Informal Intro to Gold-Style Language Learning — With a Sample Target of RAIR-Lab Interest

(or: Human Language Acquisition is Hard!; Sorry Nonhuman Animals)

S Bringsjord
Are Humans Rational?
RPI
11/20/17

Informal Intro to Gold-style Learning …
Formal reconstruction of ...
Formal reconstruction of ...

1. *theoretically possible realities;*
Formal reconstruction of ...

1. *theoretically possible realities*;

2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality is reality);
1. *theoretically possible realities*;

2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality is reality);

3. *the data about any given theoretically possible reality, were it actual* (the list corresponds to things the language learner perceives in the empirical world);
Formal reconstruction of ...

1. *theoretically possible realities*;

2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality is reality);

3. *the data about any given theoretically possible reality, were it actual* (the list corresponds to things the language learner perceives in the empirical world);

4. *language learner*;
Formal reconstruction of ...

1. *theoretically possible realities*;

2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality is reality);

3. *the data about any given theoretically possible reality, were it actual* (the list corresponds to things the language learner perceives in the empirical world);

4. *language learner*;

5. *successful/unsuccessful behavior by a language learner working in a given, possible reality.*
Simple Game Setup
Simple Game Setup

- A set of positive integers is *describable* iff it can be uniquely described by an English expression. Let $D$ be the set of such sets.
Simple Game Setup

• A set of positive integers is describable iff it can be uniquely described by an English expression. Let $D$ be the set of such sets.

• E.g., “all positive even integers.” i.e., $\{2, 4, 6, 8, \ldots\}$. These are the aforementioned theoretically possible realities.
Simple Game Setup

• A set of positive integers is *describable* iff it can be uniquely described by an English expression. Let $D$ be the set of such sets.

• E.g., “all positive even integers.” I.e., $\{2, 4, 6, 8, \ldots\}$. These are the aforementioned *theoretically possible realities*.

• $C$ is a proper subset of $D$ defined as follows: All sets that contain every positive integer, save for one.
Simple Game Setup

• A set of positive integers is *describable* iff it can be uniquely described by an English expression. Let \( D \) be the set of such sets.

• E.g., “all positive even integers.” I.e., \( \{2, 4, 6, 8, ...\} \). These are the aforementioned *theoretically possible realities*.

• \( C \) is a proper subset of \( D \) defined as follows: All sets that contain every positive integer, save for one.

• E.g., here is a member of \( C \): “all positive integers except for 2.” I.e., \( \{1, 3, 4, 5, ...\} \).
More Setup
More Setup

• Nature: “I’ve selected a member of C. You, (language-learning) Child, must discover the set I have in mind. I shall give you clues, as follows. My secret member shall be ordered in a list L that contains all members of my secret set, and I shall then present the members of L one at a time to you. Each time I do so, you offer a hypothesis in the form of an English expression that uniquely describes a set of positive integers. You win the game iff you make only a finite number of conjectures, and the last one is correct.”
More Setup

• Nature: “I’ve selected a member of C. You, (language-learning) Child, must discover the set I have in mind. I shall give you clues, as follows. My secret member shall be ordered in a list \( L \) that contains all members of my secret set, and I shall then present the members of \( L \) one at a time to you. Each time I do so, you offer a hypothesis in the form of an English expression that uniquely describes a set of positive integers. You win the game iff you make only a finite number of conjectures, and the last one is correct.”

• Nature: “Let’s play!”
More Setup

• Nature: “I’ve selected a member of C. You, (language-learning) Child, must discover the set I have in mind. I shall give you clues, as follows. My secret member shall be ordered in a list $L$ that contains all members of my secret set, and I shall then present the members of $L$ one at a time to you. Each time I do so, you offer a hypothesis in the form of an English expression that uniquely describes a set of positive integers. You win the game iff you make only a finite number of conjectures, and the last one is correct.”

• Nature: “Let’s play!”

• Child: “Okay!”
Some Playing
Some Playing

- N: I.
Some Playing

- N: I.
- Child: Silence.
Some Playing

- N: 1.
- Child: Silence.
- N: 3.
Some Playing

• N: 1.

• Child: Silence.

• N: 3.

• Child: “All positive integers except for 2.”
Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
Some Playing

• N: 1.

• Child: Silence.

• N: 3.

• Child: “All positive integers except for 2.”

• N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.

• Child: Silence.

• N: 2.
Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (multiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (multiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
- Child: 14.
Some Playing

• N: 1.
• Child: Silence.
• N: 3.
• Child: “All positive integers except for 2.”
• N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
• Child: Silence.
• N: 2.
• N (mutiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
• Child: 14.
• N: 14.
Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (multiple steps): 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
- Child: 14.
- N: 14.
- ...
Some Playing

- N: 1.
- Child: Silence.
- N: 3.
- Child: “All positive integers except for 2.”
- N (multiple steps): 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.
- Child: Silence.
- N: 2.
- N (multiple steps): 15, 16, 17, 18, 19, 20, 21, 22, 23, 24.
- Child: 14.
- N: 14.
- ...
- Is Child using some algorithm?
Guessing Algorithm (G)
Guessing Algorithm (G)

Let $S$ be the set of numbers presented so far. And let $m$ be the smallest positive integer that isn’t a member of $S$. Output the hypothesis: “All positive integers except for $m$.” If this was your last hypothesis, remain silent.
Guessing Algorithm (G)

Let \( S \) be the set of numbers presented so far. And let \( m \) be the smallest positive integer that isn’t a member of \( S \). Output the hypothesis: “All positive integers except for \( m \).” If this was your last hypothesis, remain silent.

So, e.g., \( G([4, 5, 8, 1]) = 2 \).
Theorem I.
Theorem 1.

Regardless of what member of $C$ is chosen by Nature at the start of the game, and no matter what list $L$ is produced from that member, $G$ will win the game for you!
Theorem 1.

Regardless of what member of $C$ is chosen by Nature at the start of the game, and no matter what list $L$ is produced from that member, $G$ will win the game for you!

(Sedulous-and-inquisitive: Prove it!)
One Slight Mod
One Slight Mod

We stipulate that $\mathbb{Z}^+$ is in $C$. 
Theorem 2.
Theorem 2.

There exists no algorithm that is guaranteed to win this new game. More precisely, for every algorithm $A$ there is a set in the (expanded!) set $C$, and a list $L$ of $C$, such that $A$ fails to produce a last, correct hypothesis.
Theorem 2.

There exists no algorithm that is guaranteed to win this new game. More precisely, for every algorithm $A$ there is a set in the (expanded!) set $C$, and a list $L$ of $C$, such that $A$ fails to produce a last, correct hypothesis.

(Sedulous-and-inquisitive: Prove it!)
Some Issues to Keep in Mind
Some Issues to Keep in Mind

1. For this class, and where we are in this class, a language is no doubt more appropriate than a set of integers as a “possible reality,” but the idea is that a set of integers can “code” a language ...
Some Issues to Keep in Mind

1. For this class, and where we are in this class, a language is no doubt more appropriate than a set of integers as a “possible reality,” but the idea is that a set of integers can “code” a language ...

2. The set of possible realities must on this approach be countable. What then about physical quantities whose values are arbitrary real numbers? Isn’t this what we see in physics as carried out by real physicists?
Some Issues to Keep in Mind

1. For this class, and where we are in this class, a *language* is no doubt more appropriate than a set of integers as a “possible reality,” but the idea is that a set of integers can “code” a language ...

2. The set of possible realities must on this approach be *countable*. What then about physical quantities whose values are arbitrary real numbers? Isn’t this what we see in physics as carried out by real physicists?

3. Language-learning Children are assumed here to be *mechanical*. In traditional CLT, language learners are identified with standard computer programs/Turing machines. (Since Selmer thinks that human persons are capable of information processing beyond Turing machines, the assumption here is one he can’t ultimately swallow.)
Some Issues to Keep in Mind

1. For this class, and where we are in this class, a language is no doubt more appropriate than a set of integers as a “possible reality,” but the idea is that a set of integers can “code” a language ...

2. The set of possible realities must on this approach be countable. What then about physical quantities whose values are arbitrary real numbers? Isn’t this what we see in physics as carried out by real physicists?

3. Language-learning Children are assumed here to be mechanical. In traditional CLT, language learners are identified with standard computer programs/Turing machines. (Since Selmer thinks that human persons are capable of information processing beyond Turing machines, the assumption here is one he can’t ultimately swallow.)

4. Knowledge is ignored. For example, does the Child in the above game know that some winning hypothesis is a winning hypothesis? If that were the case, then given that knowledge is at least “justified true belief,” where is the justification?
Some Issues to Keep in Mind

1. For this class, and where we are in this class, a language is no doubt more appropriate than a set of integers as a “possible reality,” but the idea is that a set of integers can “code” a language ...

2. The set of possible realities must on this approach be countable. What then about physical quantities whose values are arbitrary real numbers? Isn’t this what we see in physics as carried out by real physicists?

3. Language-learning Children are assumed here to be mechanical. In traditional CLT, language learners are identified with standard computer programs/Turing machines. (Since Selmer thinks that human persons are capable of information processing beyond Turing machines, the assumption here is one he can’t ultimately swallow.)

4. Knowledge is ignored. For example, does the Child in the above game know that some winning hypothesis is a winning hypothesis? If that were the case, then given that knowledge is at least “justified true belief,” where is the justification?

- Of course, given our Theorems 1 & 2 from above, we can in some sense know that the system Child is using will win. But the Child himself/herself doesn’t know.
CLT-based Model of Language Learning

$\sigma \in \text{SEQ}$

$S$

$L$ (r.e.)

$M_i, i \in \mathbb{N}$

such that $M_i$ accepts $L$
CLT-based Model of Language Learning

$\sigma \in \text{SEQ}$

$S$

$L \ (r.e.)$

$M_i, i \in \mathbb{N}$ such that $M_i$ accepts $L$

Turing machine
CLT-based Model of Language Learning

\[ L \text{ (r.e.)} \]

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \]

\[ i \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

Turing machine
CLT-based Model of Language Learning

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \]

\[ S \]

\[ L \text{ (r.e.)} \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

\[ M_1 \]

Turing machine
CLT-based Model of Language Learning

\[ L \text{ (r.e.)} \]

\[ \sigma \in \text{SEQ} \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)
CLT-based Model of Language Learning

\[ L \text{ (r.e.)} \]

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \# \]

\[ i \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

Turing machine
CLT-based Model of Language Learning

σ ∈ SEQ

T = u₀, u₁, #, #, #,

i

S

L (r.e.)

Mᵢ, i ∈ N

such that Mᵢ accepts L

Turing machine
CLT-based Model of Language Learning

\[ L \text{ (r.e.)} \]

\( \sigma \in \text{SEQ} \)

\( T = u_0, u_1, \#, \#, \# \)

\( i \)

\( M_2 \)

\( 0 \)

\( M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

Turing machine
CLT-based Model of Language Learning

$\sigma \in \text{SEQ}$

$L_{(r.e.)}$

$S$

$M_i, i \in \mathbb{N}$

such that $M_i$ accepts $L$

Turing machine
CLT-based Model of Language Learning

\[ L \text{ (r.e.)} \]

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

Turing machine
CLT-based Model of Language Learning

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ \sigma \in \text{SEQ} \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)
CLT-based Model of Language Learning

\[ L \text{ (r.e.)} \]

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ M_3 \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

Turing machine
CLT-based Model of Language Learning

\[ L \text{ (r.e.)} \]

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ i \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

And when is identification of \( L \) achieved?

Turing machine
CLT-based Model of Language Learning

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ i \]

\[ L \text{ (r.e.)} \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

And when is identification of \( L \) achieved?

\[ \forall n \in \mathbb{N} \ S(T^n) = M_k \]
CLT-based Model of Language Learning

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ i \]

\[ L \ (\text{r.e.}) \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

And when is identification of \( L \) achieved?

\[ \forall \infty n \in \mathbb{N} \quad S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
RAIR-Lab-relevant Target for Learning …
Recall final slide in Rikhiya’s Slide Deck ...
Chomsky Hierarchy of Languages
How About Learning the Grammar of the Pure Predicate Calculus?

\[
\begin{align*}
\text{Formula} & \Rightarrow \text{AtomicFormula} \\
& \mid (\text{Formula Connective Formula}) \\
& \mid \neg \text{Formula} \\
\hline
\text{AtomicFormula} & \Rightarrow (\text{Predicate } \text{Term}_1 \ldots \text{Term}_k) \\
& \mid (\text{Term } = \text{Term}) \\
\hline
\text{Term} & \Rightarrow (\text{Function } \text{Term}_1 \ldots \text{Term}_k) \\
& \mid \text{Constant} \\
\hline
\text{Connective} & \Rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \\
\hline
\text{Predicate} & \Rightarrow P_1 \mid P_2 \mid P_3 \ldots \\
\text{Constant} & \Rightarrow c_1 \mid c_2 \mid c_3 \ldots \\
\text{Function} & \Rightarrow f_1 \mid f_2 \mid f_3 \ldots 
\end{align*}
\]
How About Learning the Grammar of the Pure Predicate Calculus?

Sally likes Bill.
(Likes sally bill)

| Formula       | ⇒ | AtomicFormula |
|               |   | (Formula Connective Formula) |
|               |   | ¬ Formula |

| AtomicFormula | ⇒ | (Predicate Term₁ ... Termₖ) |
|              |   | (Term = Term) |

| Term           | ⇒ | (Function Term₁ ... Termₖ) |
|               |   | Constant |

| Connective     | ⇒ | ∧ | ∨ | → | ↔ |

| Predicate      | ⇒ | P₁ | P₂ | P₃ | ... |
| Constant       | ⇒ | c₁ | c₂ | c₃ | ... |
| Function       | ⇒ | f₁ | f₂ | f₃ | ... |
How About Learning the Grammar of the Pure Predicate Calculus?

| Formula |  ⇒  | AtomicFormula |
|         |  | (Formula Connective Formula) |
|         |  | ¬ Formula |

| AtomicFormula |  ⇒  | (Predicate Term₁ ... Termₖ) |
|               |  | (Term = Term) |

| Term |  ⇒  | (Function Term₁ ... Termₖ) |
|      |  | Constant |

| Connective |  ⇒  | ∧ | ∨ | → | ↔ |

| Predicate |  ⇒  | P₁ | P₂ | P₃ | ... |
| Constant  |  ⇒  | c₁ | c₂ | c₃ | ... |
| Function  |  ⇒  | f₁ | f₂ | f₃ | ... |
How About Learning the Grammar of the Pure Predicate Calculus?

Sally likes Bill.
(Likes sally bill)

<table>
<thead>
<tr>
<th>Formula</th>
<th>⇒  AtomicFormula</th>
<th>Sally likes Bill.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Formula Connective Formula)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>¬ Formula</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AtomicFormula</th>
<th>⇒  (Predicate Term₁ ... Termₖ)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Term = Term)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Term</th>
<th>⇒  (Function Term₁ ... Termₖ)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Constant</td>
</tr>
</tbody>
</table>

| Connective        | ⇒  ∧ | ∨ | → | ↔ |

| Predicate         | ⇒  P₁ | P₂ | P₃ | ... |
| Constant          | ⇒  c₁ | c₂ | c₃ | ... |
| Function          | ⇒  f₁ | f₂ | f₃ | ... |
How About Learning the Grammar of the Pure Predicate Calculus?

Formula → AtomicFormula
| (Formula Connective Formula)
| ¬ Formula

AtomicFormula → (Predicate Term₁ ... Termₖ)
| (Term = Term)

Term → (Function Term₁ ... Termₖ)
| Constant

Connective → ∧ | ∨ | → | ↔

Predicate → \( P_1 | P_2 | P_3 \ldots \)
Constant → \( c_1 | c_2 | c_3 \ldots \)
Function → \( f_1 | f_2 | f_3 \ldots \)

Lexicon

Sally likes Bill.
(Likes sally bill)
How About Learning the Grammar of the Pure Predicate Calculus?

_Sally likes Bill._

(Likes _sally bill_)

**Lexicon**

- **Predicate** → $P_1 \mid P_2 \mid P_3 \ldots$
- **Constant** → $c_1 \mid c_2 \mid c_3 \ldots$
- **Function** → $f_1 \mid f_2 \mid f_3 \ldots$

- **Formula** → AtomicFormula
  - (Formula Connective Formula)
  - $\neg$ Formula

- **AtomicFormula** → (Predicate Term$_1$ ... Term$_k$)
  - (Term = Term)

- **Term** → (Function Term$_1$ ... Term$_k$)
  - Constant

- **Connective** → $\land \mid \lor \mid \rightarrow \mid \leftrightarrow$
How About Learning the Grammar of the Pure Predicate Calculus?

Formula $\Rightarrow$ AtomicFormula
  $|$ (Formula Connective Formula)
  $|$ $\neg$ Formula

AtomicFormula $\Rightarrow$ (Predicate Term$_1$ ... Term$_k$)
  $|$ (Term = Term)

Term $\Rightarrow$ (Function Term$_1$ ... Term$_k$)
  $|$ Constant

Connective $\Rightarrow$ $\wedge$ $|$ $\vee$ $|$ $\rightarrow$ $|$ $\leftrightarrow$

Predicate $\Rightarrow$ $P_1$ $| P_2$ $| P_3$ ...
Constant $\Rightarrow$ $c_1$ $| c_2$ $| c_3$ ...
Function $\Rightarrow$ $f_1$ $| f_2$ $| f_3$ ...

Sally likes Bill.
(Likes sally bill)
How About Learning the Grammar of the Pure Predicate Calculus?

<table>
<thead>
<tr>
<th>Formula</th>
<th>⇒</th>
<th>AtomicFormula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(Formula Connective Formula)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>¬ Formula</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AtomicFormula</th>
<th>⇒</th>
<th>(Predicate Term₁ ... Termₖ)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(Term = Term)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Term</th>
<th>⇒</th>
<th>(Function Term₁ ... Termₖ)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Constant</td>
</tr>
</tbody>
</table>

| Connective          | ⇒ | ∧ | ∨ | → | ↔ |

<table>
<thead>
<tr>
<th>Likes</th>
<th>⇒</th>
<th>P₁</th>
<th>P₂</th>
<th>P₃</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicate</td>
<td></td>
<td>c₁</td>
<td>c₂</td>
<td>c₃</td>
<td>...</td>
</tr>
<tr>
<td>Constant</td>
<td></td>
<td>f₁</td>
<td>f₂</td>
<td>f₃</td>
<td>...</td>
</tr>
</tbody>
</table>

Lexicon
How About Learning the Grammar of the Pure Predicate Calculus?

<table>
<thead>
<tr>
<th>Formula</th>
<th>$\Rightarrow$</th>
<th>AtomicFormula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(Formula Connective Formula)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\neg$ Formula</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AtomicFormula</th>
<th>$\Rightarrow$</th>
<th>(Predicate Term$_1$ ... Term$_k$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(Term $=$ Term)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Term</th>
<th>$\Rightarrow$</th>
<th>(Function Term$_1$ ... Term$_k$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Constant</td>
</tr>
</tbody>
</table>

| Connective               | $\Rightarrow$ | $\wedge$ | $\vee$ | $\rightarrow$ | $\leftrightarrow$ |

<table>
<thead>
<tr>
<th>Likes Predicate</th>
<th>$\Rightarrow$</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>$\Rightarrow$</td>
<td>$c_1$</td>
<td>$c_2$</td>
<td>$c_3$</td>
<td>...</td>
</tr>
<tr>
<td>Function</td>
<td>$\Rightarrow$</td>
<td>$f_1$</td>
<td>$f_2$</td>
<td>$f_3$</td>
<td>...</td>
</tr>
</tbody>
</table>

Lexicon
How About Learning the Grammar of the Pure Predicate Calculus?

<table>
<thead>
<tr>
<th>Formula</th>
<th>$\Rightarrow$</th>
<th>AtomicFormula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$(\text{Formula Connective Formula})$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\neg \text{Formula}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AtomicFormula</th>
<th>$\Rightarrow$</th>
<th>$(\text{Predicate Term}_1 \ldots \text{Term}_k)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$(\text{Term} = \text{Term})$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Term</th>
<th>$\Rightarrow$</th>
<th>$(\text{Function Term}_1 \ldots \text{Term}_k)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Constant</td>
</tr>
</tbody>
</table>

| Connective       | $\Rightarrow$ | $\land$ | $\lor$ | $\rightarrow$ | $\leftrightarrow$ |

| Likes Predicate  | $\Rightarrow$ | $\text{P}_1$ | $\text{P}_2$ | $\text{P}_3$ \ldots |
|------------------|----------------|----------------|----------------|
| Constant         | $\Rightarrow$ | $\text{c}_1$ | $\text{c}_2$ | $\text{c}_3$ \ldots |
| Function         | $\Rightarrow$ | $\text{f}_1$ | $\text{f}_2$ | $\text{f}_3$ \ldots |
How About Learning the Grammar of the Pure Predicate Calculus?

Sally likes Bill.
(Likes sally bill)

Sally likes Bill and Bill likes Sally.
Sally likes Bill’s mother.
Sally likes Bill only if Bill’s mother is tall.

Lexicon
How About Learning the Grammar of the Pure Predicate Calculus?

Formula \( \Rightarrow \) AtomicFormula
| (Formula Connective Formula)
| \( \neg \) Formula

AtomicFormula \( \Rightarrow \) (Predicate Term\(_1\) \ldots Term\(_k\))
| (Term = Term)

Term \( \Rightarrow \) (Function Term\(_1\) \ldots Term\(_k\))
| Constant

Connective \( \Rightarrow \) \& | \lor | \rightarrow | \leftrightarrow

Likes Predicate \( \Rightarrow \) \( P_1 \mid P_2 \mid P_3 \ldots \)
Constant \( \Rightarrow \) \( c_1 \mid c_2 \mid c_3 \ldots \)
Function \( \Rightarrow \) \( f_1 \mid f_2 \mid f_3 \ldots \)

Lexicon

Sally likes Bill.
(Likes sally bill)

Sally likes Bill and Bill likes Sally.

Sally likes Bill’s mother.

Sally likes Bill only if Bill’s mother is tall.

Matilda is Bill’s super-smart mother.
How About Learning the Grammar of the Pure Predicate Calculus?

Formula $\Rightarrow$ AtomicFormula
\[ \mid (\text{Formula Connective Formula}) \]
\[ \mid \neg \text{Formula} \]

AtomicFormula $\Rightarrow$ (Predicate Term$_1$ … Term$_k$)
\[ \mid (\text{Term} = \text{Term}) \]

Term $\Rightarrow$ (Function Term$_1$ … Term$_k$)
\[ \mid \text{Constant} \]

Connective $\Rightarrow$ $\land$ | $\lor$ | $\rightarrow$ | $\leftrightarrow$

Lexicon

Sally likes Bill.
(Likes sally bill)

Sally likes Bill and Bill likes Sally.

Sally likes Bill’s mother.

Sally likes Bill only if Bill’s mother is tall.

Matilda is Bill’s super-smart mother.

5 plus 5 equals the number 10.
How About Learning the Grammar of the Pure Predicate Calculus?

<table>
<thead>
<tr>
<th>Formula</th>
<th>$\Rightarrow$</th>
<th>AtomicFormula</th>
<th>$\Rightarrow$</th>
<th>(Predicate Term₁ ... Termₖ)</th>
<th>(Term = Term)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term</td>
<td>$\Rightarrow$</td>
<td>(Function Term₁ ... Termₖ)</td>
<td>Constant</td>
<td>$\Rightarrow$</td>
<td>$\land$</td>
</tr>
<tr>
<td>Likes Predicate</td>
<td>$\Rightarrow$</td>
<td>$P_1$</td>
<td>$P_2$</td>
<td>$P_3$ ...</td>
<td>$c_1$</td>
</tr>
<tr>
<td>Constant</td>
<td>$\Rightarrow$</td>
<td>$c_1$</td>
<td>$c_2$</td>
<td>$c_3$ ...</td>
<td>$f_1$</td>
</tr>
<tr>
<td>Function</td>
<td>$\Rightarrow$</td>
<td>$f_1$</td>
<td>$f_2$</td>
<td>$f_3$ ...</td>
<td></td>
</tr>
</tbody>
</table>

Lexicon

Sally likes Bill.

(Likes sally bill)

Sally likes Bill and Bill likes Sally.

Sally likes Bill’s mother.

Sally likes Bill only if Bill’s mother is tall.

Matilda is Bill’s super-smart mother.

5 plus 5 equals the number 10.

...
How About Learning the Grammar of the Pure Predicate Calculus?

Formula $\Rightarrow$ AtomicFormula
| (Formula Connective Formula)
| ¬ Formula

AtomicFormula $\Rightarrow$ (Predicate Term$_1$ . . . Term$_k$)
| (Term = Term)

Term $\Rightarrow$ (Function Term$_1$ . . . Term$_k$)
| Constant

Connective $\Rightarrow$ $\land$ | $\lor$ | $\rightarrow$ | $\leftrightarrow$

Likes
Predicate $\Rightarrow$ $P_1$ | $P_2$ | $P_3$ . . .
Constant $\Rightarrow$ $c_1$ | $c_2$ | $c_3$ . . .
Function $\Rightarrow$ $f_1$ | $f_2$ | $f_3$ . . .

Sally likes Bill.
(Likes sally bill)

Sally likes Bill and Bill likes Sally.
Sally likes Bill’s mother.
Sally likes Bill only if Bill’s mother is tall.

Matilda is Bill’s super-smart mother.

5 plus 5 equals the number 10.

Make sure you can simulate a machine that says “Yes that sentence is okay!” whenever it’s conforms to this grammar!
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

\[ M_i, i \in \mathbb{N} \text{ such that } M_i \text{ accepts } L \]

And when is identification of \( L \) achieved?

\[ \forall \infty n \in \mathbb{N} \ S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \]

\[ i \]

\[ o \]

And when is identification of \( L \) achieved?

\[ \forall n \in \mathbb{N} \ S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
And when is identification of $L$ achieved?

$$\forall \infty n \in \mathbb{N} \ S(T[n]) = M_k$$

$S$ identifies $L$ iff $S$ identifies every text for $L$
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

And when is identification of \( L \) achieved?

\[ \forall n \in \mathbb{N} \ S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ T = u_0, u_1, \#, \#, \#, \]  

\[ \sigma \in \text{SEQ} \]

\[ \forall \infty n \in \mathbb{N} \ S(T[n]) = M_k \]

\[ S \text{ identifies } L \text{ iff } S \text{ identifies every text for } L \]
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \# \]

\[ i \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

And when is identification of \( L \) achieved?

\[ \forall \infty n \in \mathbb{N} \ S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \# , \# , \# \]

\[ i \]

\[ M_i, i \in \mathbb{N} \text{ such that } M_i \text{ accepts } L \]

And when is identification of \( L \) achieved?

\[ \forall n \in \mathbb{N} \quad S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated

And when is identification of $L$ achieved?

$$\forall \infty n \in \mathbb{N} \ S(T[n]) = M_k$$

$S$ identifies $L$ iff $S$ identifies every text for $L$
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ i \]

\[ S \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

And when is identification of \( L \) achieved?

\[ \forall \infty n \in \mathbb{N} \ S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\( \sigma \in \text{SEQ} \)

\( T = u_0, u_1, \#, \#, \#, u_2 \)

\( i \rightarrow o \)

\( M_i, i \in \mathbb{N} \)

such that \( M_i \) accepts \( L \)

And when is identification of \( L \) achieved?

\( \forall \infty n \in \mathbb{N} \quad S(T[n]) = M_k \)

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ M_i, i \in \mathbb{N} \]

such that \( M_i \) accepts \( L \)

\[ S \]

And when is identification of \( L \) achieved?

\[ \forall n \in \mathbb{N} \ S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ \sigma \in \text{SEQ} \]

Sally likes Bill and Bill likes Sally.

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

And when is identification of \( L \) achieved?

\[ \forall \infty n \in \mathbb{N} \quad S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \).
CLT-based Model Instantiated

Language of the Pure Predicate Calculus

\[ T = u_0, u_1, \#, \#, \#, u_2 \]

\[ \sigma \in \text{SEQ} \]

And when is identification of \( L \) achieved?

\[ \forall \infty \in \mathbb{N} \quad S(T[n]) = M_k \]

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
And when is identification of $L$ achieved?

$$\forall \infty n \in \mathbb{N} \ S(T[n]) = M_k$$

$S$ identifies $L$ iff $S$ identifies every text for $L$
RAIR Lab needs to work out the details and engineer an AI that can learn the language of the pure predicate calculus!
Is today’s ML mostly hype?
Is today’s ML mostly hype?

Yes …