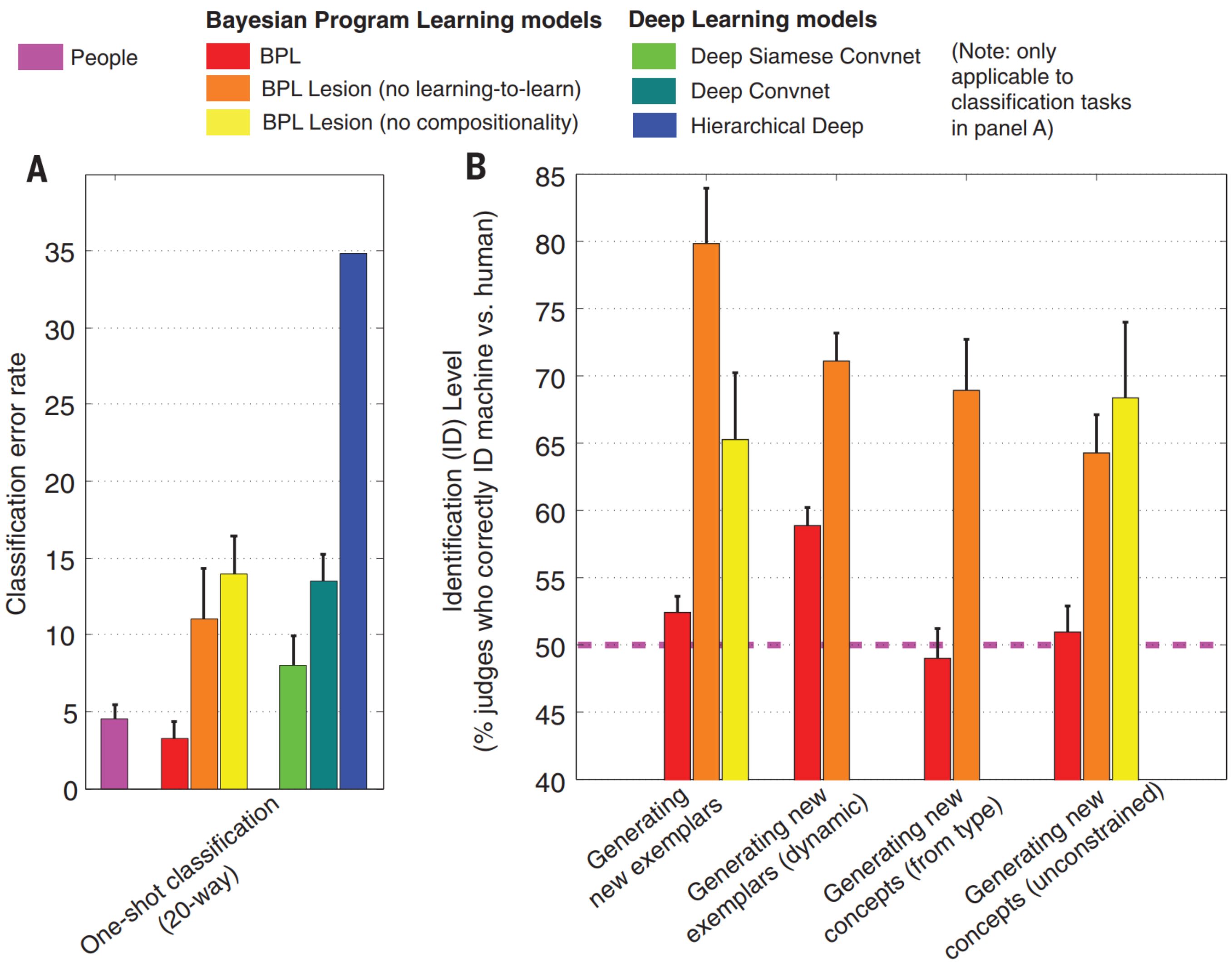
The five tasks

5. Generating new concepts (unconstrained):

(Machines are 2; 1; 1; 2)



Summary of results



Future directions?

- A natural generalization is to implement similar models for other cultural phenomena, like dances or gestures.
- Another direction is to use this model to study acquisition of characters in the alphabet.
- What else do you think could be done with this model?

Kinships terms

- Let's move onto Mollica & Piantadosi (2021), *Logical word learning: The case of kinship*.
- Kinship terms are word used to refer to where someone is in a family with respect to someone else.
- There is rich logical structure in kinship terms, since they semantically express complex relations.
- This is exactly the kind of conceptual domain where LoT models shine.
- So let's look at the way the Mollica paper sets up the LoT model!

Kinships terms – Data

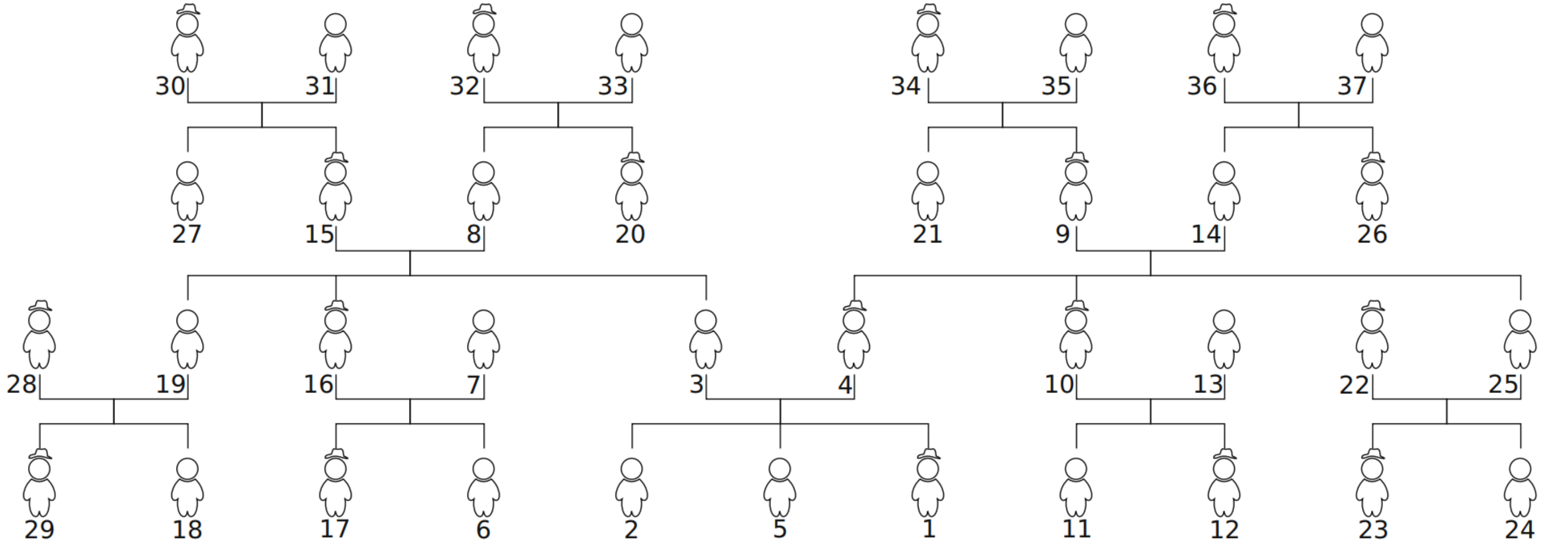
A single datapoint is a collection of four objects:

- A **speaker** who uses the kinship word
- A **word** (used by the speaker)
- A **referent** identified by the word
- A **context**, which consists of a family tree

From this kind of datapoint, the child has to infer the meaning of kinship terms!

Kinships terms – Hypothesis space

- A hypothesis is a function that characterizes a set of people in a family from the point of view of the speaker.
- The model considers 37 possible people (here numbered by the rank of number of interactions with the speaker):



Kinships terms – PCFG induced prior

The PCFG contains the following primitives:

$\text{SET} \xrightarrow{1} \text{union(SET,SET)}$	$\text{SET} \xrightarrow{1} \text{parent(SET)}$	$\text{SET} \xrightarrow{1} \text{generation0(SET)}$	$\text{SET} \xrightarrow{1} \text{male(SET)}$
$\text{SET} \xrightarrow{1} \text{intersection(SET,SET)}$	$\text{SET} \xrightarrow{1} \text{child(SET)}$	$\text{SET} \xrightarrow{1} \text{generation1(SET)}$	$\text{SET} \xrightarrow{1} \text{female(SET)}$
$\text{SET} \xrightarrow{1} \text{difference(SET,SET)}$	$\text{SET} \xrightarrow{1} \text{lateral(SET)}$	$\text{SET} \xrightarrow{1} \text{generation2(SET)}$	$\text{SET} \xrightarrow{1} \text{sameGender(SET)}$
$\text{SET} \xrightarrow{1} \text{complement(SET)}$	$\text{SET} \xrightarrow{1} \text{coreside(SET)}$	$\text{SET} \xrightarrow{\frac{1}{37}} \text{concreteReferent}$	$\text{SET} \xrightarrow{1} \text{all}$ $\text{SET} \xrightarrow{10} \text{X}$

Prior probabilities are calculated as usual! E.g.,

English	*aunt*	PZ, PGW	female(difference(generation1(X), parent(X)))
	brother	B	male(child(parent(X)))
	cousin	PGC, PGEC	difference(generation0(X), child(parent(X)))
	father	F	male(parent(X))
	grandma	PM	female(parent(parent(X)))
	grandpa	PF	male(parent(parent(X)))
	mother	M	female(parent(X))
	sister	Z	female(child(parent(X)))
	uncle	PB, PGH	male(difference(generation1(X), parent(X)))

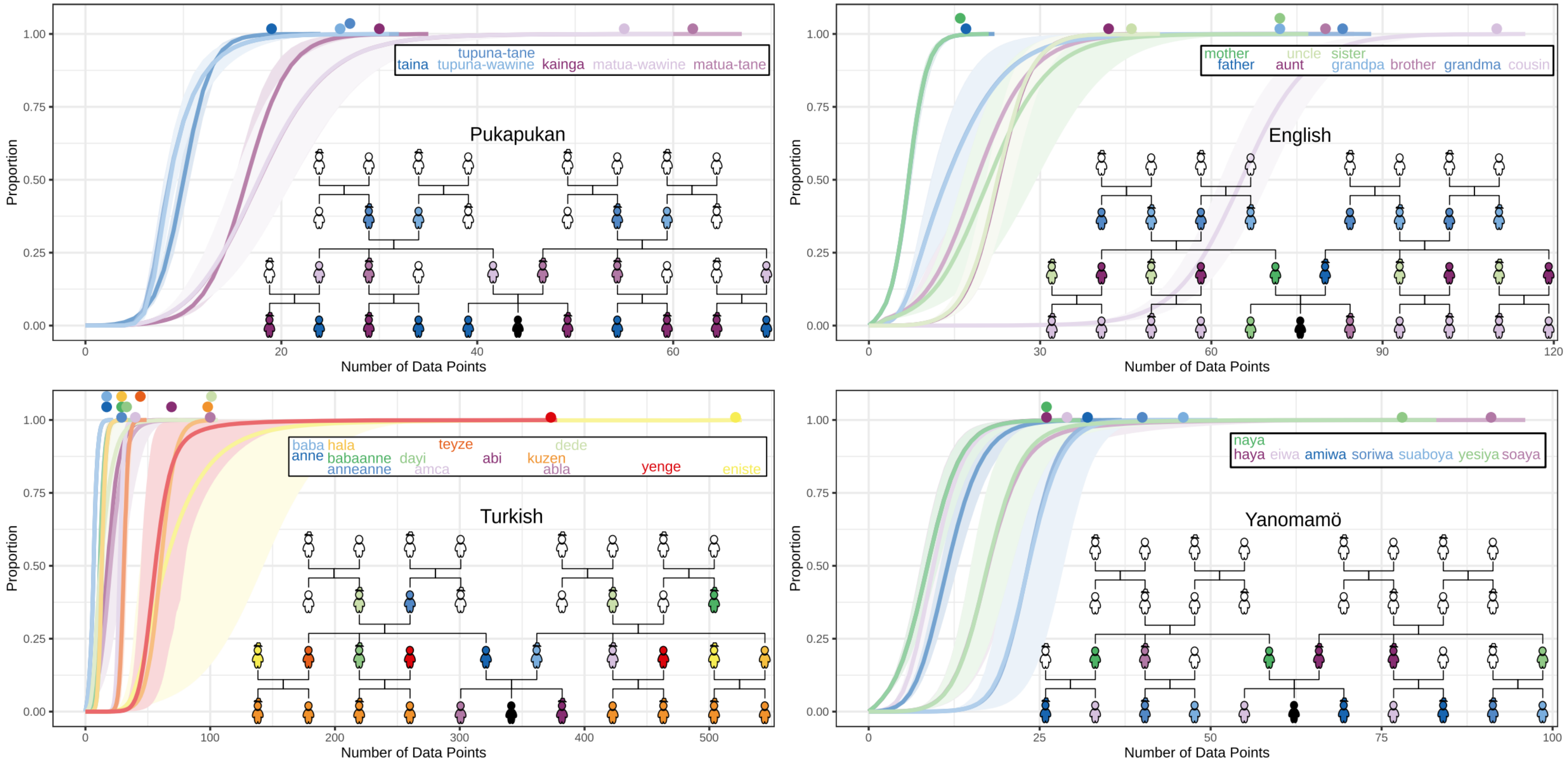
Kinships terms – Likelihood function

- The data is generated in one of two ways:
- With probability α, the data is generated by the hypothesis (i.e. one of the people is sampled)
- With probability $1 - \alpha$, the data is generated randomly.
- This produces the following likelihood function:

$$P(d|h) = \delta_{d \in h} \cdot \frac{\alpha}{|h|} + \frac{1-\alpha}{|\mathcal{D}|}$$

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Learning kinship systems in some langs



Main properties of the model

- The model shows a preference for concrete reference (single individuals) over classes of individuals when there are few datapoints.
    - This is consistent with what children do!
- The model predicts *overextension*
    - The phenomenon where children learn a larger category that includes more individuals than the word’s true reference.
- Characteristic-to-defining shift
    - A pattern in overextension where young children over-extend with characteristic features (“robbers are mean”) vs defining features (“robbers steal things”).

Main properties of the model

- Order of acquisition of model and children mostly align:

Empirical Order	Word	Original H&C Order & Formalization	Log Prior	CHILDES Freq.
1	*mother*	Level I: [X PARENT Y][FEMALE]	-9.457	6812
1	*father*	Level I: [X PARENT Y][MALE]	-9.457	3605
2	*brother*	Level III: [X CHILD A][A PARENT Y][MALE]	-13.146	41
2	*sister*	Level III: [X CHILD A][A PARENT Y][FEMALE]	-13.146	89
3	*grandma*	Level II: [X PARENT A][A PARENT Y][FEMALE]	-13.146	526
3	*grandpa*	Level II: [X PARENT A][A PARENT Y][MALE]	-13.146	199
4	*aunt*	Level IV: [X SIB A][A PARENT Y][FEMALE]	-19.320	97
4	*uncle*	Level IV: [X SIB A][A PARENT Y][MALE]	-19.320	68
4	*cousin*	Level IV: [X CHILD A][A SIB B][B PARENT Y]	-18.627	14

- The paper contains much more, including experimental results, but we do not have time to go through it all.

Summary

- Today, we have seen two new applications of the LoT, in the guise of *program induction*: learning a computer program in a domain-specific language from input/output relations.
- In the lab this week, we will see how to implement a category learning model in LOTlib3. If there's time left we'll also try to expand it to make it do more powerful stuff.
- Next week, we will see how to apply pLoT to other domains.