Informal Intro to Gold-Style Language Learning —
With a Sample Target of RAIR-Lab Interest
(or: Human Language Acquisition is Hard!; Sorry Nonhuman Animals)

Selmer Bringsjord
Naveen Sundar Govindarajulu
Are Humans Rational?
RPI
11/19/18

Informal Intro to Gold-style Learning ...
Formalization of ...
Formalization of ...

1. *theoretically possible realities*;
Formalization of ...

1. *theoretically possible realities*;

2. *intelligible hypotheses* (each of which imply that a given theoretically possible reality *is* reality);
Formalization 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);
Formalization 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. *the (language, machine, system, …) learner*;
Formalization 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. the (*language, machine, system, …*) learner;

5. *successful/unsuccessful behavior by a learner trying to learn 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., $\{2, 4, 6, 8, \ldots\}$. 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.

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

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

\[ S \]

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

\[ \text{such that } M_i \text{ accepts } L \]

\[ L \text{ (r.e.)} \]
CLT-based Model of Language Learning

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

\[ S \]

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

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

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

Turing machine
CLT-based Model of Language Learning

$\sigma \in \text{SEQ}$

$T = u_0, u_1$

$i$

$L$ (r.e.)

$S$

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

such that $M_i$ accepts $L$

Turing machine
CLT-based Model of Language Learning

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

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

\[ T = u_0, u_1, \]

\[ i \]

\[ M_1 \]

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

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

Turing machine
CLT-based Model of Language Learning

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

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

\[ S \]

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

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

\[ M_1 \]

\[ o \]

Turing machine
CLT-based Model of Language Learning

$\sigma \in \text{SEQ}$

$T = u_0, u_1, \#, \#, \#$

$L$ (r.e.)

$S$

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

such that $M_i$ accepts $L$

$M_1$

0

Turing machine
CLT-based Model of Language Learning

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

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

\[ i \rightarrow S \]

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

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

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

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

\[ i \]

\[ S \]

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

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

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

Turing machine
**CLT-based Model of Language Learning**

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

\[ S \]

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

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

Turing machine
CLT-based Model of Language Learning

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

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

\[
\text{Turing machine}
\]

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

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

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

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

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

\[ i \]

\[ S \]

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

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

Turing machine
CLT-based Model of Language Learning

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

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

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

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

Turing machine
CLT-based Model of Language Learning

And when is identification of $L$ achieved?

$\sigma \in \text{SEQ}$

$T = u_0, u_1, \#, \#, \#, u_2$

$i$

$L$ (r.e.)

$S$

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

such that $M_i$ accepts $L$

$M_3$

$0$

Turing machine
CLT-based Model of Language Learning

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

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

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

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

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

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

Turing machine
CLT-based Model of Language Learning

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

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

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

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

such that \( M_i \) accepts \( \mathcal{L} \)

And when is identification of \( \mathcal{L} \) achieved?

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

\( S \) identifies \( \mathcal{L} \) iff \( S \) identifies every text for \( \mathcal{L} \)

Turing machine
RAIR-Lab-relevant Target for Learning ...
Chomsky Hierarchy of Languages

Elements of the Chomsky Hierarchy

- Recursively enumerable languages
- Recursive languages
- Context sensitive languages
- Context free languages
- Deterministic context free languages
- Regular languages
How About Learning the Grammar of the Pure Predicate Calculus?

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

AtomicFormula \Rightarrow (Predicate \ Term_1 \ldots \ Term_k)
| (Term = Term)

Term \Rightarrow (Function \ Term_1 \ldots \ Term_k)
| Constant

Connective \Rightarrow \land \mid \lor \mid \rightarrow \mid \leftrightarrow

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
How About Learning the Grammar of the Pure Predicate Calculus?

Sally likes Bill.

(Likes sally bill)

<table>
<thead>
<tr>
<th>Formula</th>
<th>⇒</th>
<th>AtomicFormula</th>
<th>(Formula Connective Formula)</th>
<th>¬ Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>AtomicFormula</td>
<td>⇒</td>
<td>(Predicate Term₁ … Termₖ)</td>
<td>(Term = Term)</td>
<td></td>
</tr>
<tr>
<td>Term</td>
<td>⇒</td>
<td>(Function Term₁ … Termₖ)</td>
<td>Constant</td>
<td></td>
</tr>
<tr>
<td>Connective</td>
<td>⇒</td>
<td>∧</td>
<td>∨</td>
<td>→</td>
</tr>
<tr>
<td>Predicate</td>
<td>⇒</td>
<td>P₁</td>
<td>P₂</td>
<td>P₃</td>
</tr>
<tr>
<td>Constant</td>
<td>⇒</td>
<td>c₁</td>
<td>c₂</td>
<td>c₃</td>
</tr>
<tr>
<td>Function</td>
<td>⇒</td>
<td>f₁</td>
<td>f₂</td>
<td>f₃</td>
</tr>
</tbody>
</table>
How About Learning the Grammar of the Pure Predicate Calculus?

<table>
<thead>
<tr>
<th>Formula</th>
<th>⇒</th>
<th>AtomicFormula</th>
<th>Sally likes Bill.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(Formula Connective Formula)</td>
<td>(Likes sally bill)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>¬ Formula</td>
<td></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       | ⇒ | ∧ | ∨ | → | ↔ |

| Predicate        | ⇒ | P₁ | P₂ | P₃ | ... |
| Constant         | ⇒ | c₁ | c₂ | c₃ | ... |
| Function         | ⇒ | f₁ | f₂ | f₃ | ... |

Lexicon
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       | ⇒ | ∧ | ∨ | → | ↔ |

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

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

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

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

Connective \[ \Rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftarrow \]

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

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

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

Term \( \Rightarrow \) \( (\text{Function } \text{Term}_1 \ldots \text{Term}_k) \)  
| Constant

Connective \( \Rightarrow \) \( \land \mid \lor \mid \rightarrow \mid \leftrightarrow \)

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

<table>
<thead>
<tr>
<th>Formula</th>
<th>$\Rightarrow$</th>
<th>AtomicFormula</th>
<th>(Formula Connective Formula)</th>
<th>$\neg$ Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>AtomicFormula</td>
<td>$\Rightarrow$</td>
<td>(Predicate Term$_1$ ... Term$_k$)</td>
<td>(Term = Term)</td>
<td></td>
</tr>
<tr>
<td>Term</td>
<td>$\Rightarrow$</td>
<td>(Function Term$_1$ ... Term$_k$)</td>
<td>Constant</td>
<td></td>
</tr>
<tr>
<td>Connective</td>
<td>$\Rightarrow$</td>
<td>$\land$</td>
<td>$\lor$</td>
<td>$\rightarrow$</td>
</tr>
<tr>
<td>Predicate</td>
<td>$\Rightarrow$</td>
<td>$P_1$</td>
<td>$P_2$</td>
<td>$P_3$ ...</td>
</tr>
<tr>
<td>Constant</td>
<td>$\Rightarrow$</td>
<td>$c_1$</td>
<td>$c_2$</td>
<td>$c_3$ ...</td>
</tr>
<tr>
<td>Function</td>
<td>$\Rightarrow$</td>
<td>$f_1$</td>
<td>$f_2$</td>
<td>$f_3$ ...</td>
</tr>
</tbody>
</table>

Lexicon

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

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

\[\text{AtomicFormula} \Rightarrow (\text{Predicate Term}_1 \ldots \text{Term}_k) \]
\[\text{AtomicFormula} \Rightarrow (\text{Term} = \text{Term}) \]

\[\text{Term} \Rightarrow (\text{Function Term}_1 \ldots \text{Term}_k) \]
\[\text{Term} \Rightarrow \text{Constant} \]

\[\text{Connective} \Rightarrow \land | \lor | \to | \equiv \]

\[\text{Likes Predicate} \Rightarrow P_1 | P_2 | P_3 \ldots \]
\[\text{Likes Constant} \Rightarrow c_1 | c_2 | c_3 \ldots \]
\[\text{Likes Function} \Rightarrow f_1 | f_2 | f_3 \ldots \]
How About Learning the Grammar of the Pure Predicate Calculus?

Formula \[ \Rightarrow \]

- AtomicFormula
- \((\text{Formula Connective Formula})\)
- \(\neg \text{Formula}\)

AtomicFormula \[ \Rightarrow \]

- \((\text{Predicate Term}_1 \ldots \text{Term}_k)\)
- \((\text{Term} = \text{Term})\)

Term \[ \Rightarrow \]

- \((\text{Function Term}_1 \ldots \text{Term}_k)\)
- Constant

Connective \[ \Rightarrow \]

- \(\land\)
- \(\lor\)
- \(\rightarrow\)
- \(\leftrightarrow\)

Lexicon

- Likes
- Predicate
  \[ \Rightarrow \]
  \(P_1 \mid P_2 \mid P_3 \ldots\)
- Constant
  \[ \Rightarrow \]
  \(c_1 \mid c_2 \mid c_3 \ldots\)
- Function
  \[ \Rightarrow \]
  \(f_1 \mid f_2 \mid f_3 \ldots\)
How About Learning the Grammar of the Pure Predicate Calculus?

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

AtomicFormula \implies (Predicate Term_1 \ldots Term_k)
| (Term = Term)

Term \implies (Function Term_1 \ldots Term_k)
| Constant

Connective \implies \land | \lor | \rightarrow | \leftrightarrow

Lexicon

Sally likes Bill.
(Likes sally bill)

Sally likes Bill and Bill likes Sally.
Sally likes Bill’s mother.
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.
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.

Matilda is Bill’s super-smart mother.

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

<table>
<thead>
<tr>
<th>Formula</th>
<th>AtomicFormula</th>
<th>(Formula Connective Formula)</th>
<th>¬ Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>AtomicFormula</td>
<td>(Predicate Term₁ ... Termₖ)</td>
<td>(Term = Term)</td>
<td></td>
</tr>
<tr>
<td>Term</td>
<td>(Function Term₁ ... Termₖ)</td>
<td>Constant</td>
<td></td>
</tr>
<tr>
<td>Connective</td>
<td>∧</td>
<td>∨</td>
<td>→</td>
</tr>
</tbody>
</table>

**Lexicon**

- **Likes**
  - Predicate: \( P_1 | P_2 | P_3 \ldots \)
  - Constant: \( c_1 | c_2 | c_3 \ldots \)
  - Function: \( f_1 | f_2 | f_3 \ldots \)

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

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

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

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

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

<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>

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

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

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

| Likes Predicate | ⇒ | P₁ | P₂ | P₃ | ... |
| Constant        | ⇒ | c₁ | c₂ | c₃ | ... |
| Function        | ⇒ | f₁ | f₂ | f₃ | ... |

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

$\sigma \in \text{SEQ}$

$S$

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

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

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

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

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

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

The 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?

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

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

Language of the Pure Predicate Calculus

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

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

\[ i \]

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

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

Language of the Pure Predicate Calculus

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

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

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?

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

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

Language of the Pure Predicate Calculus

$\sigma \in \text{SEQ}$

$T = u_0, u_1, \#, \#, \#, u_2$

$i$

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

$M_2$

$0$

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

$\sigma \in \text{SEQ}$

$T = u_0, u_1, \#, \#, \#, u_2$

$i$

$S$

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

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

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

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

\( i \)

\( S \)

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

\( M_3 \)

\( o \)

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

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

Sally likes Bill and Bill likes Sally.

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

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

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

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)

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

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

\( M_3 \)

0
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?

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

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

Language of the Pure Predicate Calculus

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

Matilda is Bill's super-smart mother.

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

\[ i \]

\[ M_3 \]

\[ 0 \]

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

\[ S \text{ identifies } L \text{ iff } S \text{ identifies every text for } L \]
Language of the Pure Predicate Calculus, and Its Inference Schemata

Let $\sigma \in \text{SEQ}$ and $T = u_0, u_1$, $i \in \mathbb{N}$, 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 to Target Deductive Calculi
CLT-based Model Instantiated to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

\[ T = u_0, u_1, \]

\[ i \]

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

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

\[ S \]

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

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

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

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

\( i \) 

\( S \) 

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

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

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

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
CLT-based Model Instantiated to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

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

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

\[ M_i, i \in \mathbb{N} \] 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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

\[ i \]

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

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

\[ i \]

\[ S \]

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

And when is identification of } L \text{ achieved?

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

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

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

Sally likes Bill and Bill likes Sally.

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

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

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

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

\[ i \]

\[ S \]

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

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

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

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

\( S \) identifies \( L \) iff \( S \) identifies every text for \( L \)
Matilda is Bill’s super-smart mother.

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 to Target Deductive Calculi

Language of the Pure Predicate Calculus, and Its Inference Schemata

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

Matilda is Bill's super-smart mother.

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

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 \( L \)

Sally likes Bill and Bill likes Sally.
Bill likes Sally.

\[ \frac{Lsb \land Lbs}{Lbs} \]
RAIR Lab needs to work out the details and engineer an AI that can learn the language of the pure predicate calculus!