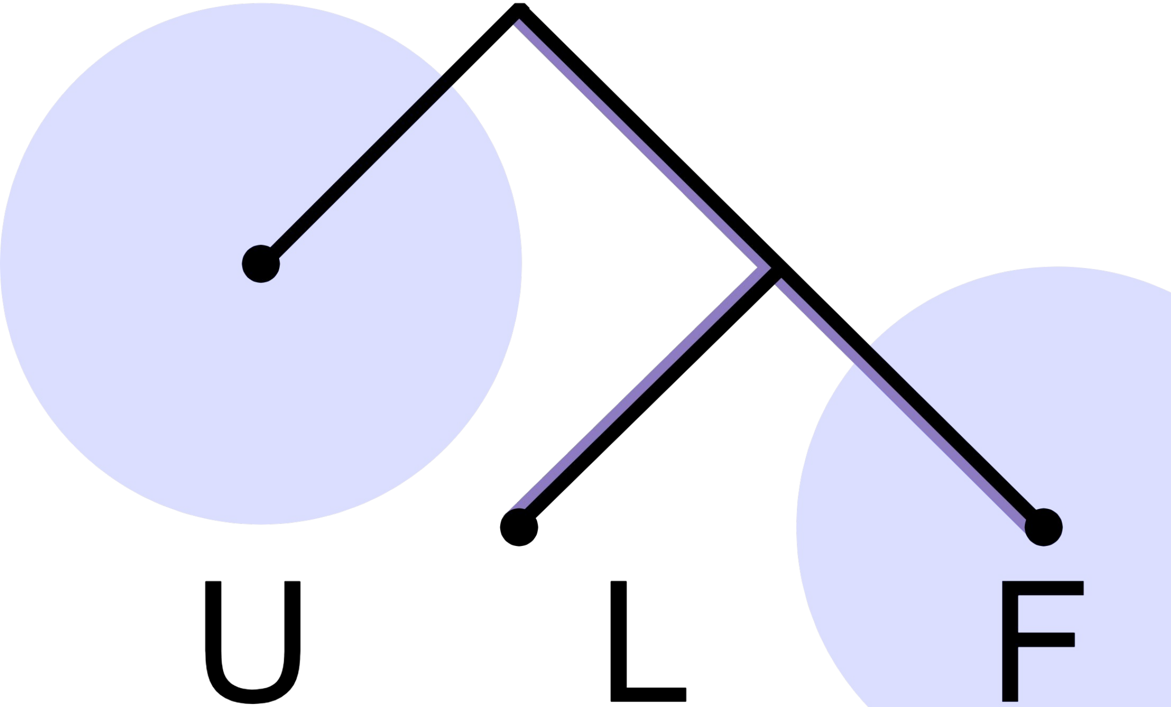
Corpus Annotation, Parsing, and Inference for Episodic Logic Type Structure

Gene Louis Kim
University of South Florida

ILFC

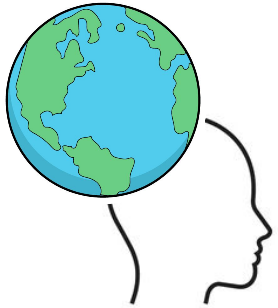
14 June 2022



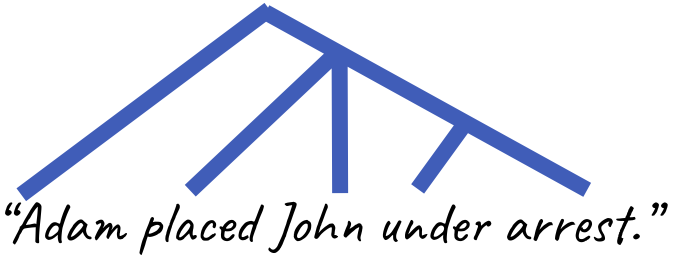
What does it mean to understand language?

AI-complete?

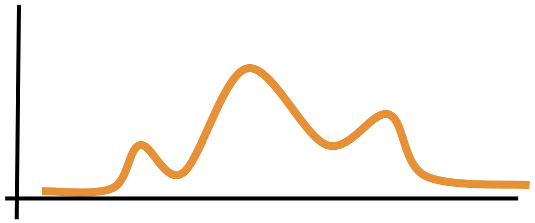
To understand language



1. parse the structure
2. relate to world knowledge
3. consider the participants



Distributional



or

Symbolic

Abc

x = y Ω

Feature of Symbolic Systems

Effect of single interactions on

- complex plans
- model of the world

Major systematic change

Requires modeling of precise relationships

Interface for world model & communicative intent

→ **Language Meaning** (Bender & Koller 2020)

Symbols for Language Meaning

Shared across languages: purpose + human cognition

• truth/falsity	FOL
• predicates	
• identity	
• generalized quantifiers	most, few, many, no, at most 10
• modification	very, gracefully, nearly, possibly
• reification	Beauty is subjective. That exoplanets exist is now certain.
• event reference	Many children had not been vaccinated against measles; this situation caused sporadic outbreaks of the disease.
• comparatives	Doorways are taller than most people

Proposal

Bridge the gap with

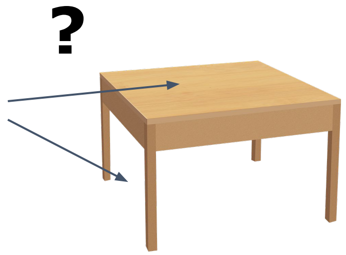
<u>a type system</u>

+

<u>ambiguity</u>

$$\underset{\text{“Spot”}}{D} + \underset{\text{“runs”}}{(D \rightarrow T)} = \overset{\text{“Spot runs”}}{T}$$

“there”



Unscoped Episodic Logical Forms (ULF)

Underspecified Expressive Logic

ULF Parsing

Neural Model Over a Transition System

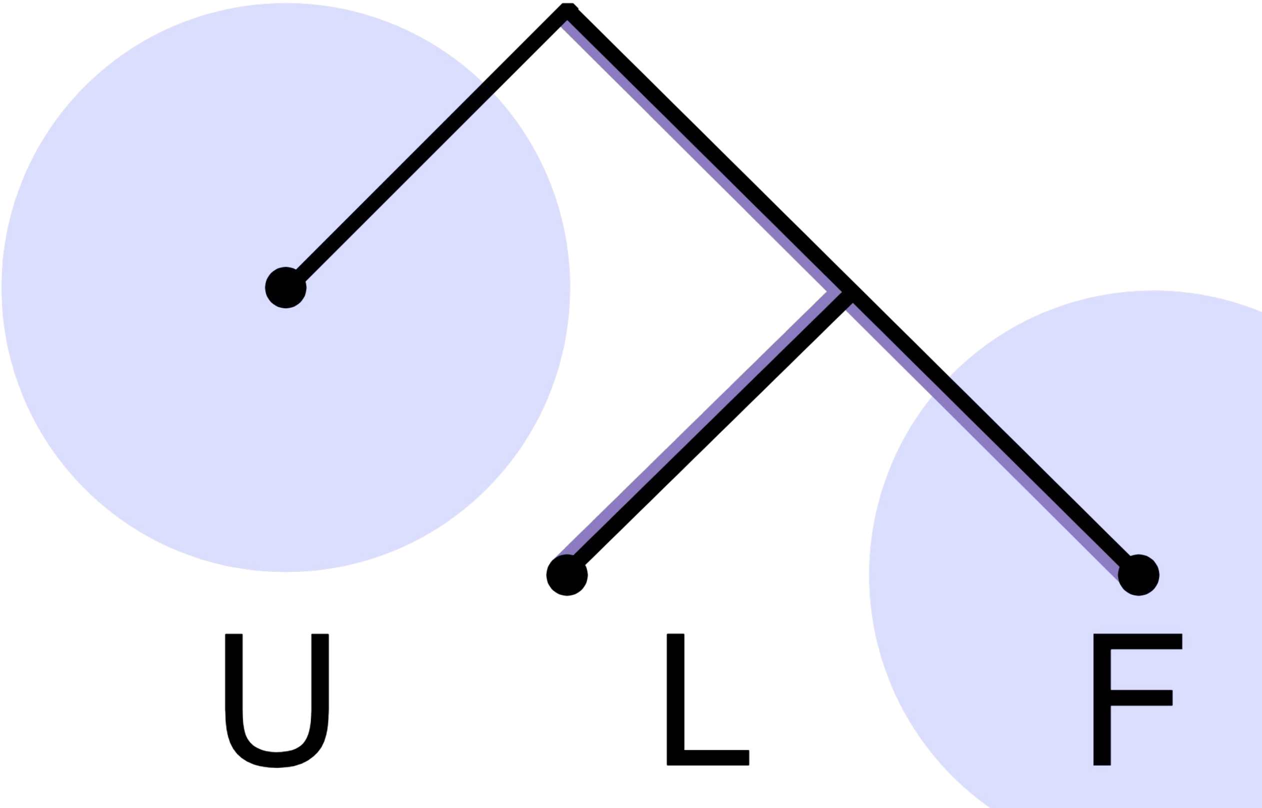
ULF Inference

Pragmatic Discourse and Natural Logic

Wider Use of ULF

Spatial Reasoning Agent & Schema Learning

Design of ULF



Episodic Logic (EL)

- Extended FOL
- Closely matches expressivity of natural languages
    - Predicates, connectives, quantifiers, equality
    - Predicate and sentence modification
    - Predicate and sentence reification
    - Generalized quantifiers
    - Intensional predicates
    - Reference to events and situations

EL Inference

- Suitable for deductive and uncertain inference
- EPILOG for fast and comprehensive theorem proving

Morbini and Schubert, 2009; Schubert and Hwang, 2000; Schubert, 2014

How hard is it to annotate and parse Episodic Logic?

"I want to dance in my new shoes"

Episodic Logic

$$
\begin{array}{l}
(\exists e\text{: } [e \text{ at-about Now}] \\
\qquad [[\text{Gene want1.v} \\
\qquad\qquad (\text{ka } (\lambda x\text{: } [[x \text{ dance1.v}] \wedge \\
\qquad\qquad\qquad (\iota y\text{: } [[y \text{ shoes.n}] \wedge \\
\qquad\qquad\qquad\quad [y \text{ poss-by Gene}] \wedge \\
\qquad\qquad\qquad\quad [x \text{ in-wear } y]])]))] \; ** \; e])
\end{array}
$$

Errors for **1 in 3** verb definitions! (Kim and Schubert, 2016)

What if we leave things that are ambiguous without context?

“I want to dance in my new shoes”

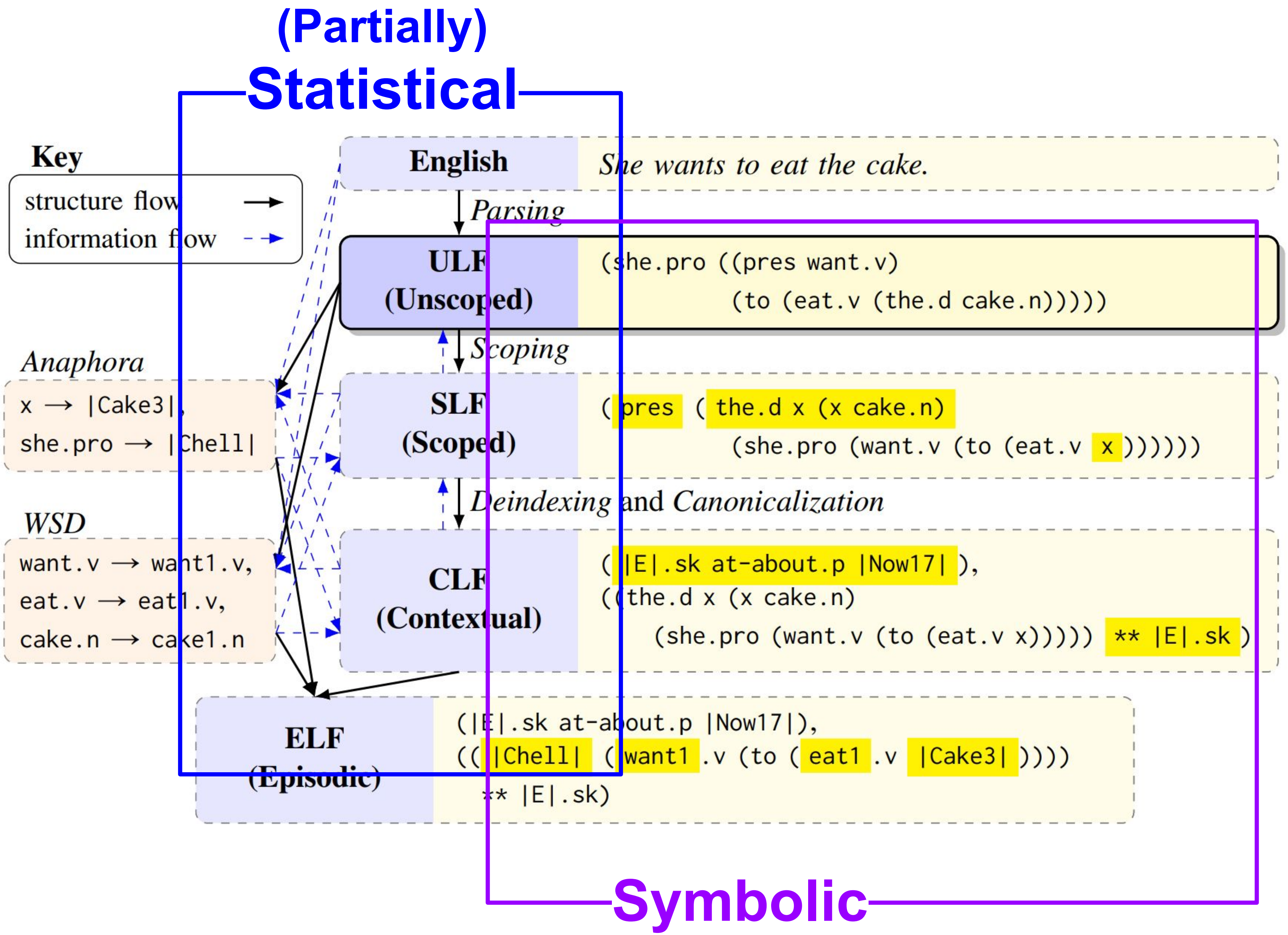
Episodic Logic

$$
\begin{array}{l}
(\exists e\text{: } [e \text{ at-about Now}] \\
\quad [[\text{Gene want1.v} \\
\qquad (\text{ka } (\lambda x\text{: } [[x \text{ dance1.v}] \wedge \\
\qquad\quad (\iota y\text{: } [[y \text{ shoes.n}] \wedge \\
\qquad\qquad [y \text{ poss-by Gene}] \wedge \\
\qquad\qquad [x \text{ in-wear } y]])]))] ** e])
\end{array}
$$

Unscoped Logical Form

```
(i.pro ((pres want.v)
        (to (dance.v
             (adv-a (in.p (my.d ((mod-n new.a)
                                  (plur shoe.n))
```

1. **Retain ambiguity of**
   a. *scopes*
   b. *word sense*
   c. *anaphora*
   d. *event relations*
2. **Maintain semantic coherence**
3. **Reflect syntactic structure**



(Partially) Statistical

Symbolic

Key

structure flow →

information flow - - →

Anaphora

x → |Cake3|,

she.pro → |Chell|

WSD

want.v → want1.v,

eat.v → eat1.v,

cake.n → cake1.n

English

She wants to eat the cake.

Parsing

ULF (Unscoped)

```
(she.pro ((pres want.v)
          (to (eat.v (the.d cake.n)))))
```

Scoping

SLF (Scoped)

```
(pres (the.d x (x cake.n)
          (she.pro (want.v (to (eat.v x))))))
```

Deindexing and Canonicalization

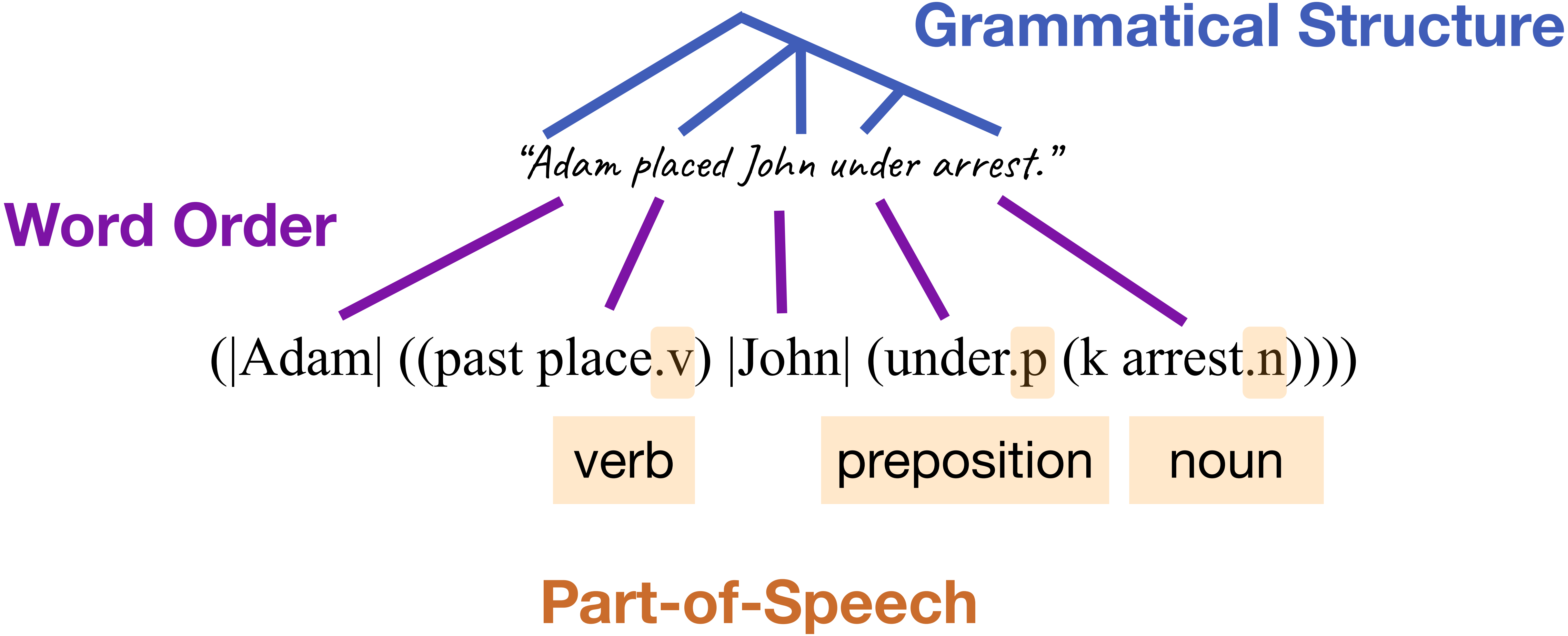
CLF (Contextual)

```
(|E|.sk at-about.p |Now17|),
((the.d x (x cake.n)
     (she.pro (want.v (to (eat.v x))))) ** |E|.sk)
```

ELF (Episodic)

```
(|E|.sk at-about.p |Now17|),
((|Chell| (want1.v (to (eat1.v |Cake3|))))
  ** |E|.sk)
```

ULF & Syntax



Grammatical Structure

“Adam placed John under arrest.”

Word Order

(|Adam| ((past place.v) |John| (under.p (k arrest.n))))

verb

preposition

noun

Part-of-Speech

ULF & Semantics

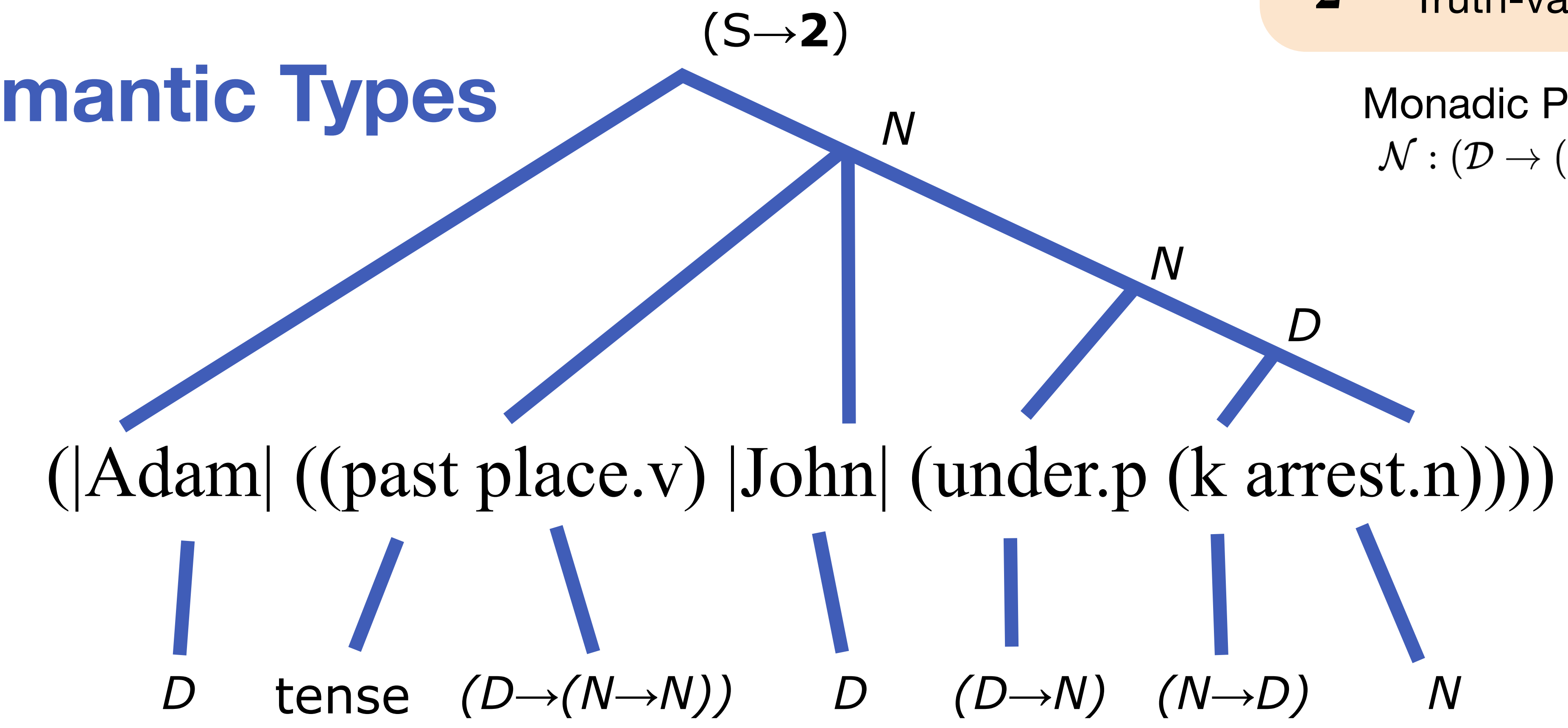
Basic Ontological Types

$\mathcal{D}$	Domain
$\mathcal{S}$	Situations
$\mathbf{2}$	Truth-value

Semantic Types

Monadic Predicate

$\mathcal{N} : (\mathcal{D} \rightarrow (\mathcal{S} \rightarrow \mathbf{2}))$



ULF & Semantics

Basic Ontological Types

$\mathcal{D}$	Domain
$\mathcal{S}$	Situations
$\mathbf{2}$	Truth-value

Monadic Predicate

$\mathcal{N} : (\mathcal{D} \rightarrow (\mathcal{S} \rightarrow \mathbf{2}))$

"Alice thinks that John nearly fell"

```
(|Alice| (((pres think.v)
          (that (|John| (nearly.adv-a (past fall.v)))))))
```

"You made the order for me"

```
(you.pro ((past make.v) (the.d order.n) (adv-a (for.p me.pro))))
```

Determiner $(\mathcal{N} \rightarrow \mathcal{D})$: `the.d`

Modifier Constructor $(\mathcal{N} \rightarrow (\mathcal{N} \rightarrow \mathcal{N}))$: `adv-a`

Predicate modifier $(\mathcal{N} \rightarrow \mathcal{N})$: `nearly.adv-a`

Sentence reifier $((\mathcal{S} \rightarrow \mathbf{2}) \rightarrow \mathcal{D})$: `that`

“I want to dance in my new shoes”

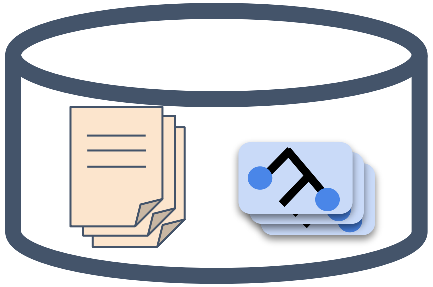
Episodic Logic

$$
\begin{array}{l}
(\exists e\!: [e \text{ at-about Now}] \\
\quad [[\text{Gene want1.v} \\
\qquad (\text{ka } (\lambda x\!: [[x \text{ dance1.v}] \wedge \\
\qquad\qquad (\iota y\!: [[y \text{ shoes.n}] \wedge \\
\qquad\qquad\quad [y \text{ poss-by Gene}] \wedge \\
\qquad\qquad\quad [x \text{ in-wear } y]])]))] ** e])
\end{array}
$$

→

Unscoped Logical Form

```
(i.pro ((pres want.v)
        (to (dance.v
             (adv-a (in.p (my.d ((mod-n new.a)
                                 (plur shoe.n))
```



Dataset

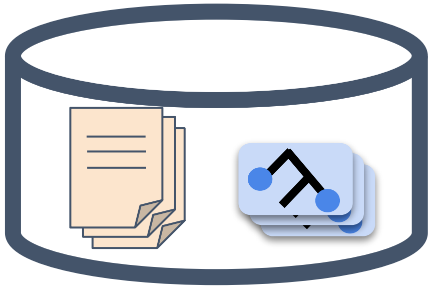
“I want to dance in my new shoes”

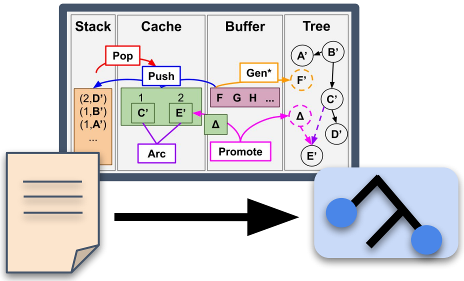
Episodic Logic

$$
\begin{array}{l}
(\exists e\text{: } [e \text{ at-about Now}] \\
\quad [[\text{Gene want1.v} \\
\qquad (\text{ka } (\lambda x\text{: } [[x \text{ dance1.v}] \wedge \\
\qquad\quad (\iota y\text{: } [[y \text{ shoes.n}] \wedge \\
\qquad\qquad [y \text{ poss-by Gene}] \wedge \\
\qquad\qquad [x \text{ in-wear } y]])]))] ** e])
\end{array}
$$

Unscoped Logical Form

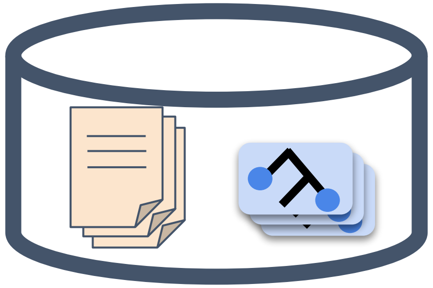
```
(i.pro ((pres want.v)
        (to (dance.v
             (adv-a (in.p (my.d ((mod-n new.a)
                                  (plur shoe.n))
```

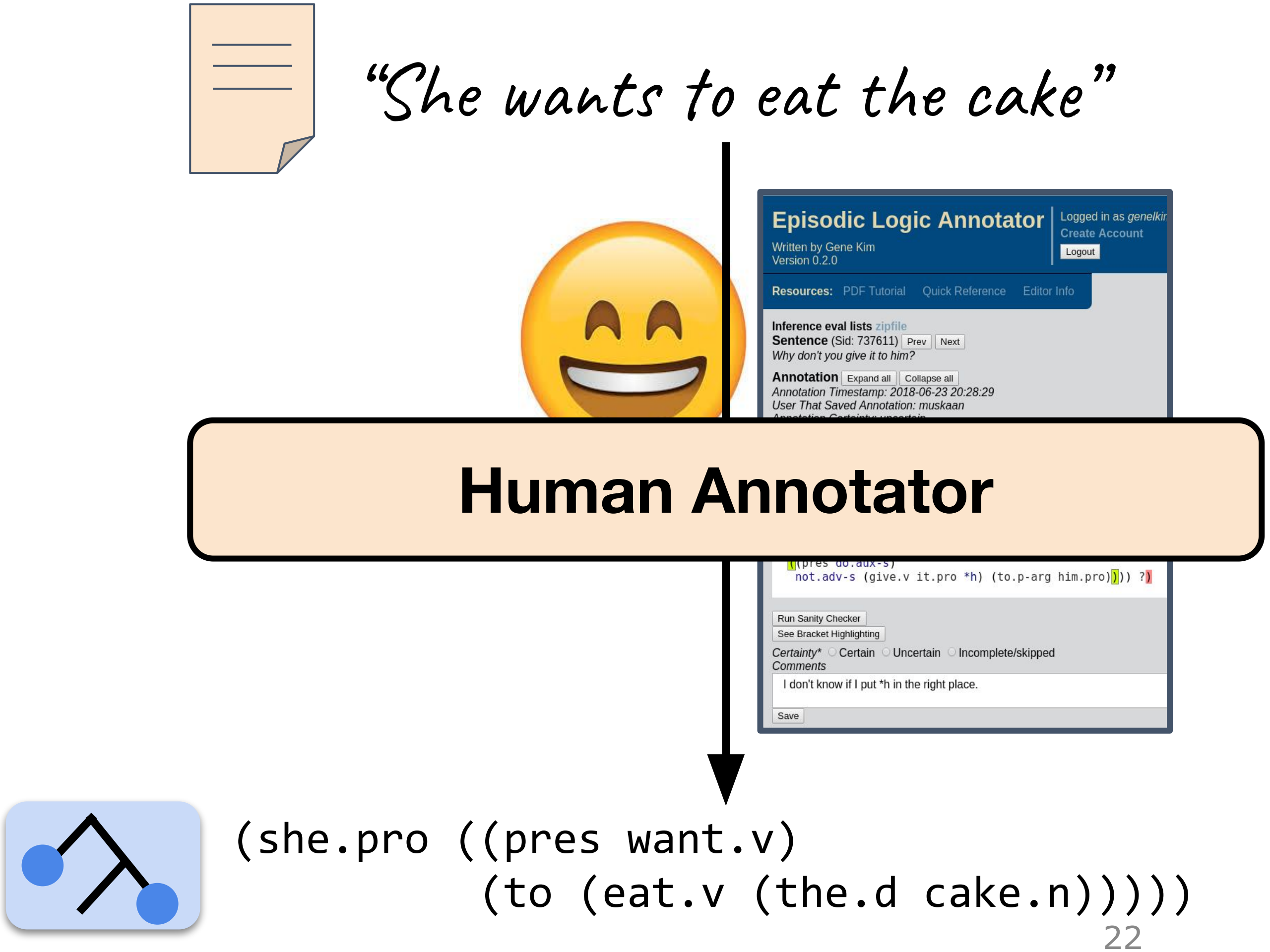




Dataset & Parser

Dataset Annotation





"She wants to eat the cake"

Episodic Logic Annotator

Written by Gene Kim
Version 0.2.0

Logged in as *genelkir*
Create Account
Logout

Resources: PDF Tutorial Quick Reference Editor Info

Inference eval lists zipfile
Sentence (Sid: 737611) Prev Next
Why don't you give it to him?

Annotation Expand all Collapse all
Annotation Timestamp: 2018-06-23 20:28:29
User That Saved Annotation: muskaan

```
((pres do.aux-s)
 not.adv-s (give.v it.pro *h) (to.p-arg him.pro)))) ?)
```

Run Sanity Checker
See Bracket Highlighting
*Certainty** ○ Certain ○ Uncertain ○ Incomplete/skipped
Comments
I don't know if I put *h in the right place.
Save

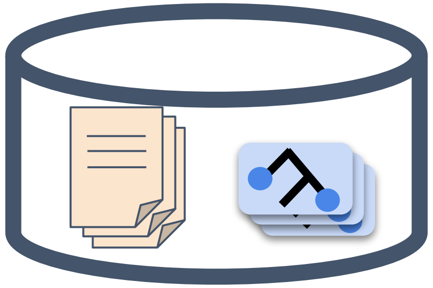
Human Annotator

```
(she.pro ((pres want.v)
          (to (eat.v (the.d cake.n)))))
```

Human ULF annotations

- are *fast*
  (~8 min/sent)
- are *consistent*
  (up to 0.88 IAA)

Data (ULF Release)



1,738 sentences

Trained student annotators

+

Reviewed by an expert annotator

Text Sources

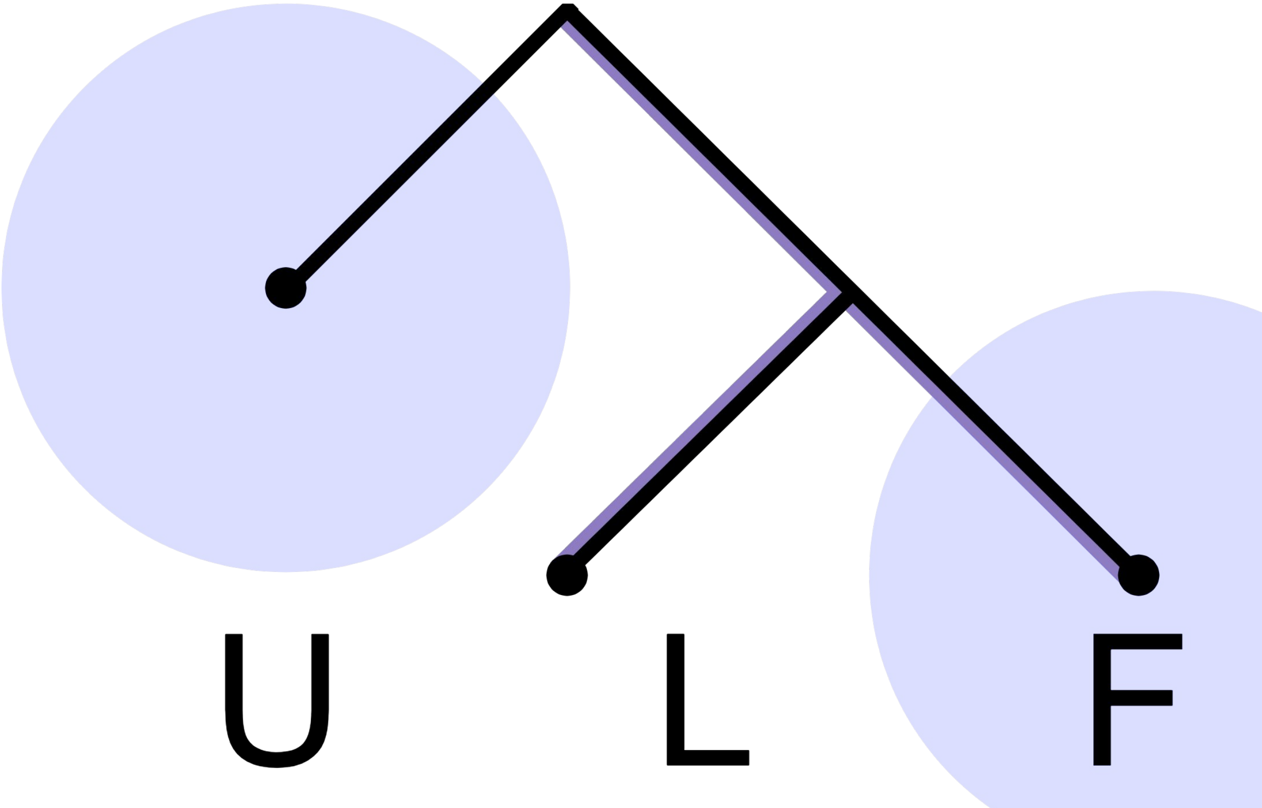
Tatoeba (crowd-sourced translations)

Project Gutenberg (100 most popular)

Discourse Graphbank (WSJ subset) [Wolf, 2005]

UIUC Question Classification [Li & Roth, 2002]

Parsing into ULF



Viet Duong
UR

Xin (Lucy) Lu
Stanford
(formerly UR)

Lenhart Schubert
UR

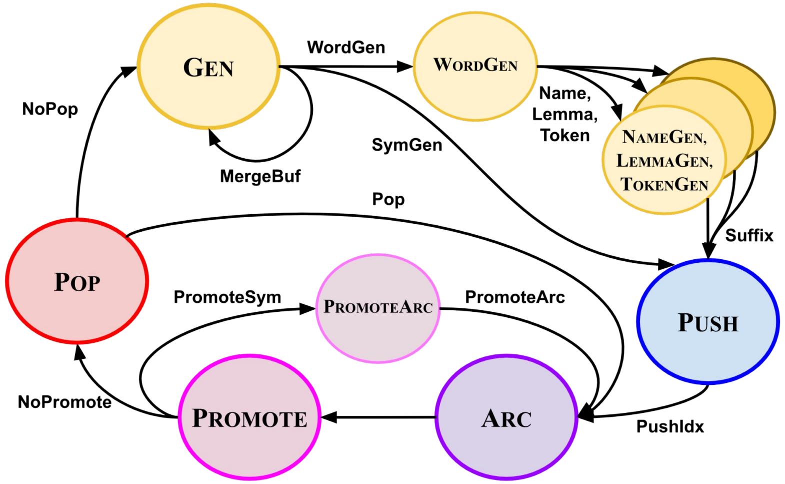
Can we actually learn a parser from English to ULF?

Challenge

Relatively modest dataset size

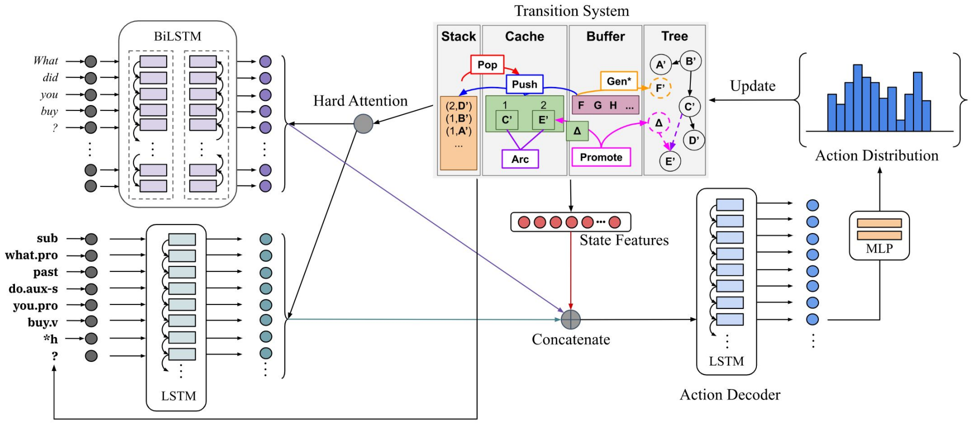
Parser Design

ULF-oriented transition system



+

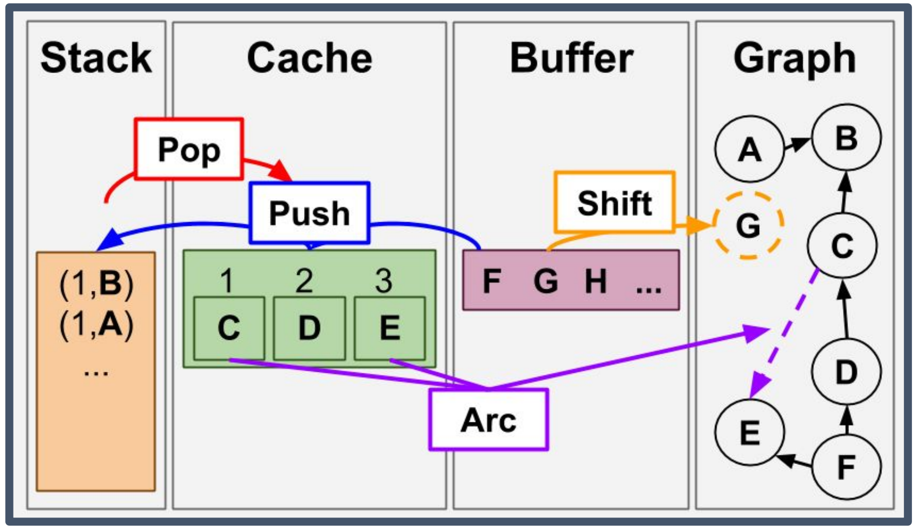
Neural action selector

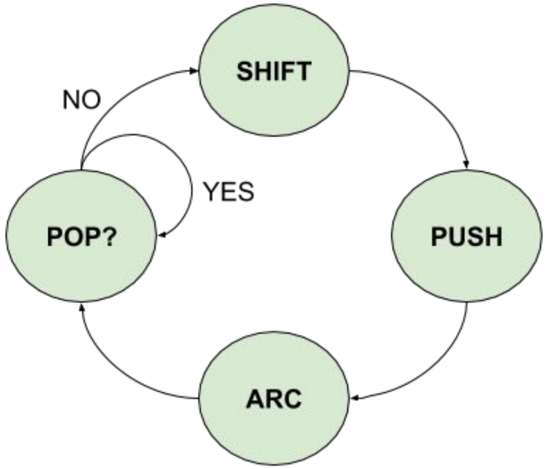


Cache Transition System

Initialize with empty stack & cache, buffer of node labels

1. **Shift:** add buffer node to graph
2. **Push:** insert shifted node to cache (move prior one to stack)
3. **Arc:** make edges in cache
4. **Pop:** remove rightmost cache element (move elements to right)

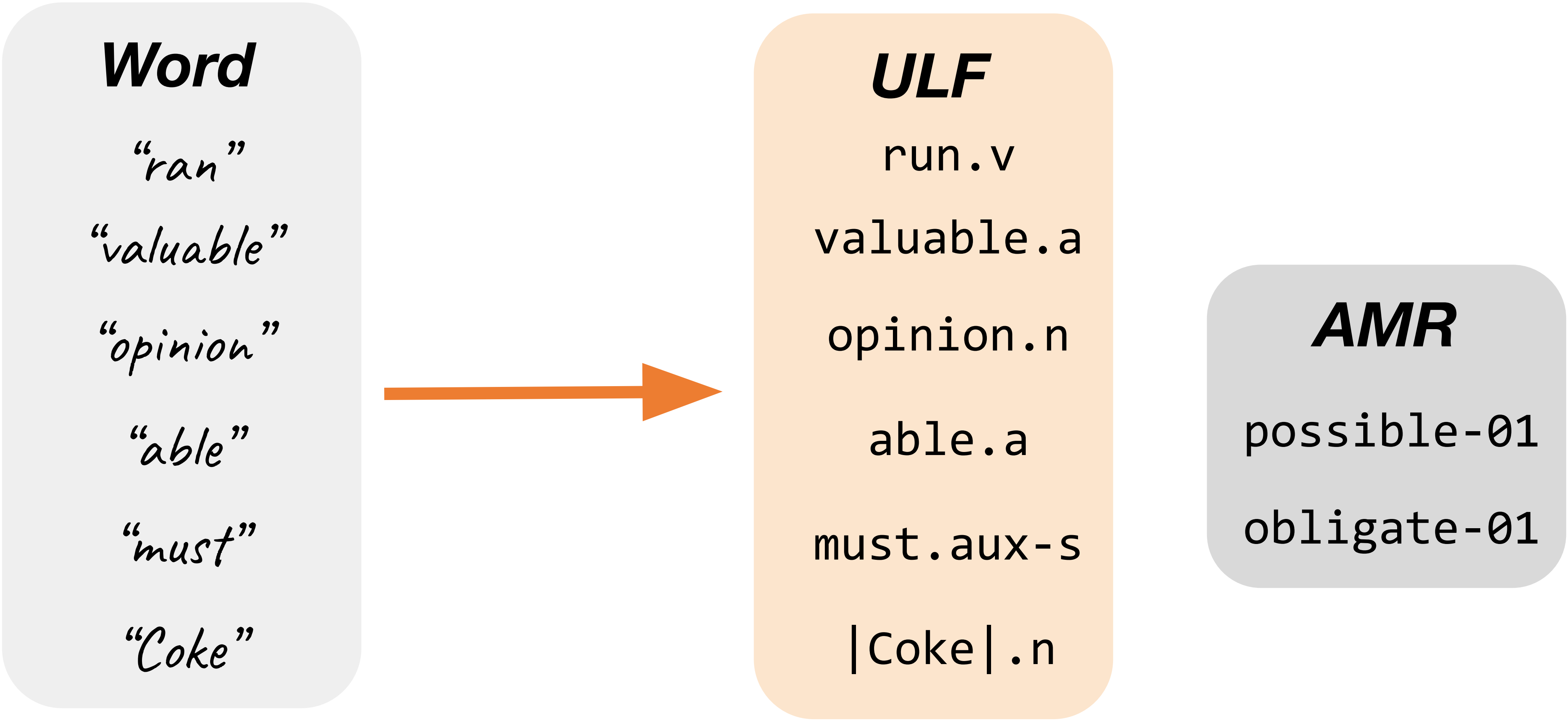




How do we tailor this to ULF?

Node label regularity

Word-based Node Labels



Word

"ran"

"valuable"

"opinion"

"able"

"must"

"Coke"

ULF

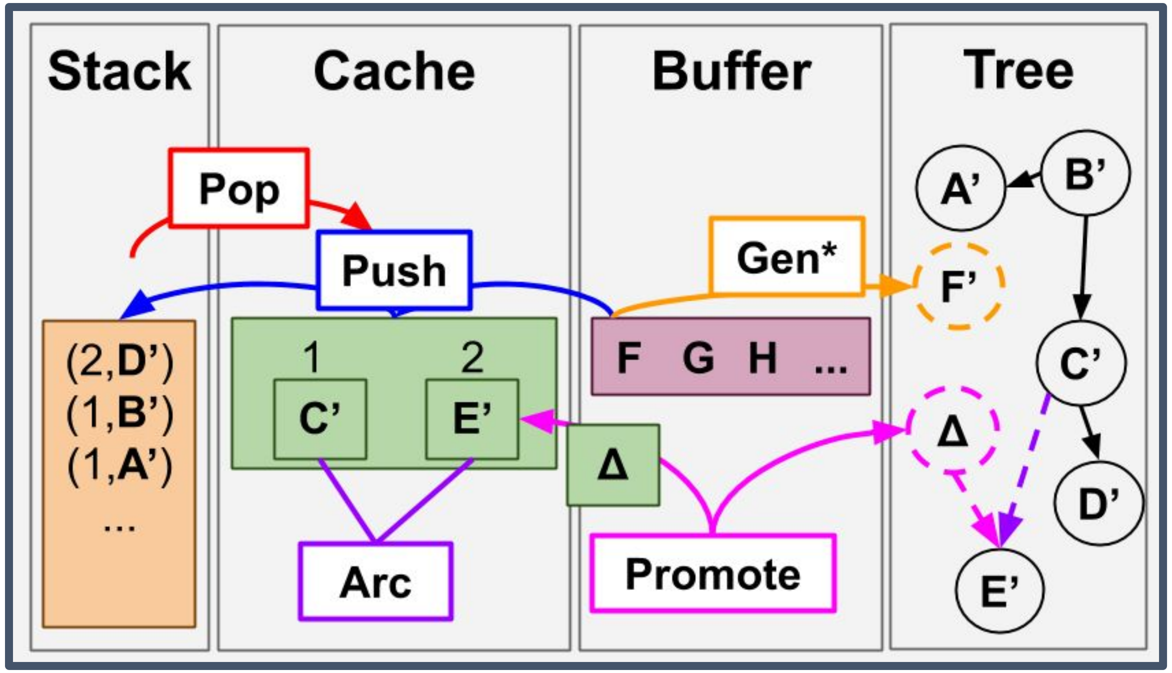
```
run.v
valuable.a
opinion.n
able.a
must.aux-s
|Coke|.n
```

AMR

```
possible-01
obligate-01
```

Structure-based Node Labels

Type-shifter		***Operand***	
k		noun predicates	(**k** gold.n)
ka		verb predicates	(**ka** (run.v quickly.adv-a))
	operates on		
that		sentences	(**that** (i.pro (past win.v)))
adv-a		any predicates	(**adv-a** (for.p you.pro))



Gen*

“ran” $\xrightarrow{.v}$ run.v

“refreshed” $\xrightarrow{.a}$ refreshed.a

“Coke” $\xrightarrow{||.n}$ |Coke|.n

Promote

dog.n $\xrightarrow{k}$ (k dog.n)

quick.a $\xrightarrow{adv-a}$ (adv-a quick.a)

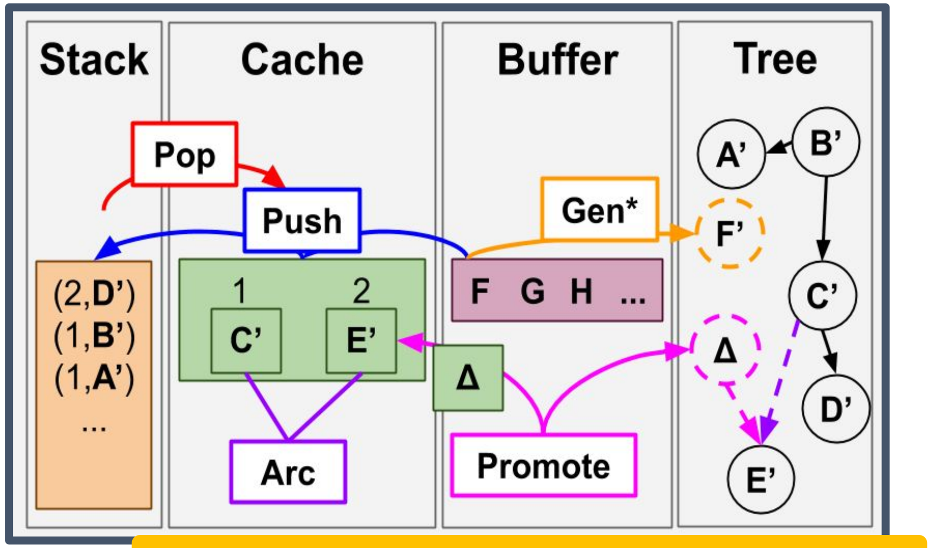
(i.pro (past win.v))

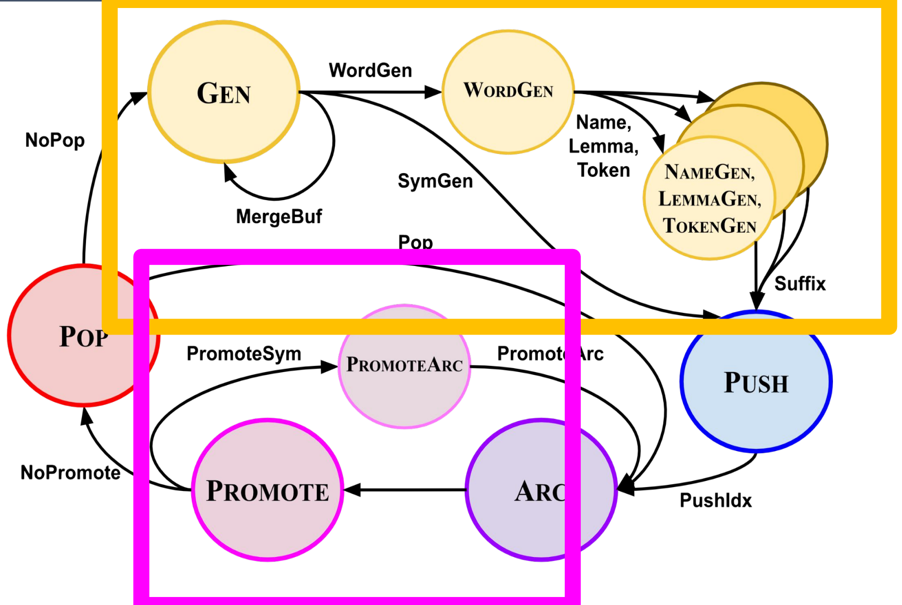
$\xrightarrow{that}$ (that (i.pro (past win.v)))

Transition System Procedure

Initialize with empty stack & cache, buffer of *word* sequence.

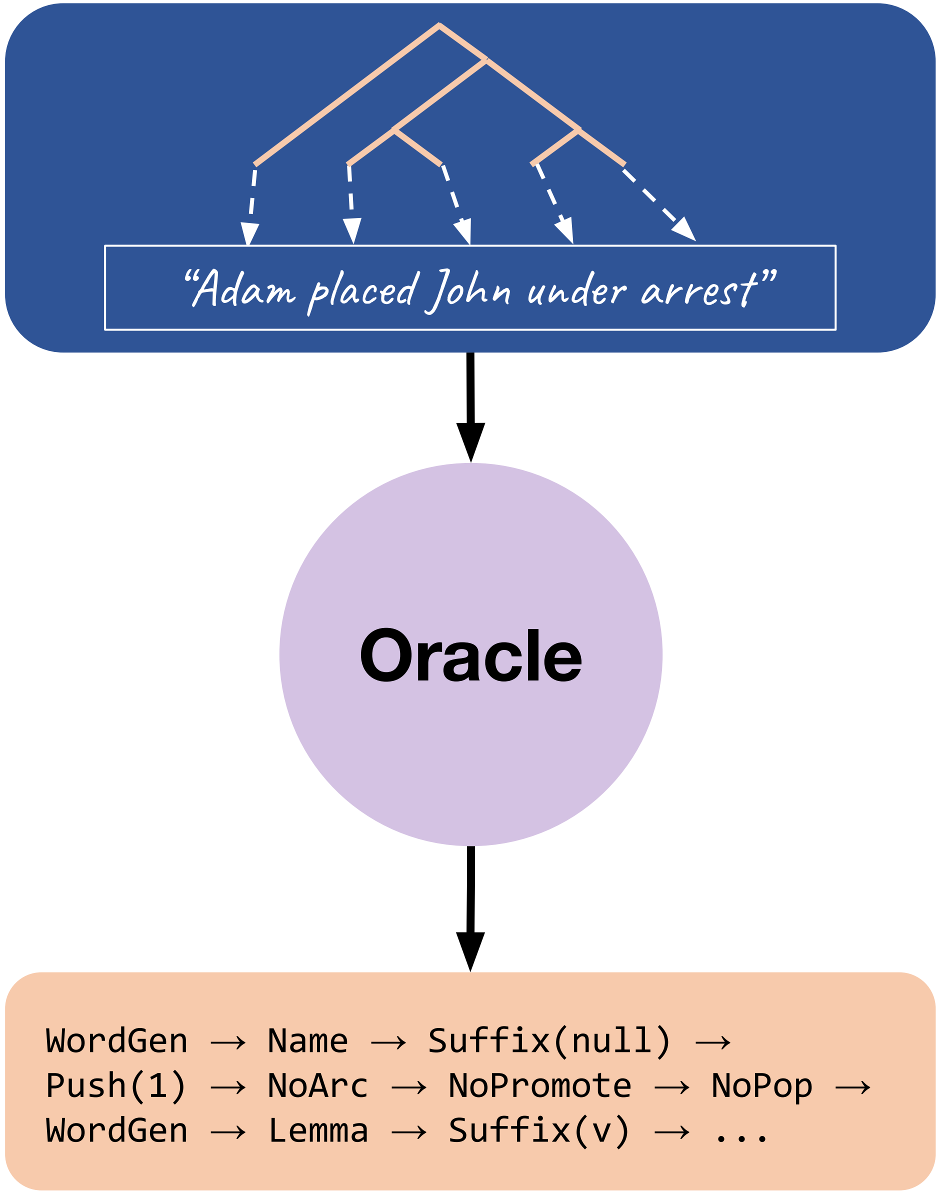
1. **Gen:** generate a symbol and add to tree
2. **Push:** insert gen'd node to cache
3. **Arc:** make edges in cache
4. **Promote:** type-shift rightmost cache element
5. **Pop:** remove rightmost cache element (move elements to right)





How do we train an action selector?

Labeled ULF + Alignment



```
WordGen → Name → Suffix(null) →
Push(1) → NoArc → NoPromote → NoPop →
WordGen → Lemma → Suffix(v) → ...
```

Parsing Action Sequence

Oracle

Gen & Arc

Greedy symbol and edge generation while tracking word-symbol alignment

Skip words if their alignment is earlier than predicted

Push

Choose the cache index whose closest edge or path including only promoted symbols into buffer is farthest away

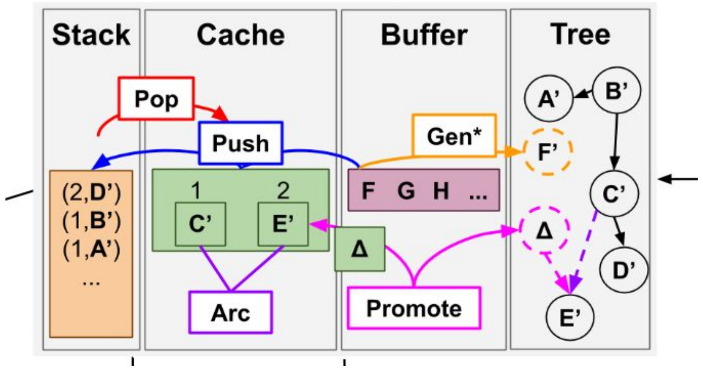
Unaligned symbols may be generated via promote

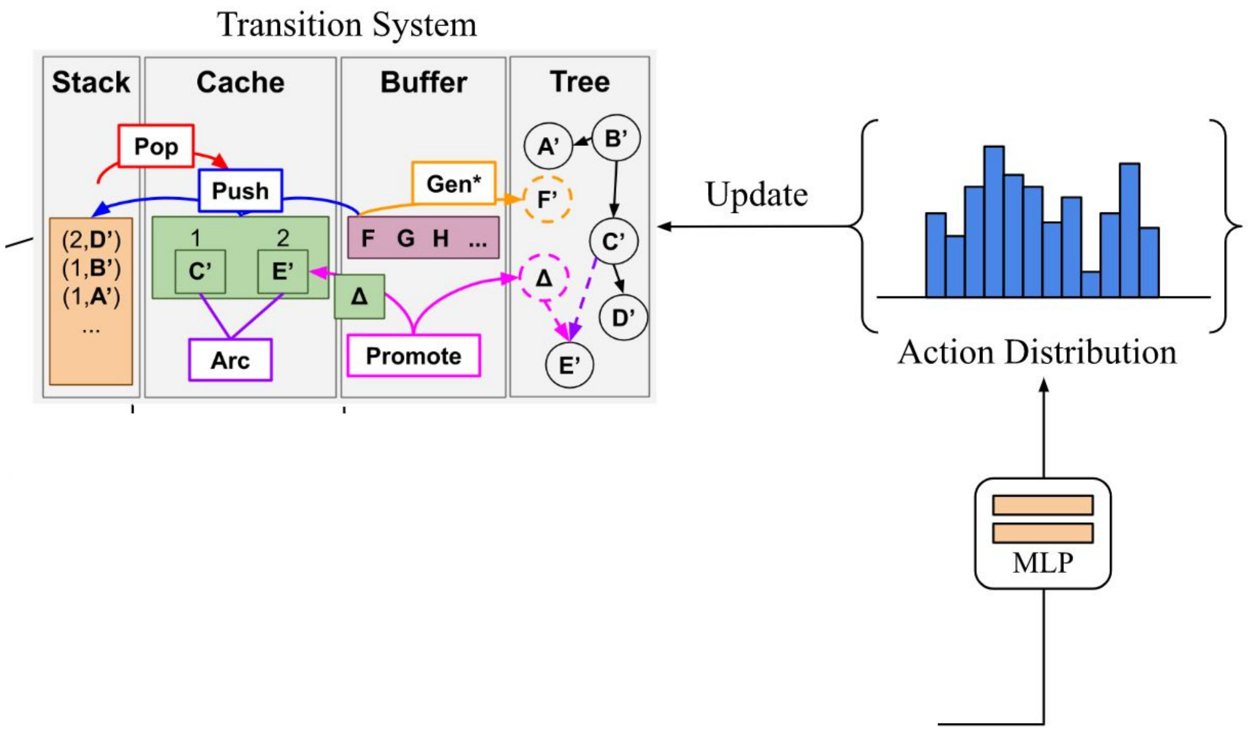
Promote

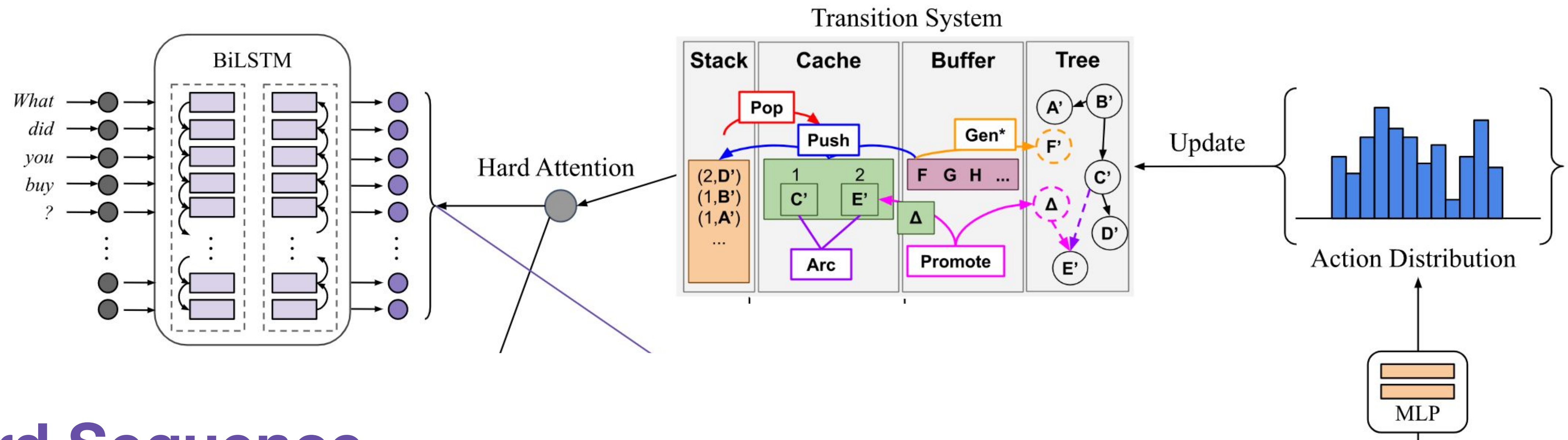
If promoted gold edge exists to rightmost cache item and child is fully formed, add it.

Bottom-up enforced for Promote & Type Constrained Decoding

Transition System

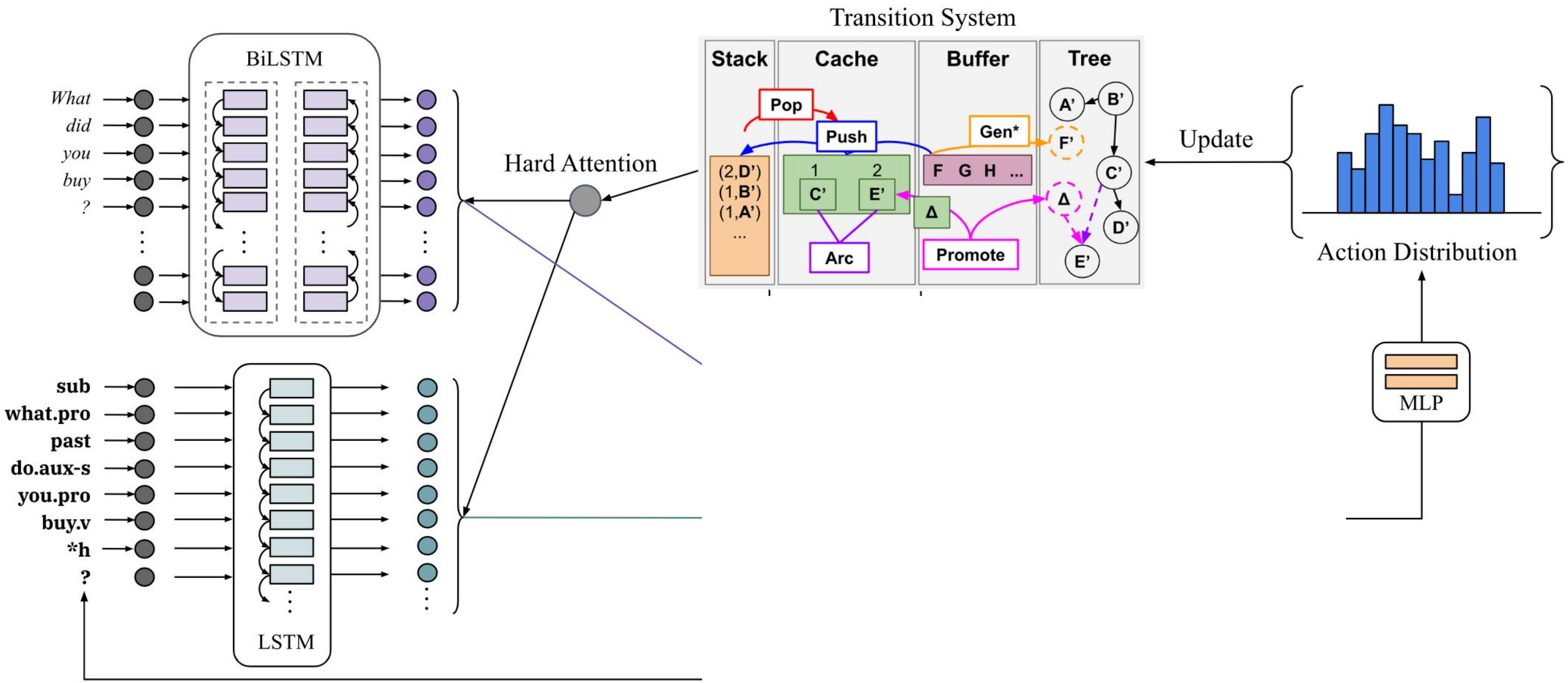



Transition System
Stack
Cache
Buffer
Tree
Pop
Push
Gen*
(2,D')
(1,B')
(1,A')
...
1
2
C'
E'
F G H ...
Δ
Arc
Promote
A'
B'
F'
C'
Δ
D'
E'
Update
Action Distribution
MLP



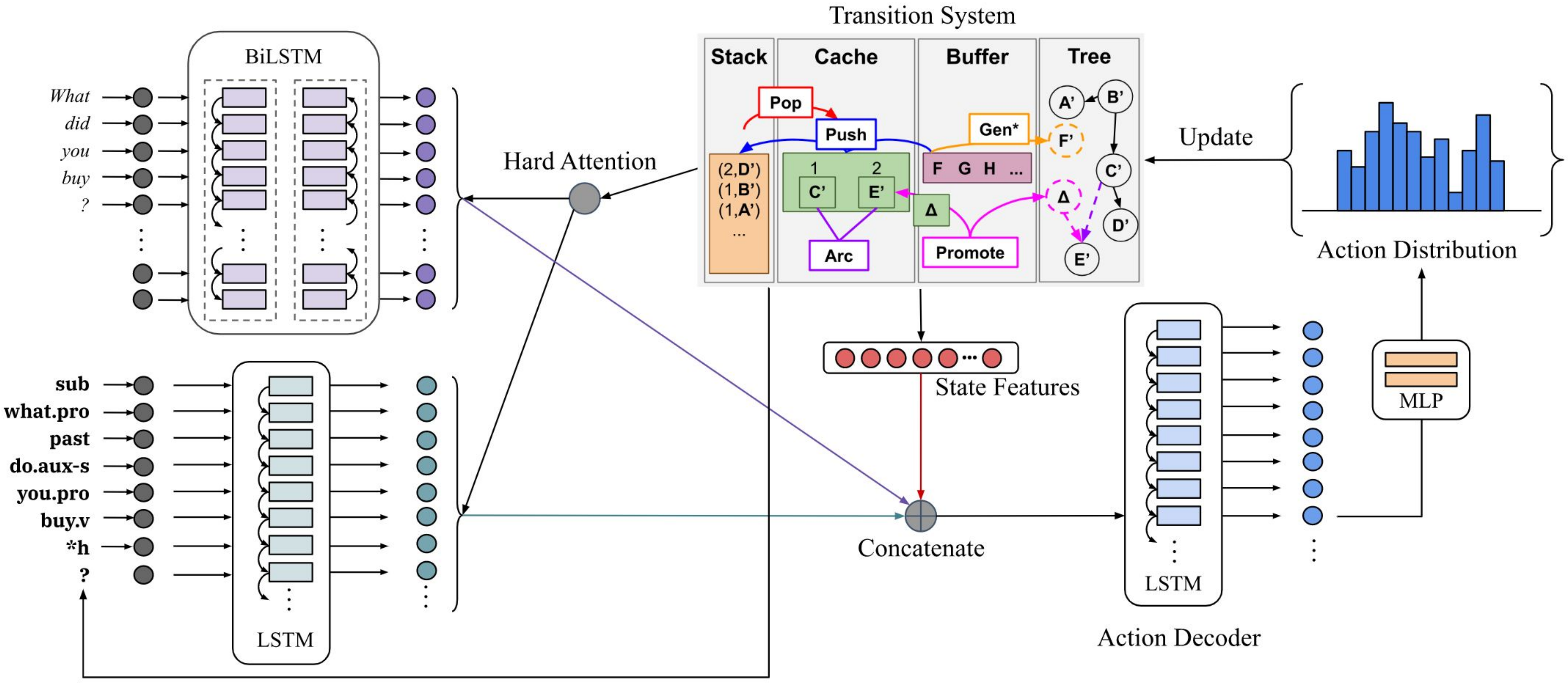
Word Sequence

GloVe + RoBERTa + CharCNN + lemmas + POS + NER



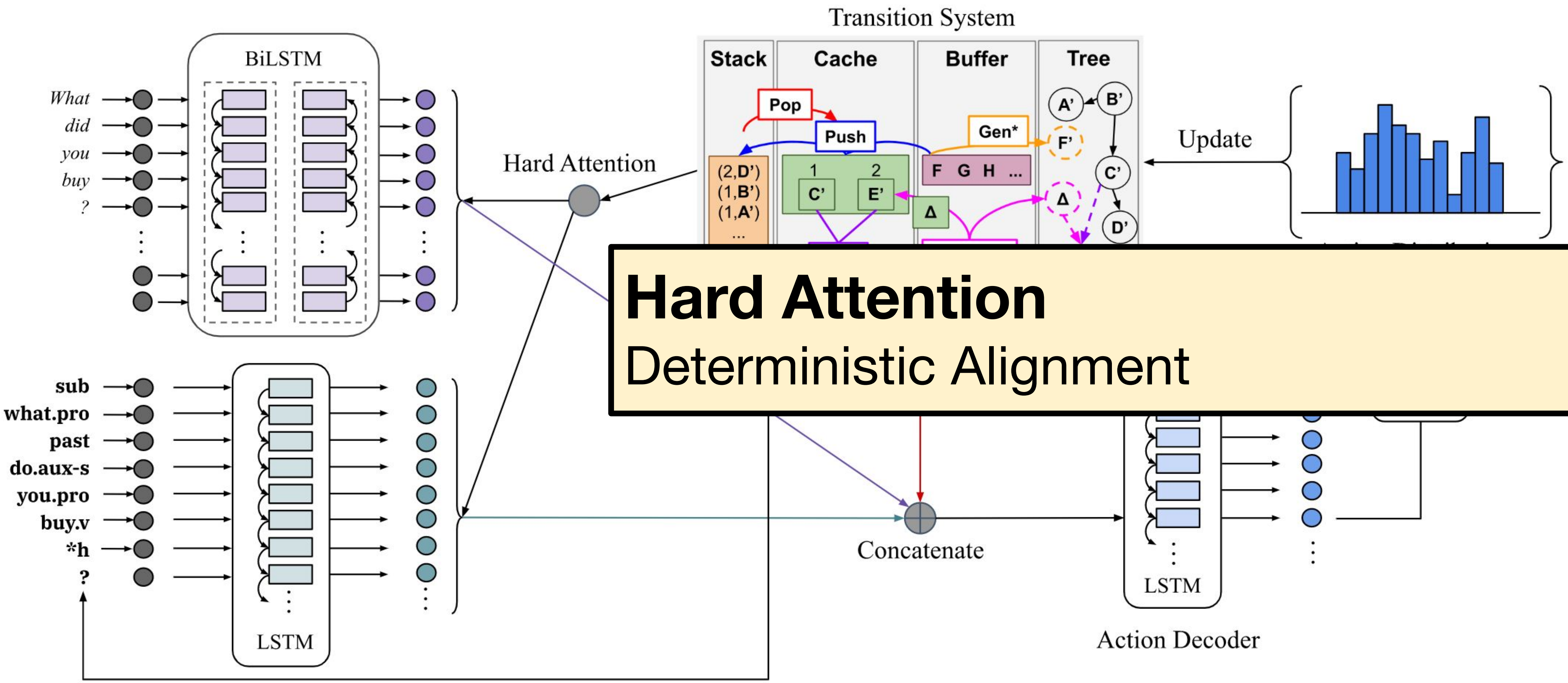
Symbol Sequence

Symbol + CharCNN (of aligned word)


Transition System
Stack
Cache
Buffer
Tree
Pop
Push
Gen*
(2,D')
(1,B')
(1,A')
...
1
2
C'
E'
Δ
F G H ...
Arc
Promote
A'
B'
F'
C'
Δ
D'
E'
BiLSTM
What
did
you
buy
?
Hard Attention
sub
what.pro
past
do.aux-s
you.pro
buy.v
*h
?
LSTM
State Features
Concatenate
LSTM
Action Decoder
MLP
Update
Action Distribution

Hard Attention

Deterministic Alignment

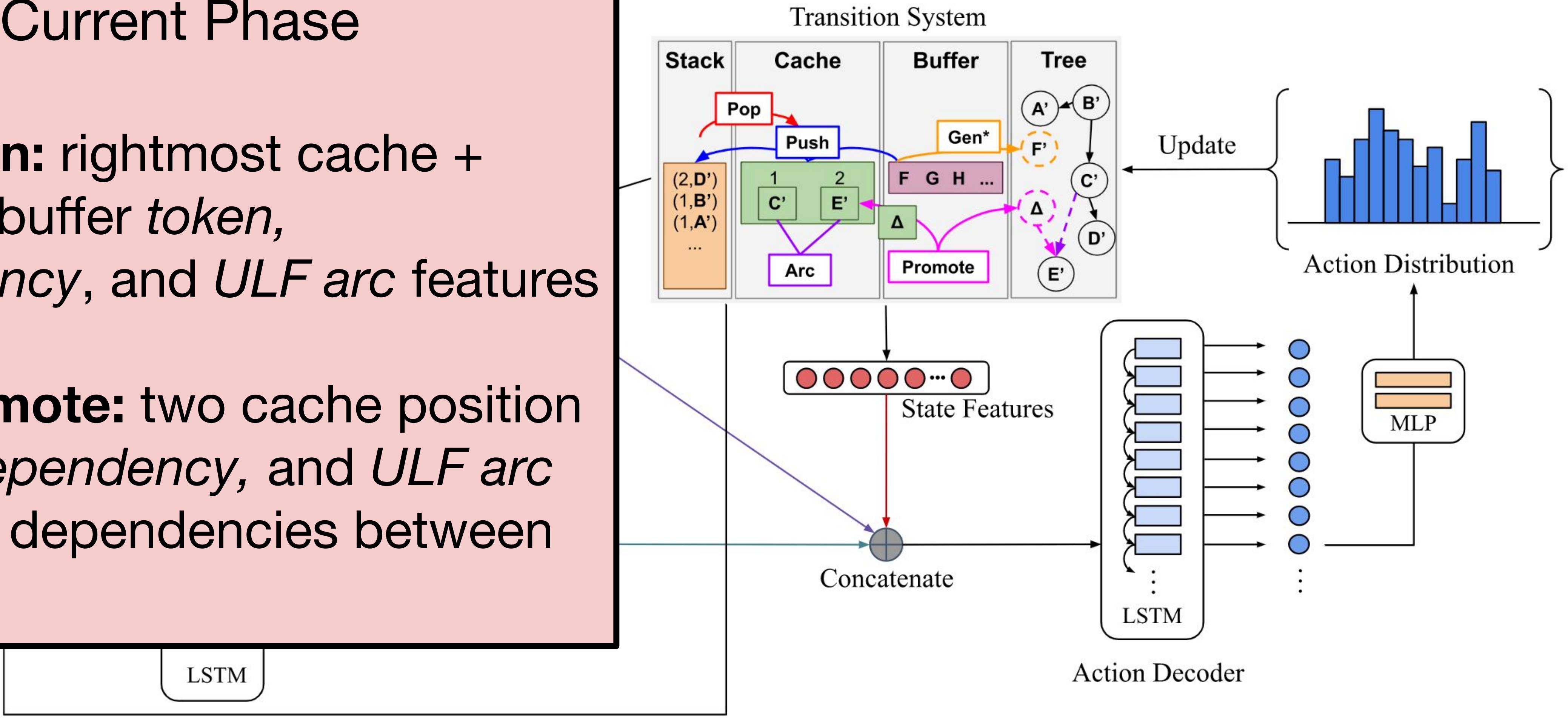


Transition State Features

Always: Current Phase

Pop/*Gen: rightmost cache + leftmost buffer *token, dependency*, and *ULF arc* features

Arc/Promote: two cache position *token, dependency,* and *ULF arc* features; dependencies between them



Experimental Details

Data Split (~8/1/1)

1,738 sentences

- 1,378 train
- 180 dev
- 180 test

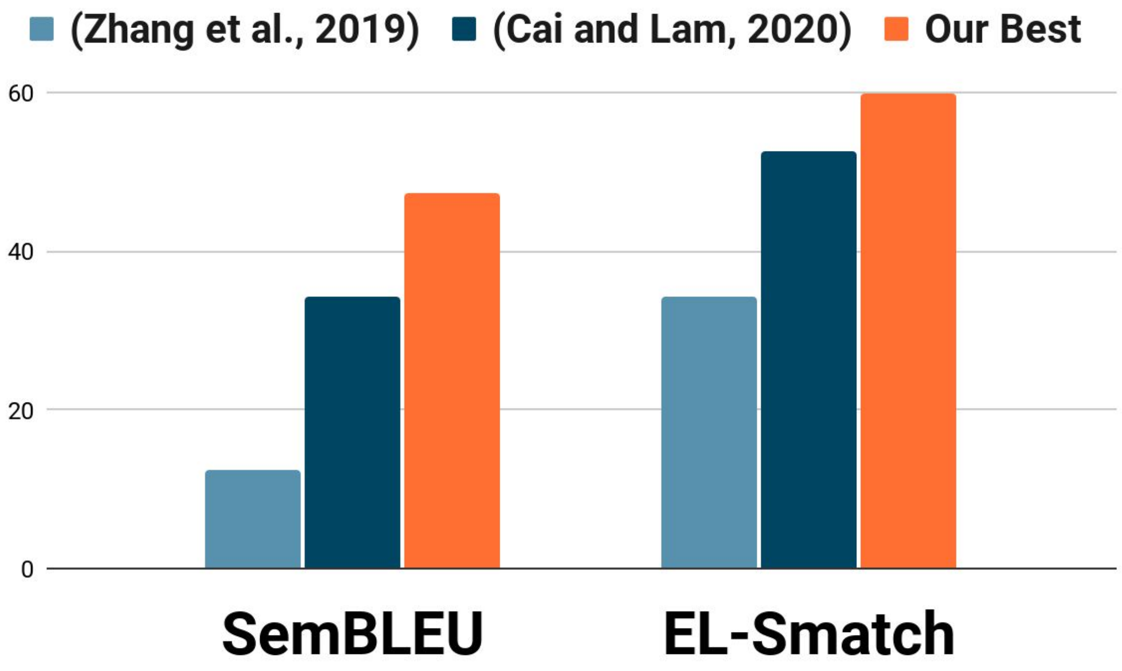
SemBLEU

Extends BLEU to graphs. Based on overlaps of path segments in a graph. [Song & Gildea 2019]

EL-Smatch

Extends smatch to non-atomic operators. Computes node alignment with highest possible overlap of node and edge labels. [Kai & Knight, 2013; Kim & Schubert, 2016]

Comparison to Baselines



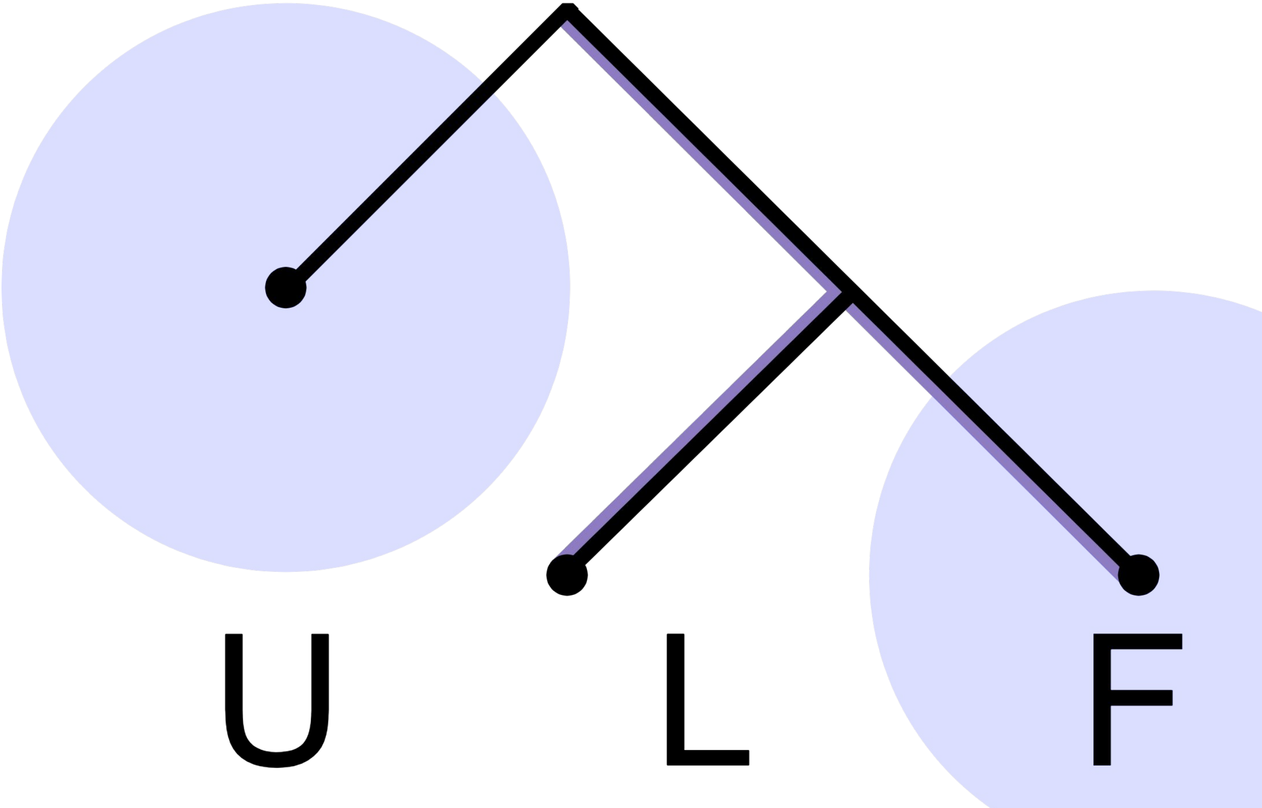
Baselines

Strong AMR parsers w/ minimal AMR-specific assumptions

They struggle on node-label prediction

- dataset is too small

Inference with ULF



Graeme McGuire
Formerly UR

Sophie Sackstein
Booz Allen Hamilton
Formerly UR

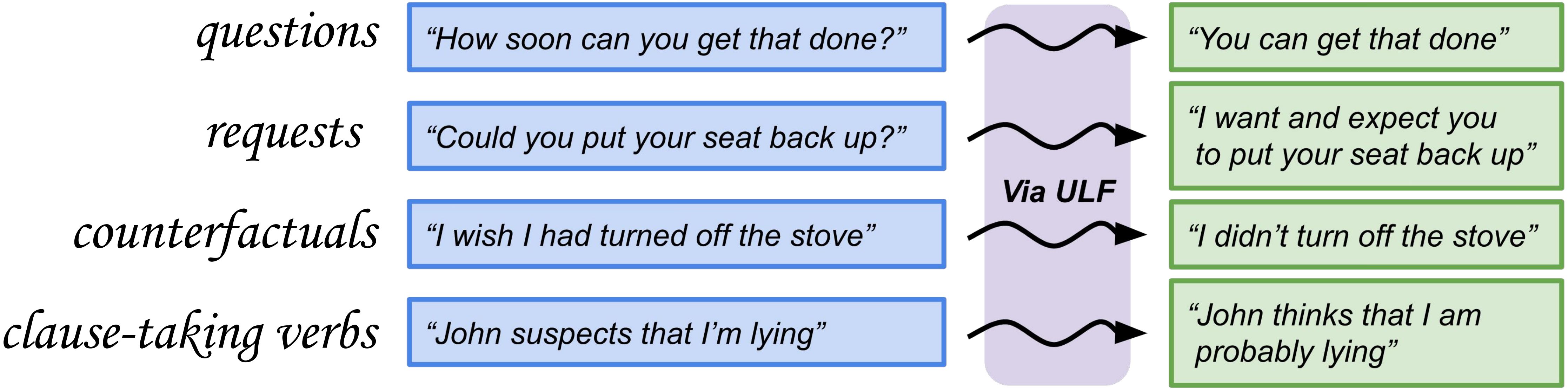
Muskaan Mendiratta
Barclays
Formerly UR

Benjamin Kane
UR

Viet Duong
William & Mary
Formerly UR

Georgiy Platonov
Amazon
Formerly UR

Lenhart Schubert
UR



questions

“How soon can you get that done?”

“You can get that done”

requests

“Could you put your seat back up?”

“I want and expect you to put your seat back up”

counterfactuals

“I wish I had turned off the stove”

“I didn’t turn off the stove”

clause-taking verbs

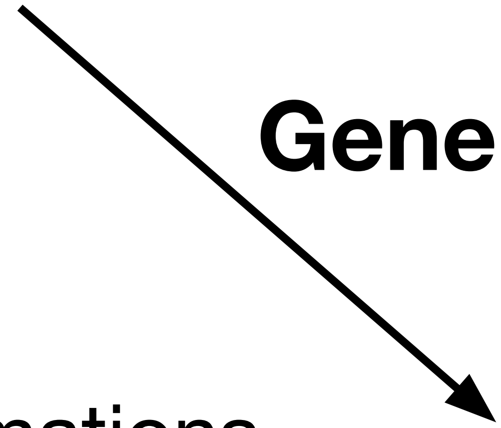
“John suspects that I’m lying”

“John thinks that I am probably lying”

Generative

```
((sub what.pro
   ((past do.aux-s)
    you.pro (buy.v *h))) ?)
```

“what did you buy?”



Generation

Structure

simple symbolic transformations

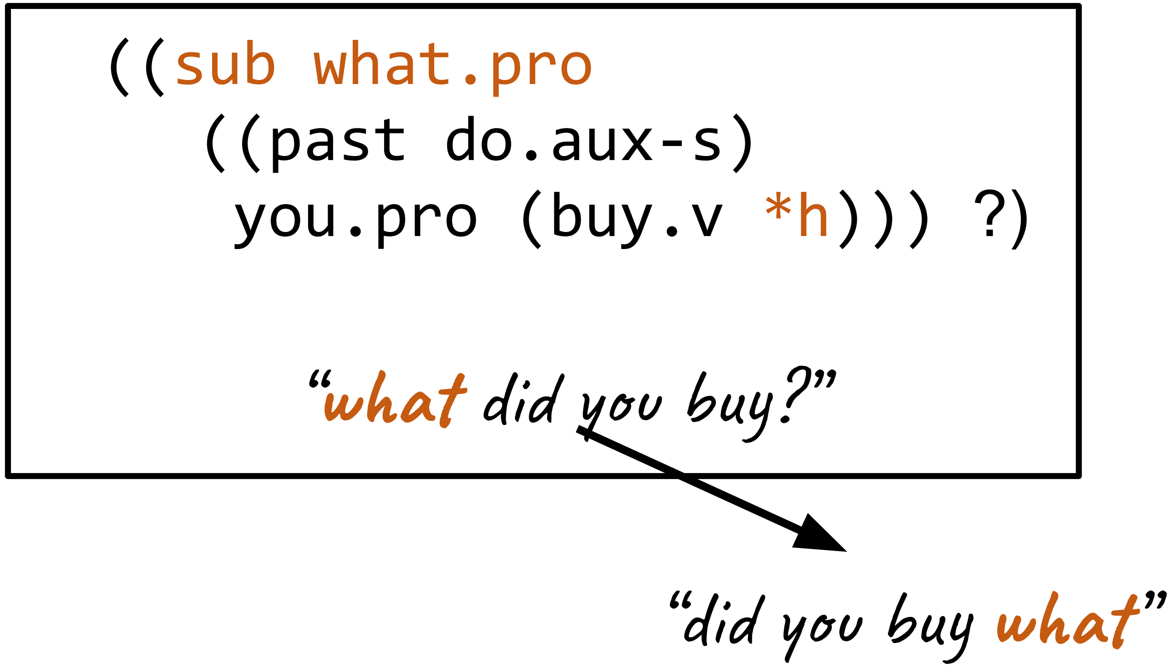
Type System

maintain semantic coherence

“you did buy something”

```
(you.pro ((past do.aux-s)
          (buy.v something.pro)))
```

De-topicalization



```
((sub what.pro
   ((past do.aux-s)
    you.pro (buy.v *h))) ?)
```

"what did you buy?"

"did you buy what"

"you did buy something"

```
(you.pro ((past do.aux-s)
          (buy.v something.pro)))
```

```
((sub what.pro
   ((past do.aux-s)
    you.pro (buy.v *h))) ?)
```

"what did you buy?"

Un-inversion

"did you buy what"

"you did buy what"

"you did buy something"

```
(you.pro ((past do.aux-s)
          (buy.v something.pro)))
```

De-questioning

```
((sub what.pro
   ((past do.aux-s)
    you.pro (buy.v *h))) ?)
```

“what did you buy?”

“did you buy what?”

“you did buy what?”

“you did buy something”

```
(you.pro ((past do.aux-s)
          (buy.v something.pro)))
```

Experimental Details

Precision

Freely generate inferences and judge a sample with human evaluators

- 3 or 4 evaluations per inference

127 inferences

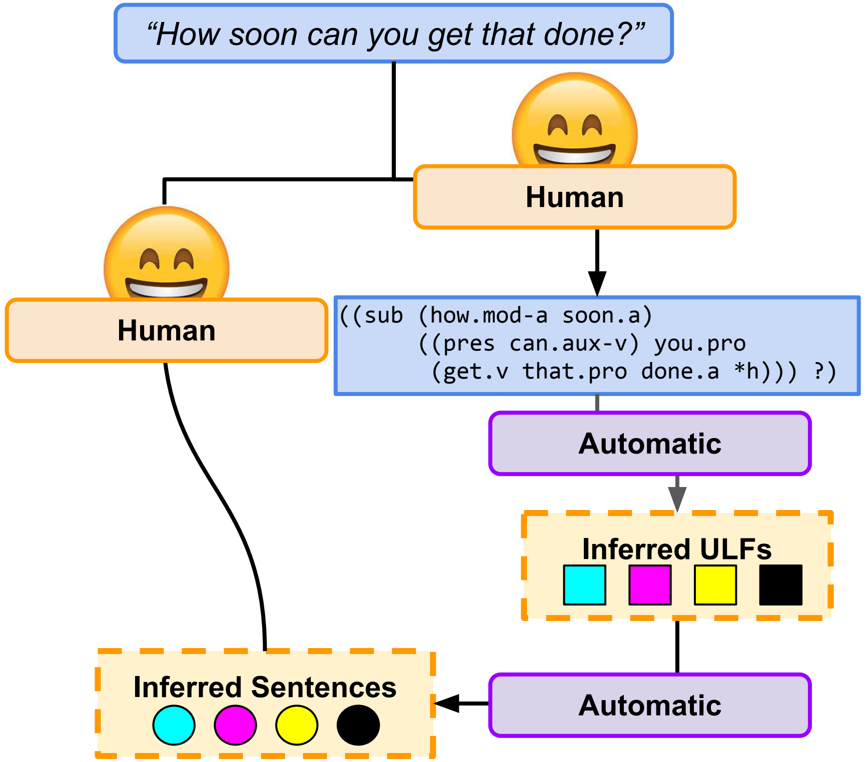
Recall

Get human inferences for a sample of sentences and check coverage that the automatic inferences achieve

- annotators are trained for these phenomena

698 inferences
406 sentences

Precision Evaluation



```
((sub (how.mod-a soon.a)
      ((pres can.aux-v) you.pro
       (get.v that.pro done.a *h))) ?)
```

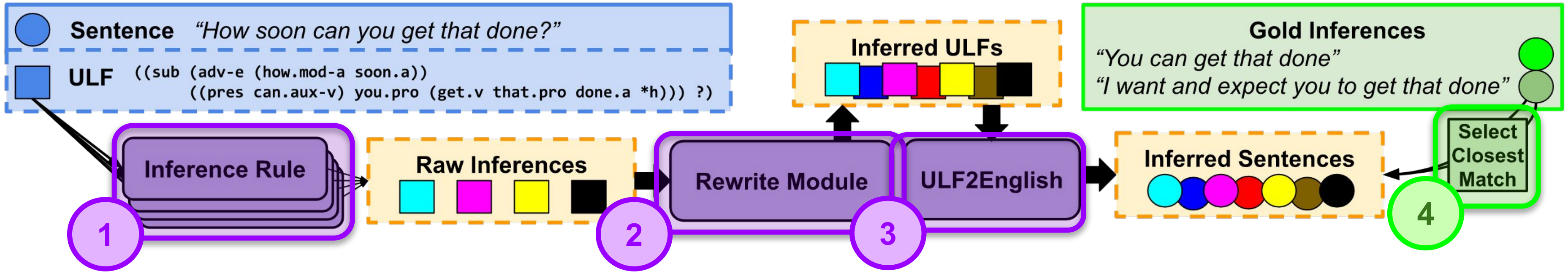
Correct 68.5%

Contextual 15.0%

Incorrect 16.5%

Grammatical 78.0%

Recall Evaluation


Sentence "How soon can you get that done?"
ULF ((sub (adv-e (how.mod-a soon.a)) ((pres can.aux-v) you.pro (get.v that.pro done.a *h))) ?)
Inference Rule
1
Raw Inferences
2
Rewrite Module
3
Inferred ULFs
ULF2English
Inferred Sentences
Gold Inferences
"You can get that done"
"I want and expect you to get that done"
Select Closest Match
4

1. **Basic Inference**
2. **Paraphrasing & Coordination [In ULF]**

   "I want you to get that done" + "I expect you to get that done" → "I want and expect you to get that done"

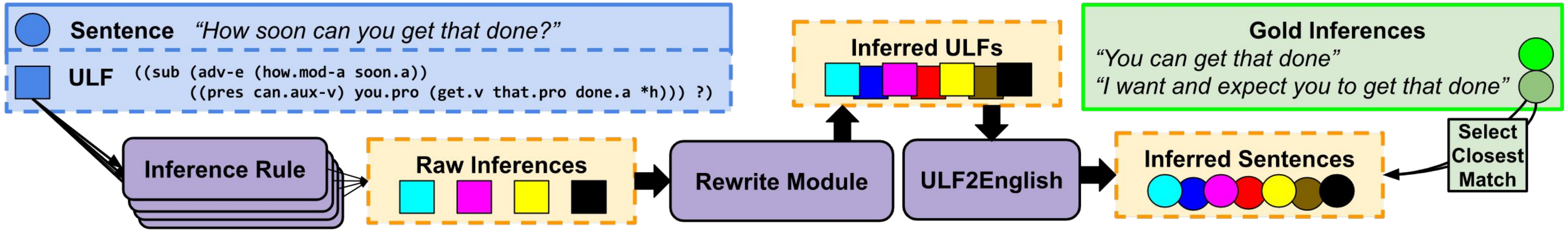
3. **Translate to English**

   `(i.pro (((pres want.v) and.cc (pres expect.v)) you.pro (to (get.v that.pro done.a))))`

   → *"I want and expect you to get that done"*

4. **Select closest match with minimal difference**
   a. Allow 3 character edit distance

Recall Evaluation


Sentence “How soon can you get that done?”
ULF ((sub (adv-e (how.mod-a soon.a)) ((pres can.aux-v) you.pro (get.v that.pro done.a *h))) ?)
Inference Rule
Raw Inferences
Rewrite Module
Inferred ULFs
ULF2English
Inferred Sentences
Gold Inferences
“You can get that done”
“I want and expect you to get that done”
Select Closest Match

Out of 662 inferences, 112 found (~17%)

*Simple baseline ~0%

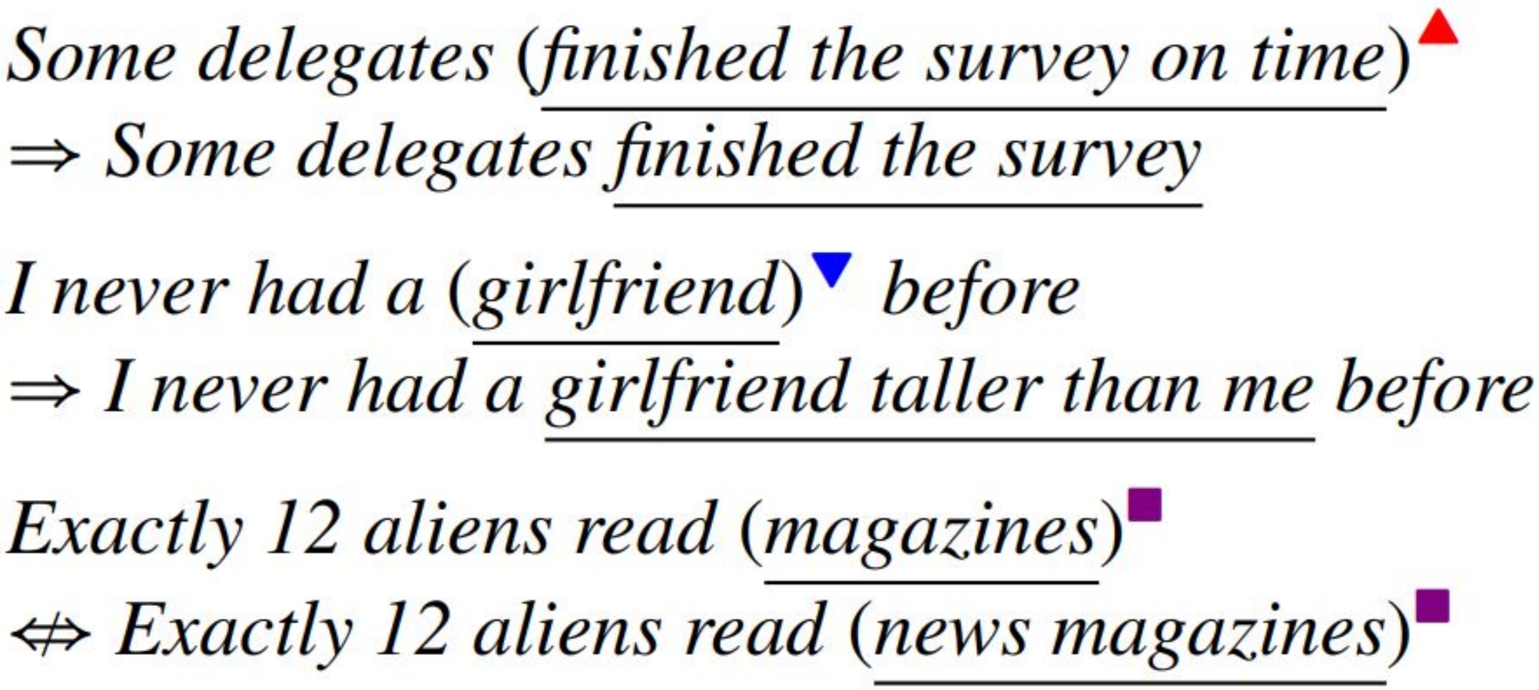
Natural Logic

Generate natural language inferences based on syntactic structure and local semantic properties

Van Benthem et al., 1986; Sánchez Valencia, 1991

Monotonicity Inference

Specialization and generalization inferences based on contexts imposed by polarity operators



Some delegates ($\underline{\textit{finished the survey on time}}$)▲

$\Rightarrow$ *Some delegates* $\underline{\textit{finished the survey}}$

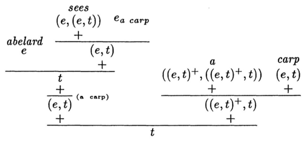
I never had a ($\underline{\textit{girlfriend}}$)▼ *before*

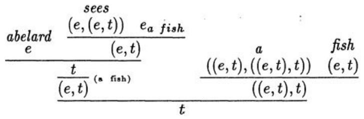
$\Rightarrow$ *I never had a* $\underline{\textit{girlfriend taller than me}}$ *before*

Exactly 12 aliens read ($\underline{\textit{magazines}}$)■

$\not\Leftrightarrow$ *Exactly 12 aliens read* ($\underline{\textit{news magazines}}$)■

Sánchez Valencia





Lambek Derivations
Tableau-style proofs

"abelard sees a carp"
"every carp is a fish"

"abelard sees a fish"

Replace Lambek derivations and sentences with ULFs

(|Abelard| (see.v (a.d fish.n)))

(|Abelard| (see.v (a.d carp.n)))

ULF

Mandar Juvekar
UR

Junis Ekmekciu
UR

Viet Duong
UR

Lenhart Schubert
UR

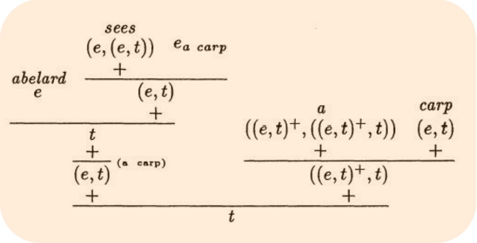
Sánchez Valencia’s System

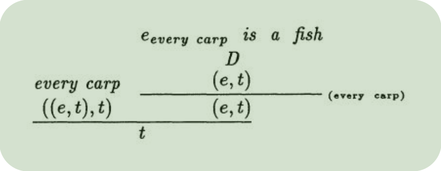
Inference 1 *abelard sees a carp*, *every carp is a fish* / *abelard sees a fish*

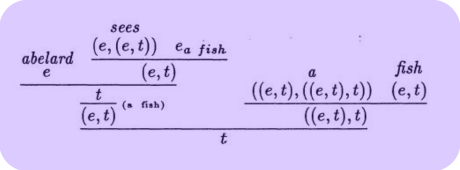
Monotonicity

$$(every\ x)^{\#}\ is\ a\ y, F(x^{+}), X \bullet Y$$

$$(every\ x)^{\#}\ is\ a\ y, F(y), X \bullet Y$$







abe see a carp, every carp is a fish $\bullet$ *abe see a fish*

abe sees (*a carp*)$^{\#}$, (*every carp*)$^{\#}$ is a fish $\bullet$ *abe sees* (*a fish*)$^{\#}$ **marking**

abe sees (*a carp*$^{+}$)$^{\#}$, (*every carp*)$^{\#}$ is a fish $\bullet$ *abe sees* (*a fish*)$^{\#}$ **marking**

abe sees (*a fish*)$^{\#}$, (*every carp*)$^{\#}$ is a fish $\bullet$ *abe sees* (*a fish*)$^{\#}$ **monotonicity**

Natural Logic with ULFs

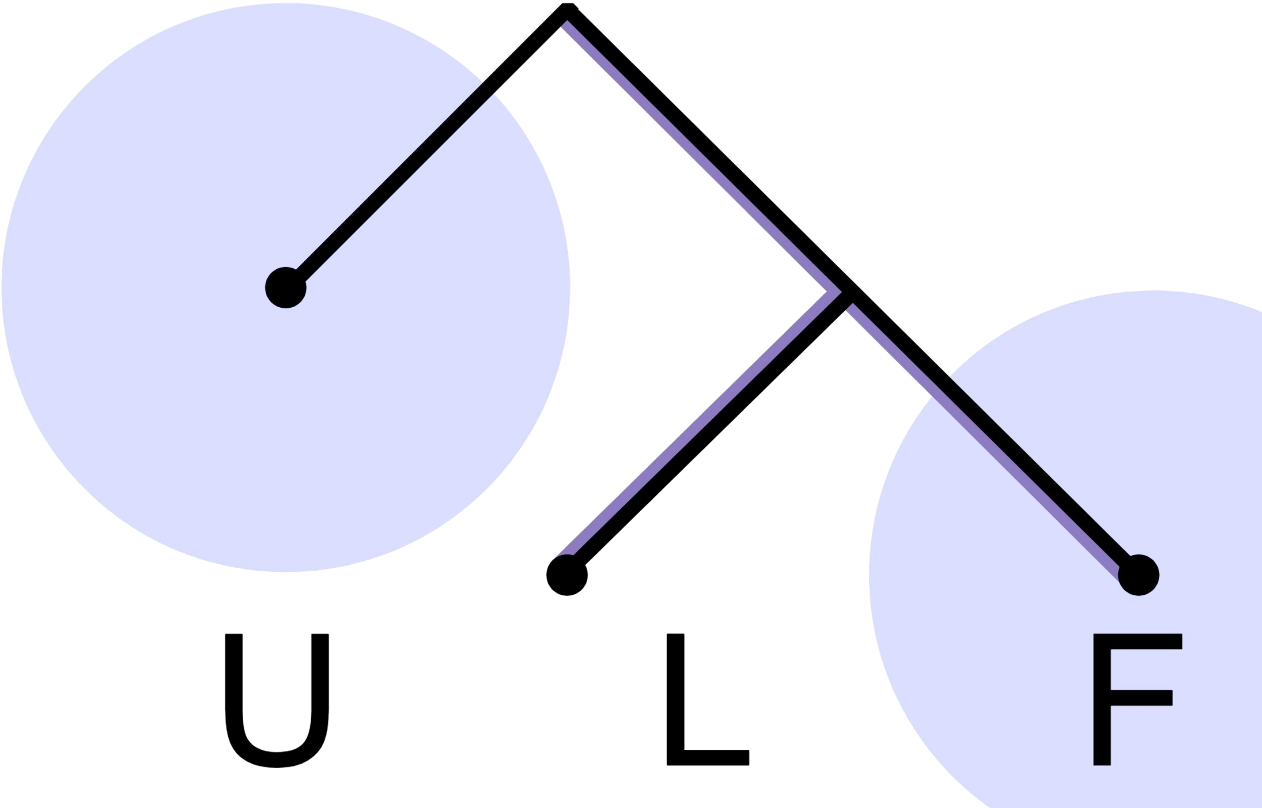
"Abelard sees a carp"	1.	(|Abelard| (see.v (a.d carp.n)))	Assumption
"Every carp is a fish"	2.	((every.d carp.n) (be.v (= (a.d fish.n))))	Assumption
↓	3.	$(\text{a.d } x\text{: } (x \text{ carp.n})^{+}\ (|\text{Abelard}|\ (\text{see.v } x)^{+})^{+})$	SLF of 1. w/ polarity
	4.	$(|\text{Abelard}|\ (\text{see.v } (\text{a.d carp.n})^{+}))$	Pol marking 1.,3.
"Abelard sees a fish"	5.	(|Abelard| (see.v (a.d fish.n)))	UMI 2.,4.

Monotonicity (UMI)

$$\frac{\phi[(\delta\ P1)^{+}],\ ((\text{every.d } P1)\ (\text{be.v } (= (\text{a.d } P2))))}{\phi[(\delta\ P2)]}$$

where δ is a determiner.

Data

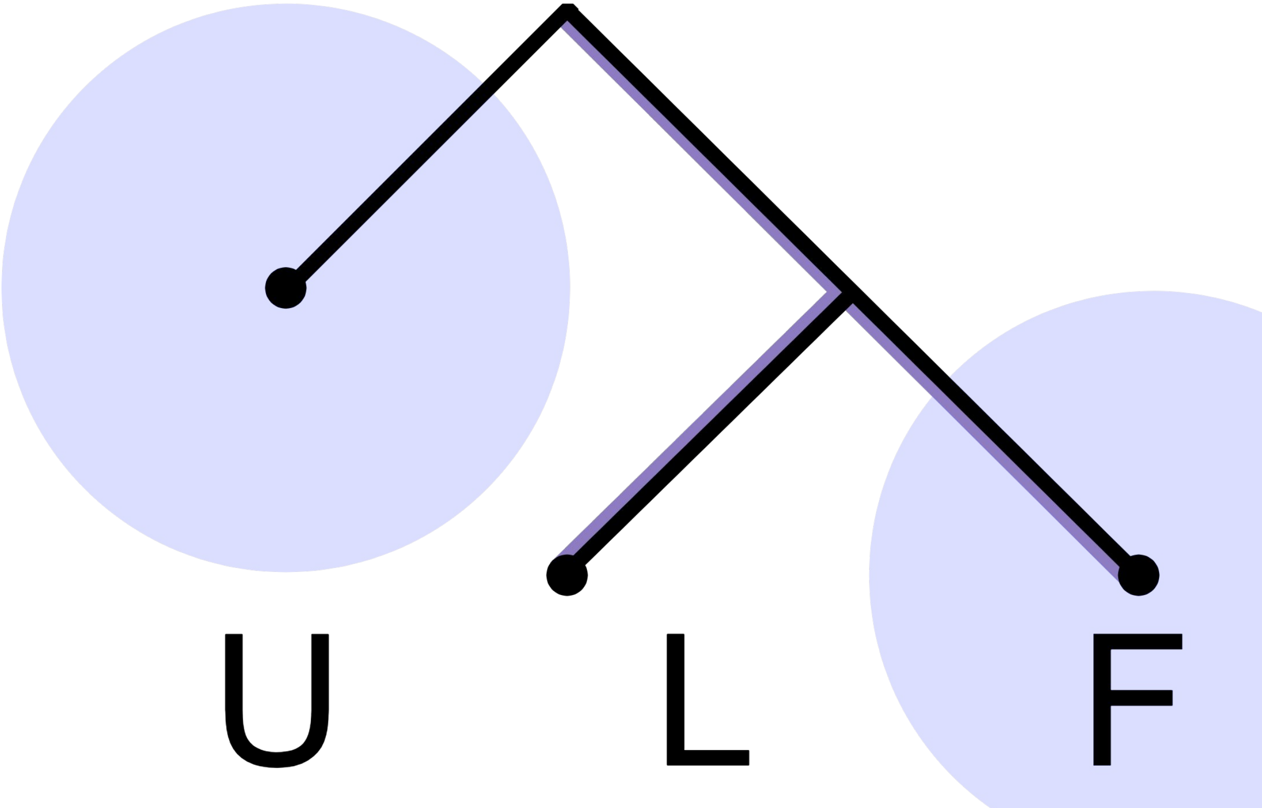


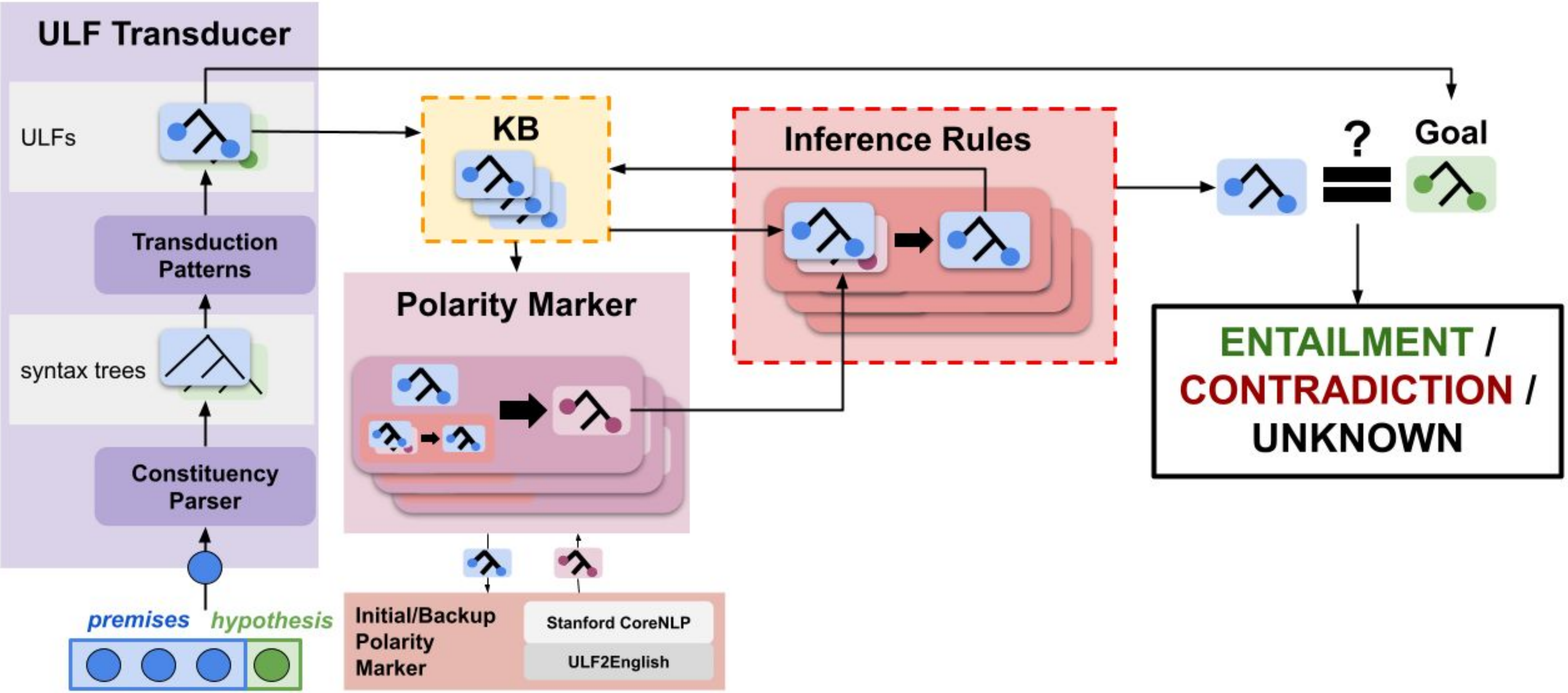
Premises	*Some delegates finished the survey on time*
Hypothesis	*Some delegates finished the survey*
Label	**ENTAILMENT**

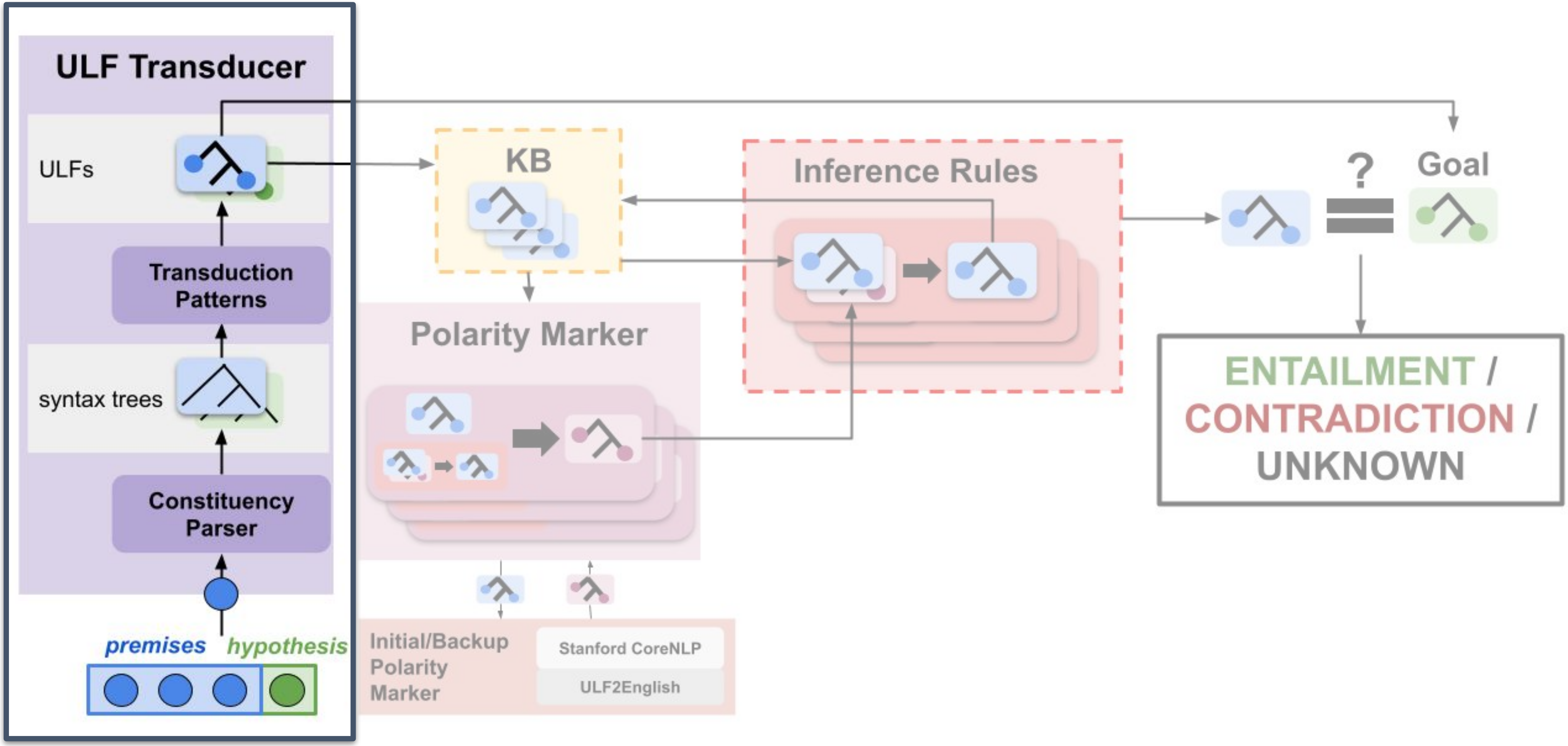
FraCaS Generalized Quantifiers (GQs)

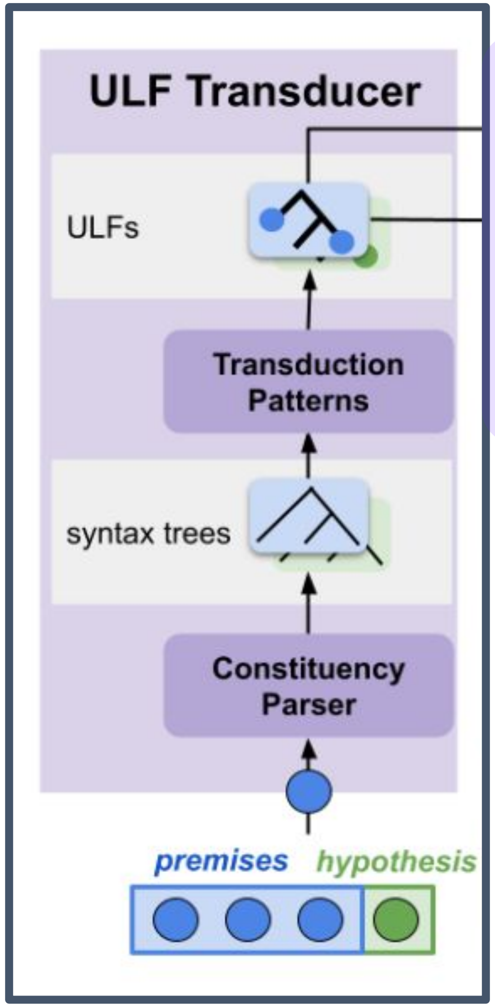
1. Curated by linguists
2. Largest section of FraCaS (80/346, 23%)
3. Quantifiers impose polarities on restrictor and scope

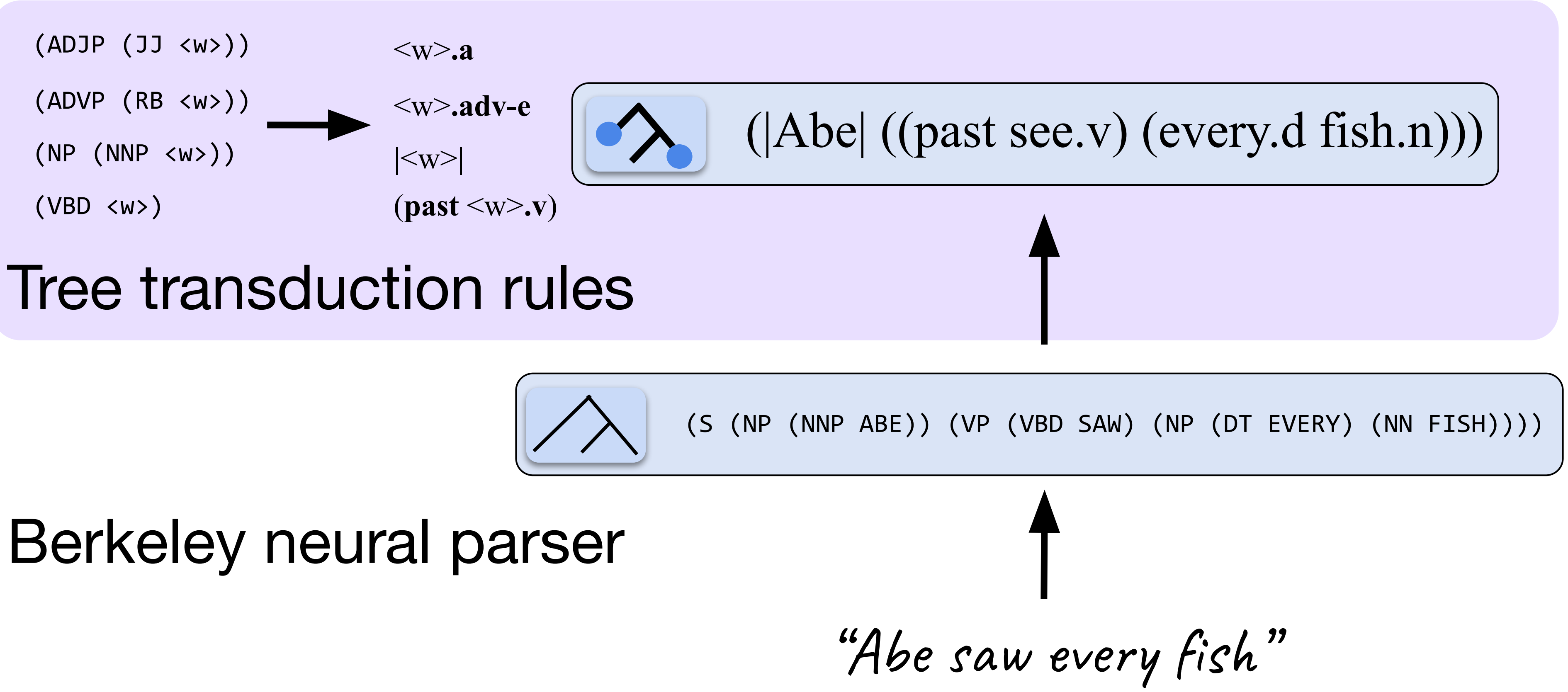
Inference System




ULF Transducer
ULFs
Transduction Patterns
syntax trees
Constituency Parser
premises
hypothesis
KB
Polarity Marker
Initial/Backup Polarity Marker
Stanford CoreNLP
ULF2English
Inference Rules
?
Goal
ENTAILMENT /
CONTRADICTION /
UNKNOWN


ULF Transducer
ULFs
Transduction Patterns
syntax trees
Constituency Parser
premises
hypothesis
KB
Inference Rules
?
Goal
Polarity Marker
ENTAILMENT /
CONTRADICTION /
UNKNOWN
Initial/Backup Polarity Marker
Stanford CoreNLP
ULF2English





(ADJP (JJ <w>))	<w>.a
(ADVP (RB <w>))	<w>.adv-e
(NP (NNP <w>))	|<w>|
(VBD <w>)	(past <w>.v)

(|Abe| ((past see.v) (every.d fish.n)))

Tree transduction rules

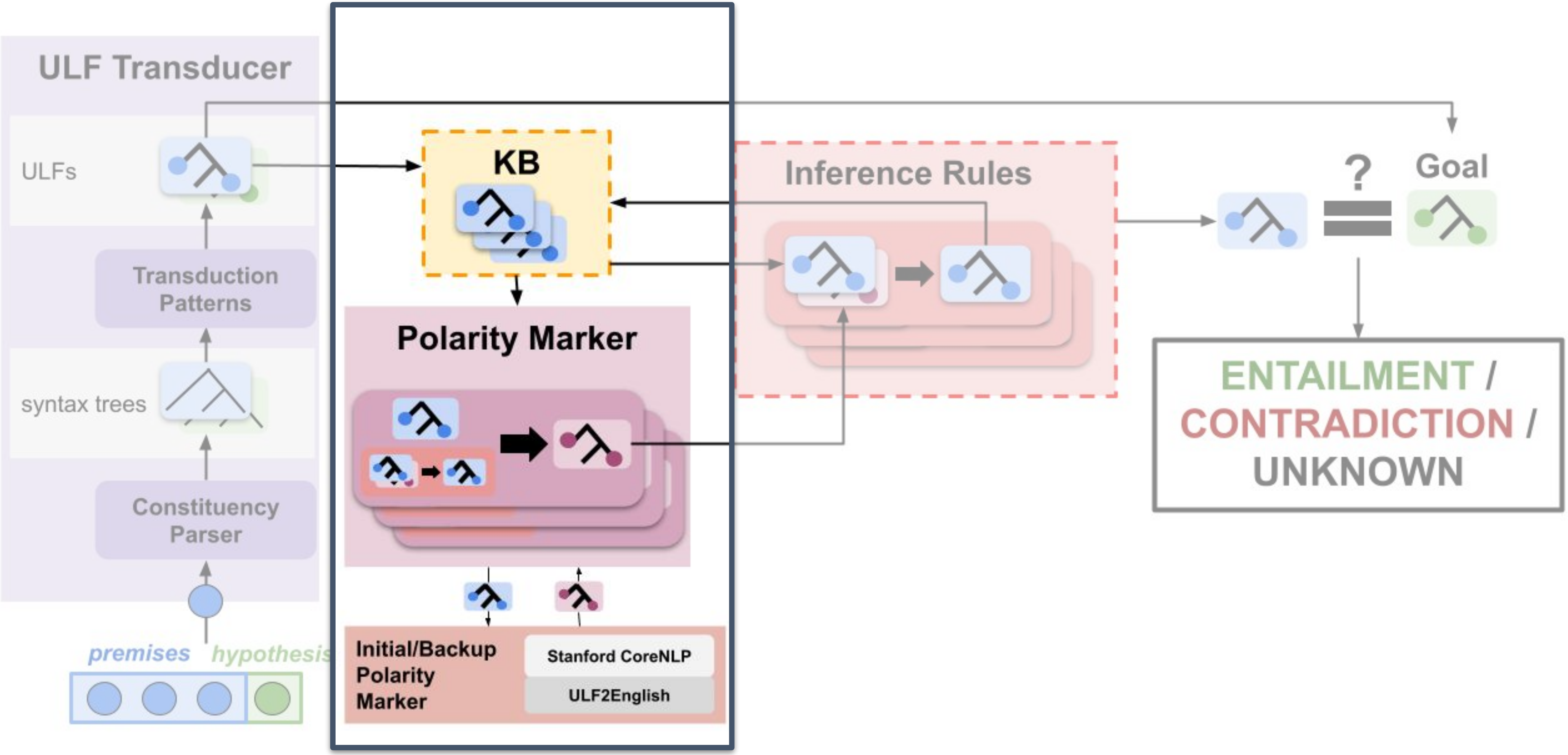
(S (NP (NNP ABE)) (VP (VBD SAW) (NP (DT EVERY) (NN FISH))))

Berkeley neural parser

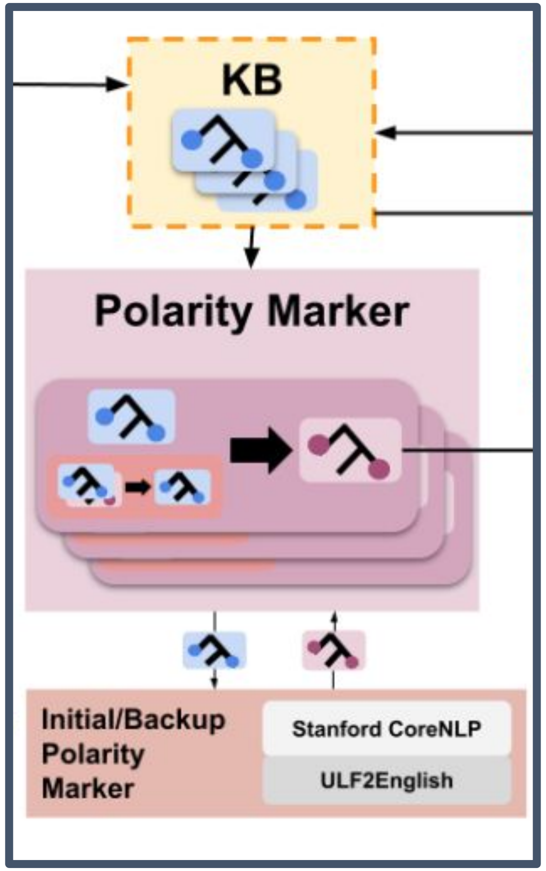
“Abe saw every fish”

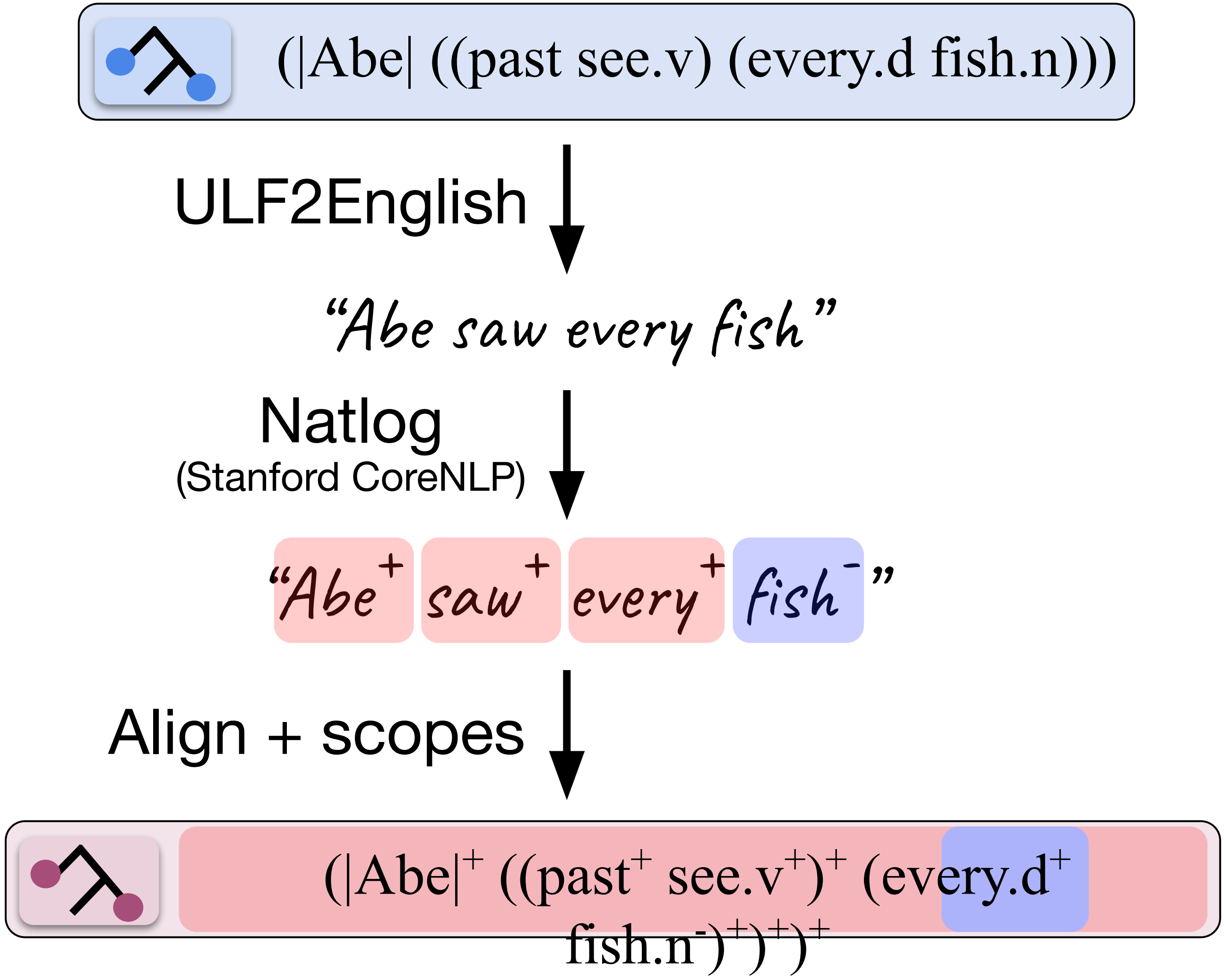
Why not a trained ULF parser?

1. Short, grammatical sentences
2. Errors are more regular and predictable


ULF Transducer
ULFs
Transduction Patterns
syntax trees
Constituency Parser
premises
hypothesis
KB
Polarity Marker
Initial/Backup Polarity Marker
Stanford CoreNLP
ULF2English
Inference Rules
?
Goal
ENTAILMENT / CONTRADICTION / UNKNOWN

Initial Polarity Marking





(|Abe| ((past see.v) (every.d fish.n)))

ULF2English

"Abe saw every fish"

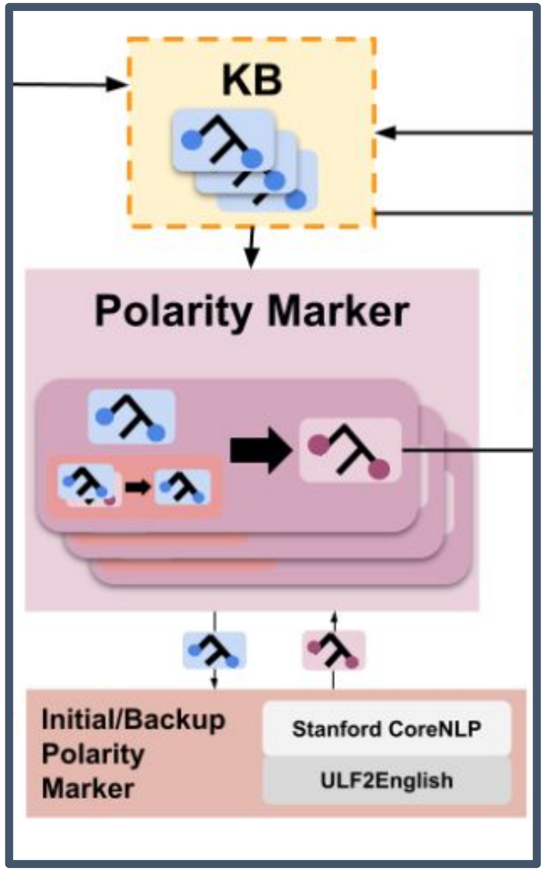
Natlog
(Stanford CoreNLP)

"Abe^{+} saw^{+} $every^{+}$ $fish^{-}$"

Align + scopes

$(|Abe|^{+}\ ((past^{+}\ see.v^{+})^{+}\ (every.d^{+}\ fish.n^{-})^{+})^{+})^{+}$

Polarity Propagation



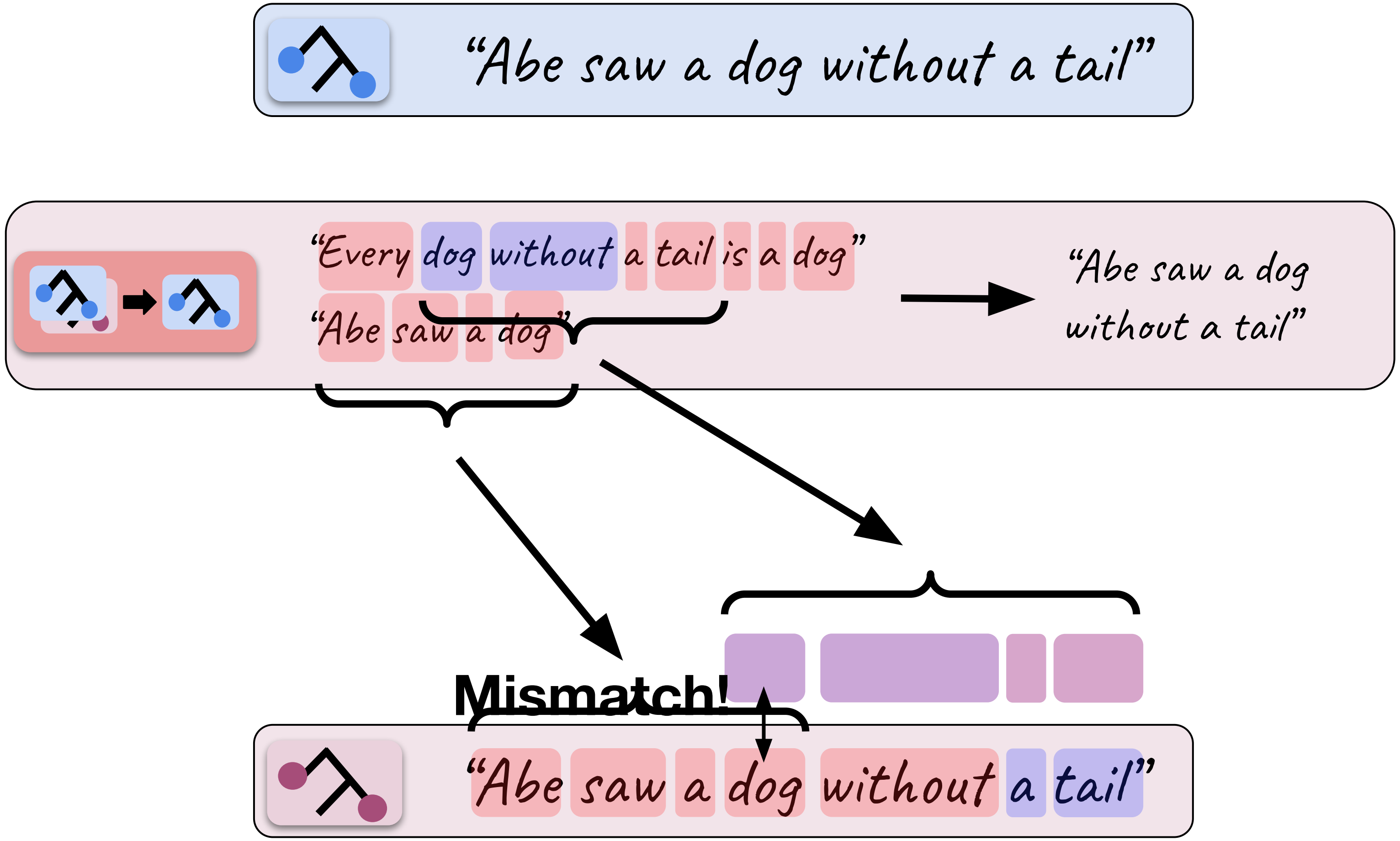
KB

Polarity Marker

Initial/Backup Polarity Marker

Stanford CoreNLP

ULF2English



"Abe saw a dog without a tail"

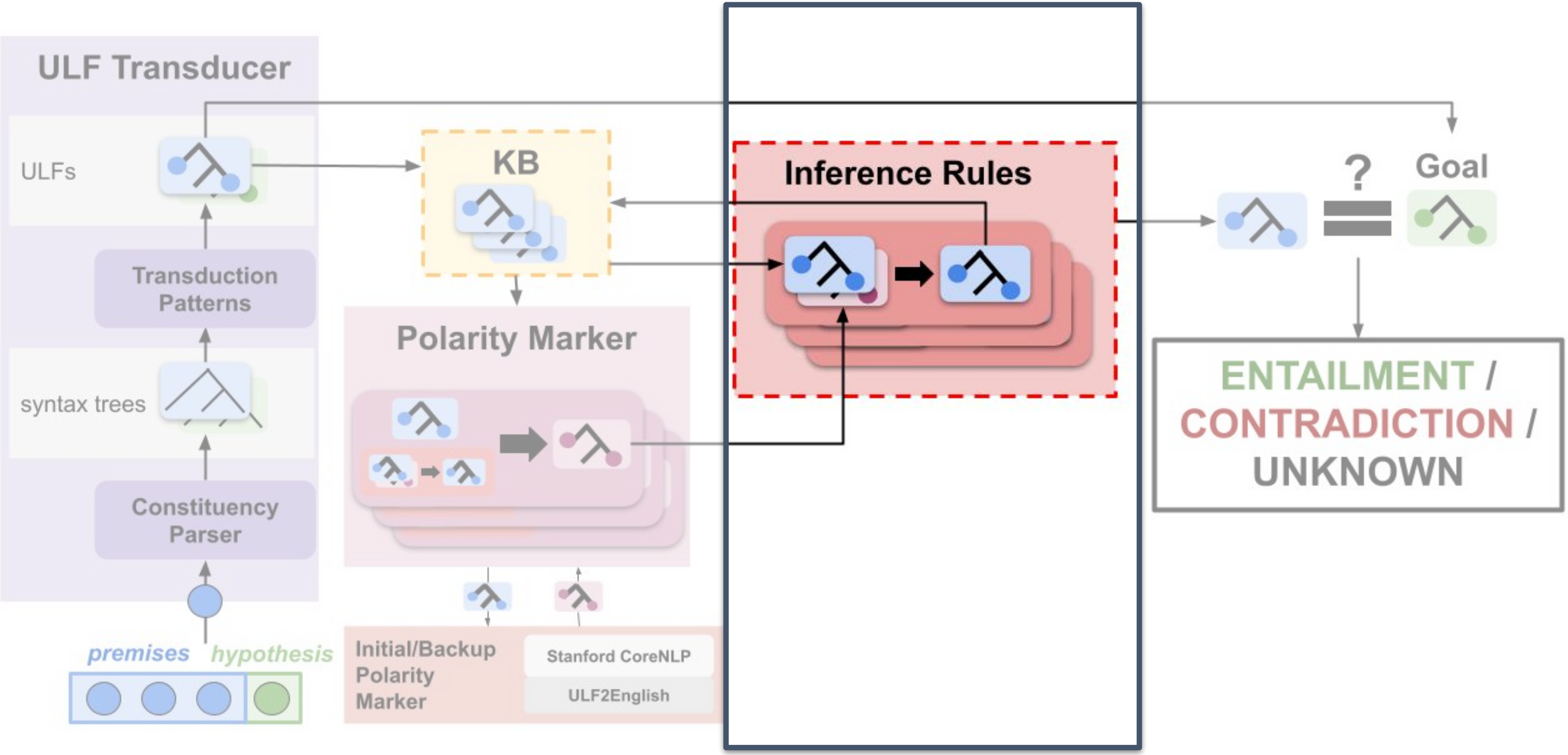
"Every dog without a tail is a dog"

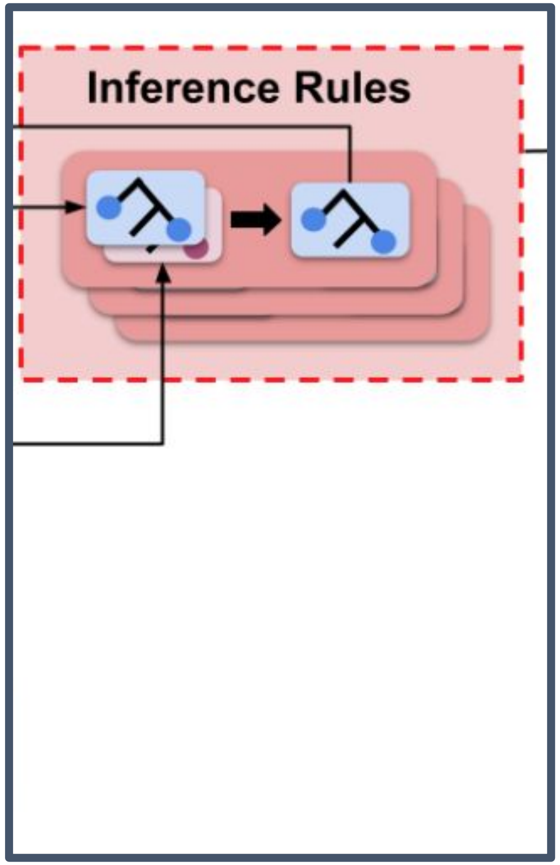
"Abe saw a dog"

"Abe saw a dog without a tail"

Mismatch!

"Abe saw a dog without a tail"


ULF Transducer
ULFs
Transduction Patterns
syntax trees
Constituency Parser
premises
hypothesis
KB
Polarity Marker
Initial/Backup Polarity Marker
Stanford CoreNLP
ULF2English
Inference Rules
?
Goal
ENTAILMENT /
CONTRADICTION /
UNKNOWN



1. **Monotonicity Substitution**

   Every A is a B + S[A+] ⇒ S[B]

2. **Conversion**

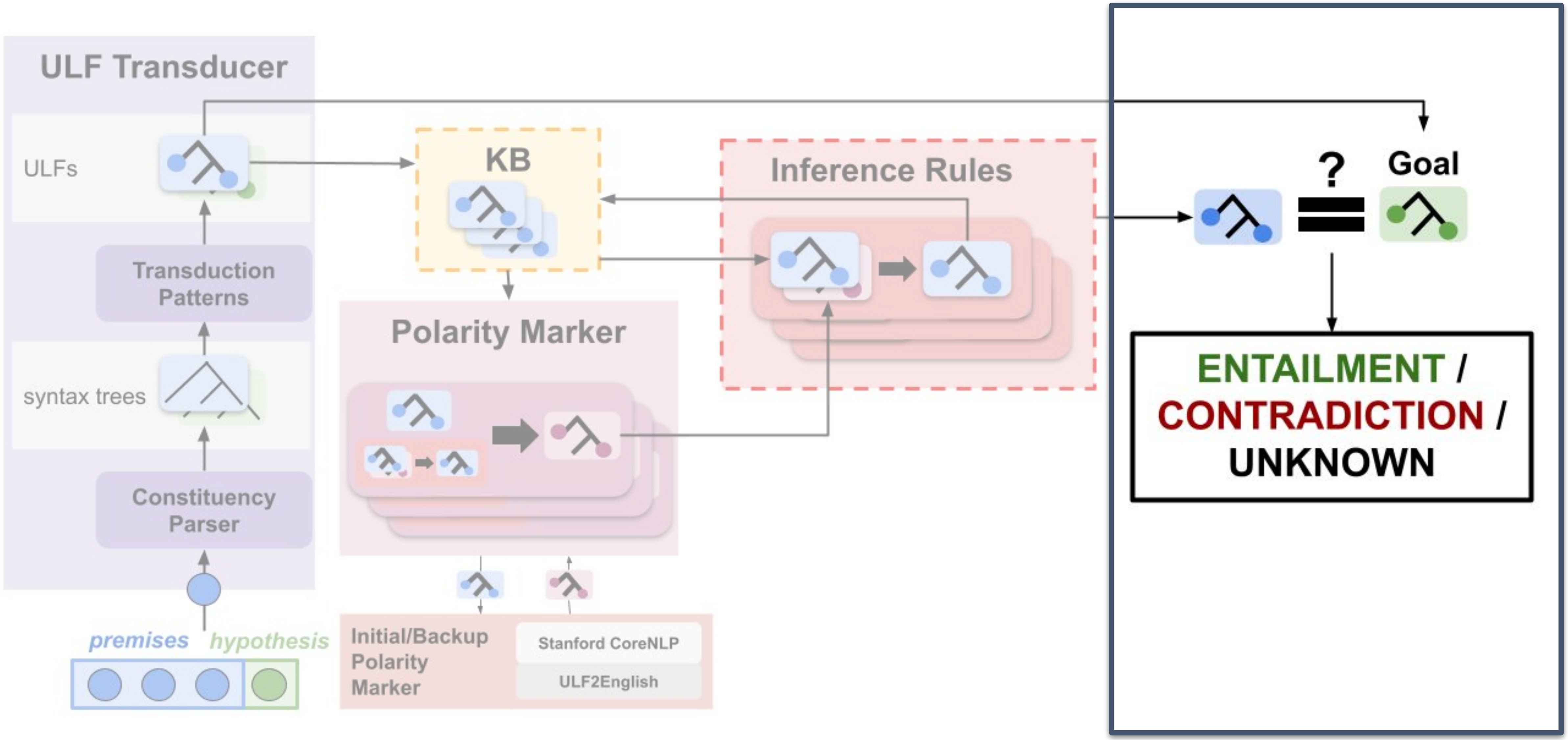
   Some A is a B ⇔ *Some B is an A*

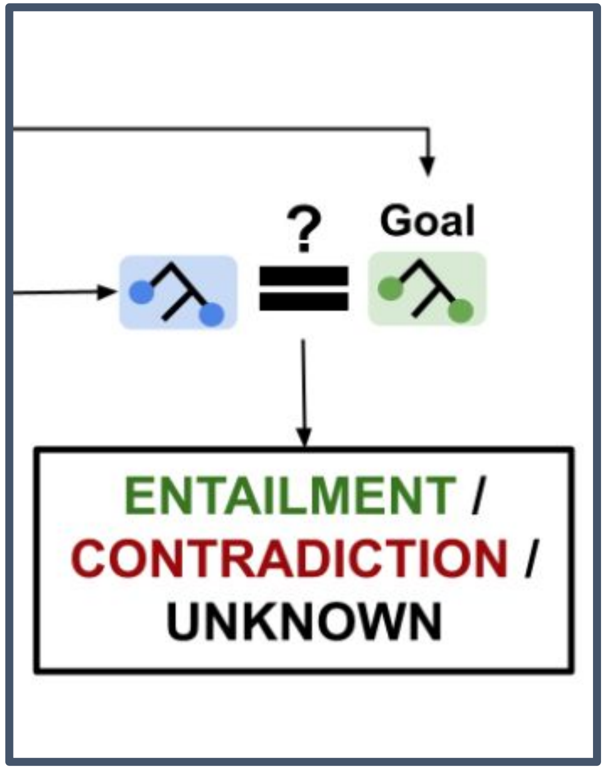
3. **Conservativity**

   DET As are Bs ⇔ *DET As are As that/who are Bs*

4. **Equivalences**

   e.g., Every dog is happy ⇔ *All dogs are happy*


ULF Transducer
ULFs
Transduction Patterns
syntax trees
Constituency Parser
premises
hypothesis
KB
Polarity Marker
Initial/Backup Polarity Marker
Stanford CoreNLP
ULF2English
Inference Rules
?
Goal
ENTAILMENT /
CONTRADICTION /
UNKNOWN



Search: Interleaved heuristic and breadth-first search

maintain completeness with simple/quick heuristic

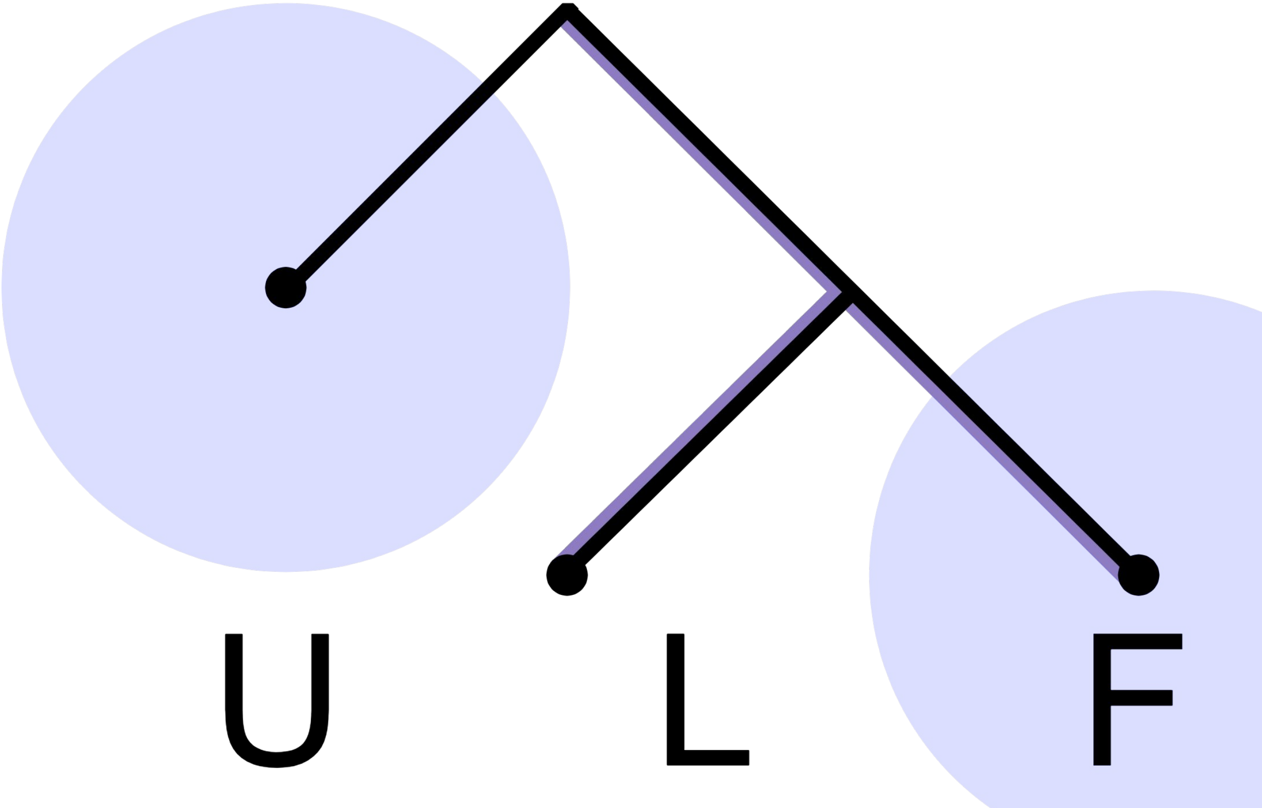
Heuristic: F1 score between atoms of new formula and goal

ENTAILMENT : exact match

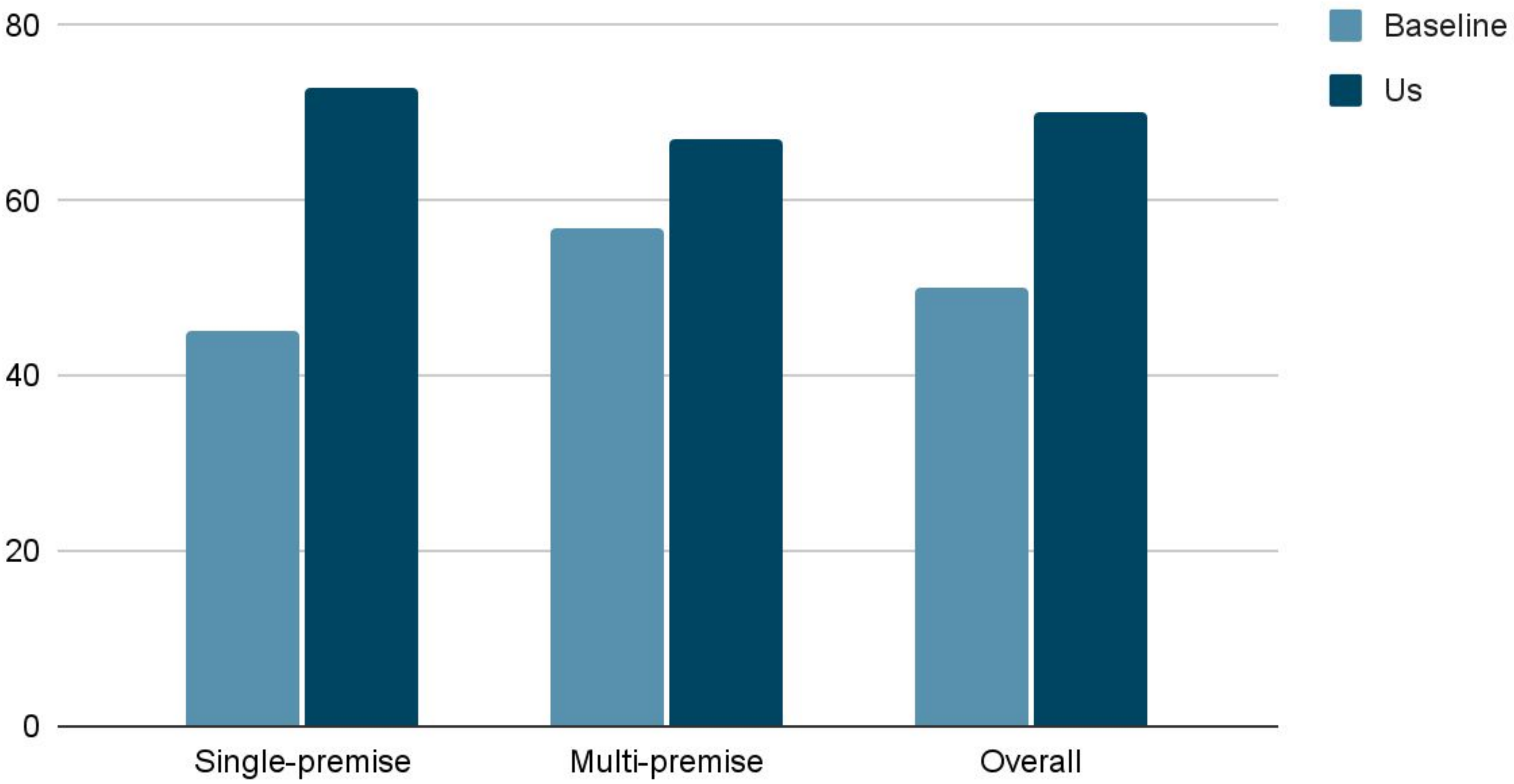
CONTRADICTION : top-level negation + exact match

UNKNOWN : reached max # of steps or exhausted all inferences

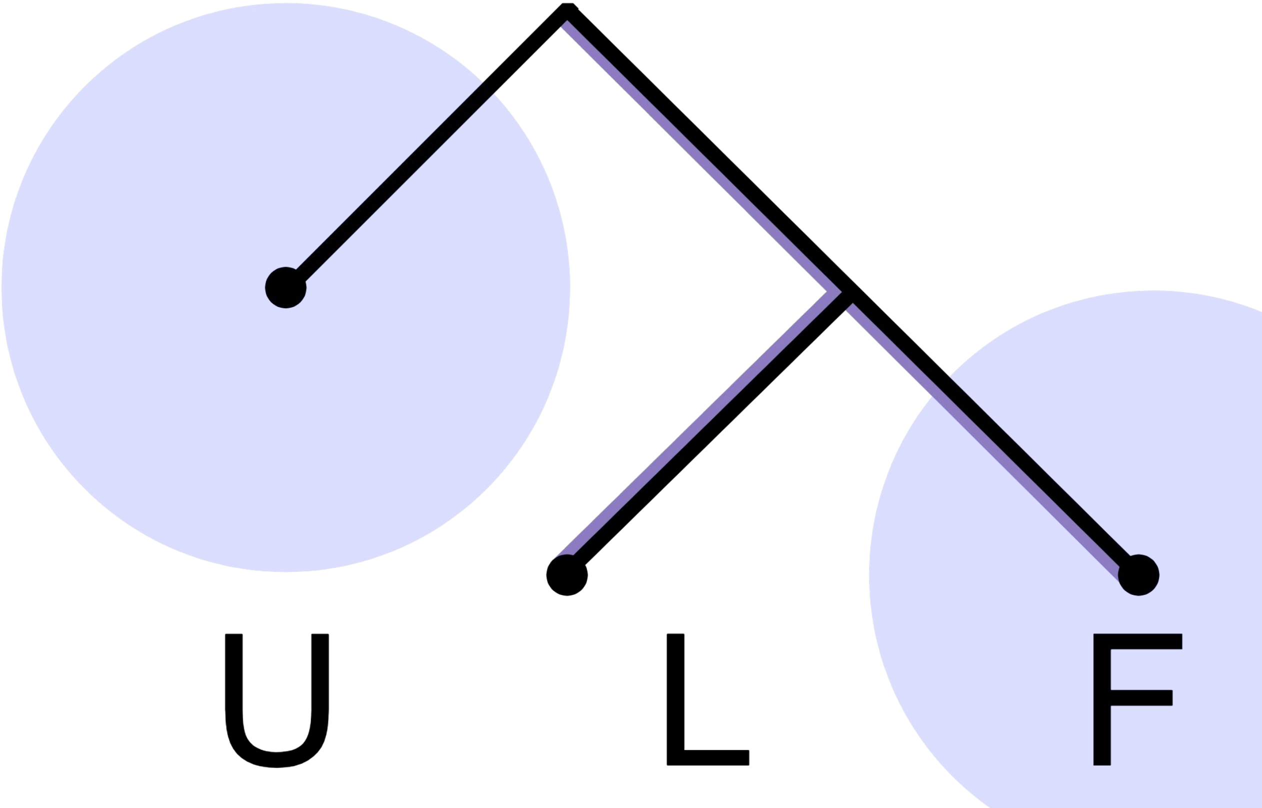
Results



FraCaS GQ Performance

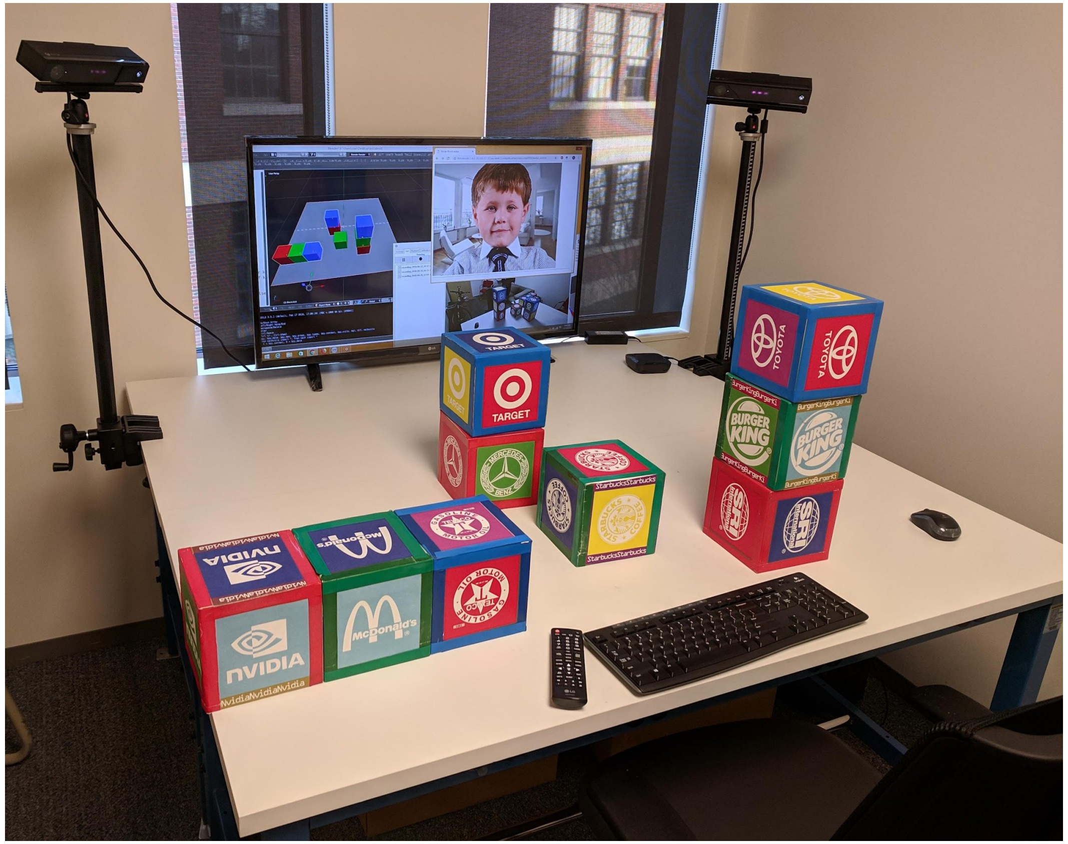


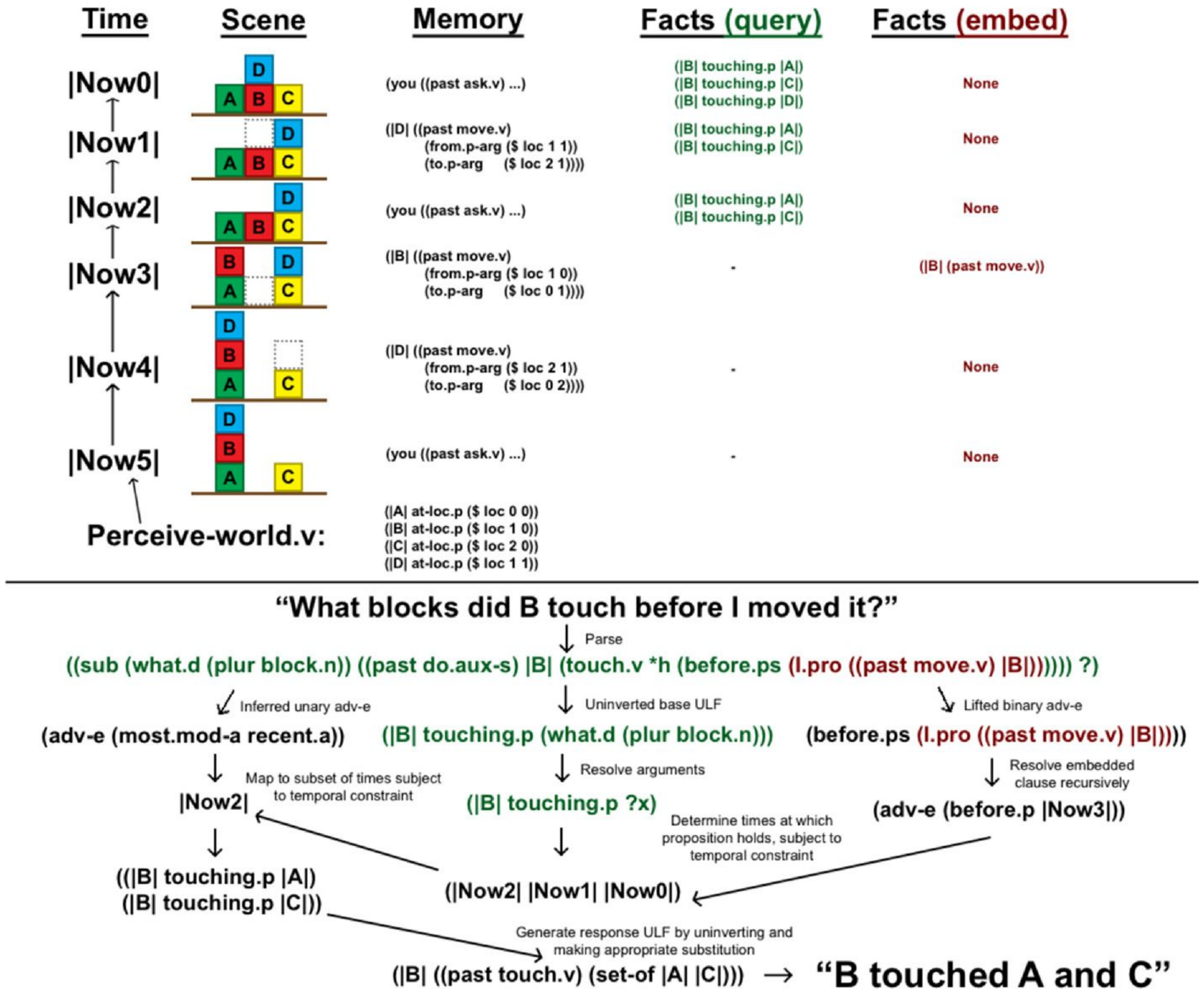
Wider Use of ULF



Spatial Reasoning

David





Time	Scene	Memory	Facts (query)	Facts (embed)
|Now0|	D A B C	(you ((past ask.v) ...)	(|B| touching.p |A|) (|B| touching.p |C|) (|B| touching.p |D|)	None
|Now1|	D A B C	(|D| ((past move.v) (from.p-arg ($ loc 1 1)) (to.p-arg ($ loc 2 1))))	(|B| touching.p |A|) (|B| touching.p |C|)	None
|Now2|	D A B C	(you ((past ask.v) ...)	(|B| touching.p |A|) (|B| touching.p |C|)	None
|Now3|	B D A C	(|B| ((past move.v) (from.p-arg ($ loc 1 0)) (to.p-arg ($ loc 0 1))))	-	(|B| (past move.v))
|Now4|	D B A C	(|D| ((past move.v) (from.p-arg ($ loc 2 1)) (to.p-arg ($ loc 0 2))))	-	None
|Now5|	D B A C	(you ((past ask.v) ...)	-	None

Perceive-world.v:
(|A| at-loc.p ($ loc 0 0))
(|B| at-loc.p ($ loc 1 0))
(|C| at-loc.p ($ loc 2 0))
(|D| at-loc.p ($ loc 1 1))

"What blocks did B touch before I moved it?"

Parse

((sub (what.d (plur block.n)) ((past do.aux-s) |B| (touch.v *h (before.ps (I.pro ((past move.v) |B|)))))) ?)

Inferred unary adv-e

(adv-e (most.mod-a recent.a))

Map to subset of times subject to temporal constraint

|Now2|

((|B| touching.p |A|)
(|B| touching.p |C|))

Uninverted base ULF

(|B| touching.p (what.d (plur block.n)))

Resolve arguments

(|B| touching.p ?x)

Determine times at which proposition holds, subject to temporal constraint

(|Now2| |Now1| |Now0|)

Lifted binary adv-e

(before.ps (I.pro ((past move.v) |B|)))

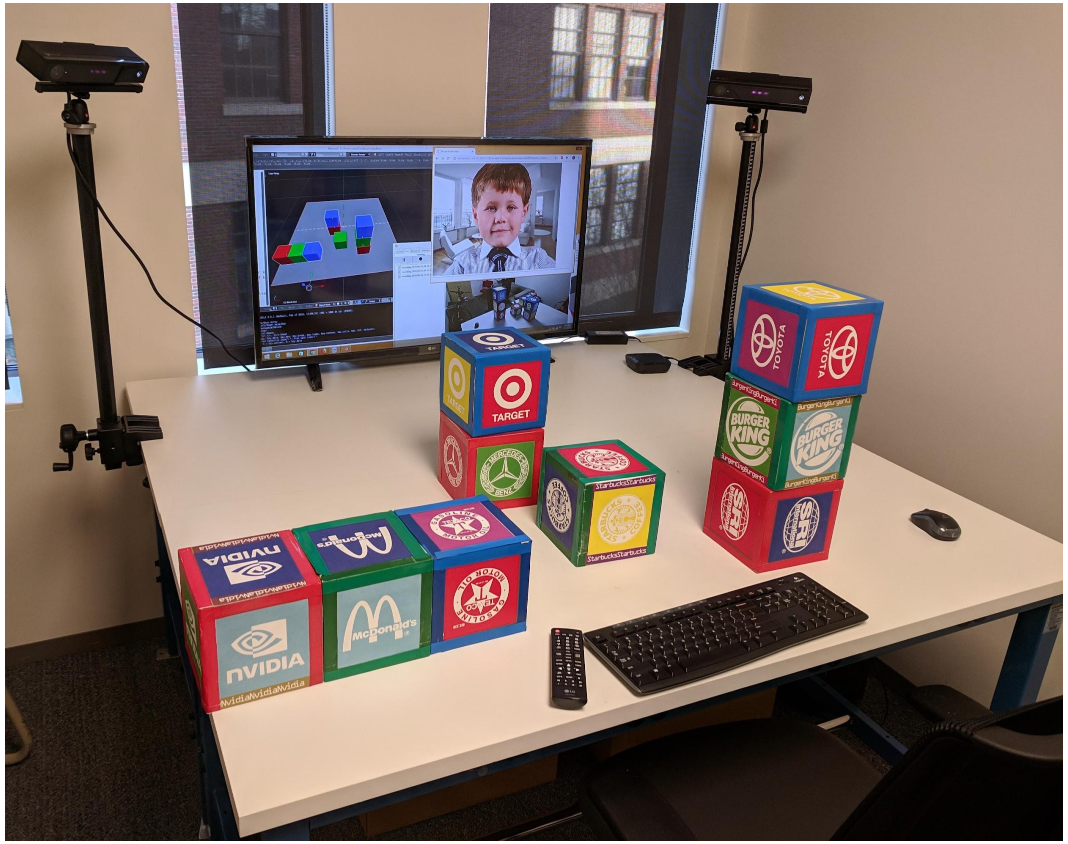
Resolve embedded clause recursively

(adv-e (before.p |Now3|))

Generate response ULF by uninverting and making appropriate substitution

(|B| ((past touch.v) (set-of |A| |C|))) → "B touched A and C"

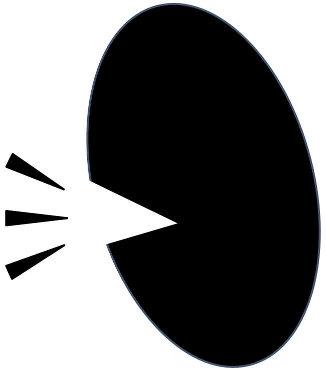
David



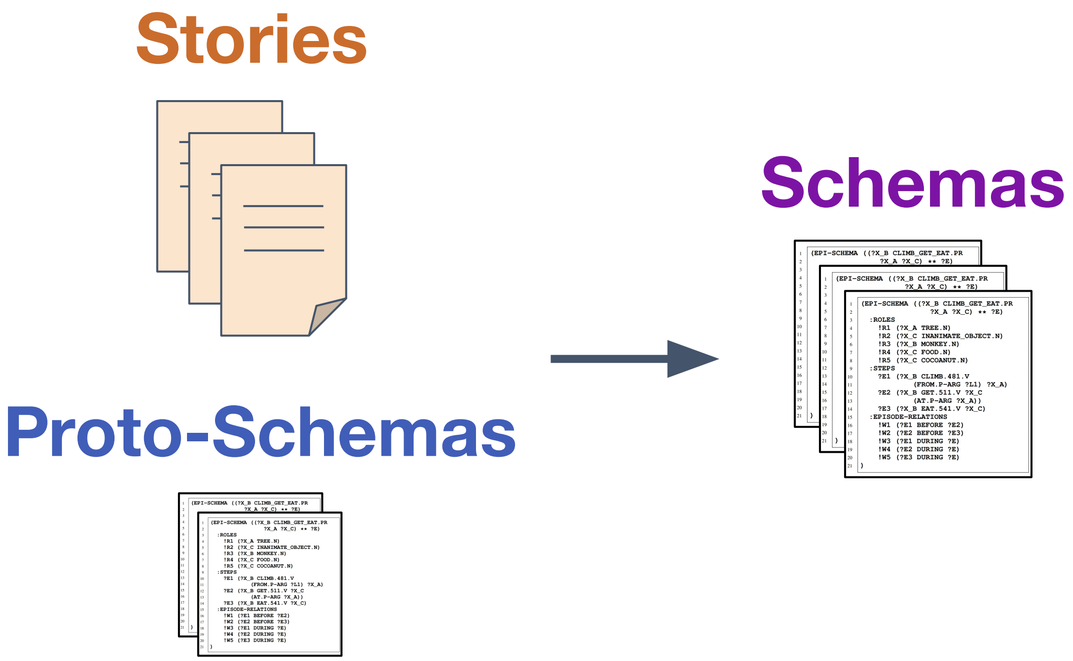
Hey David, what block is next to Target block?

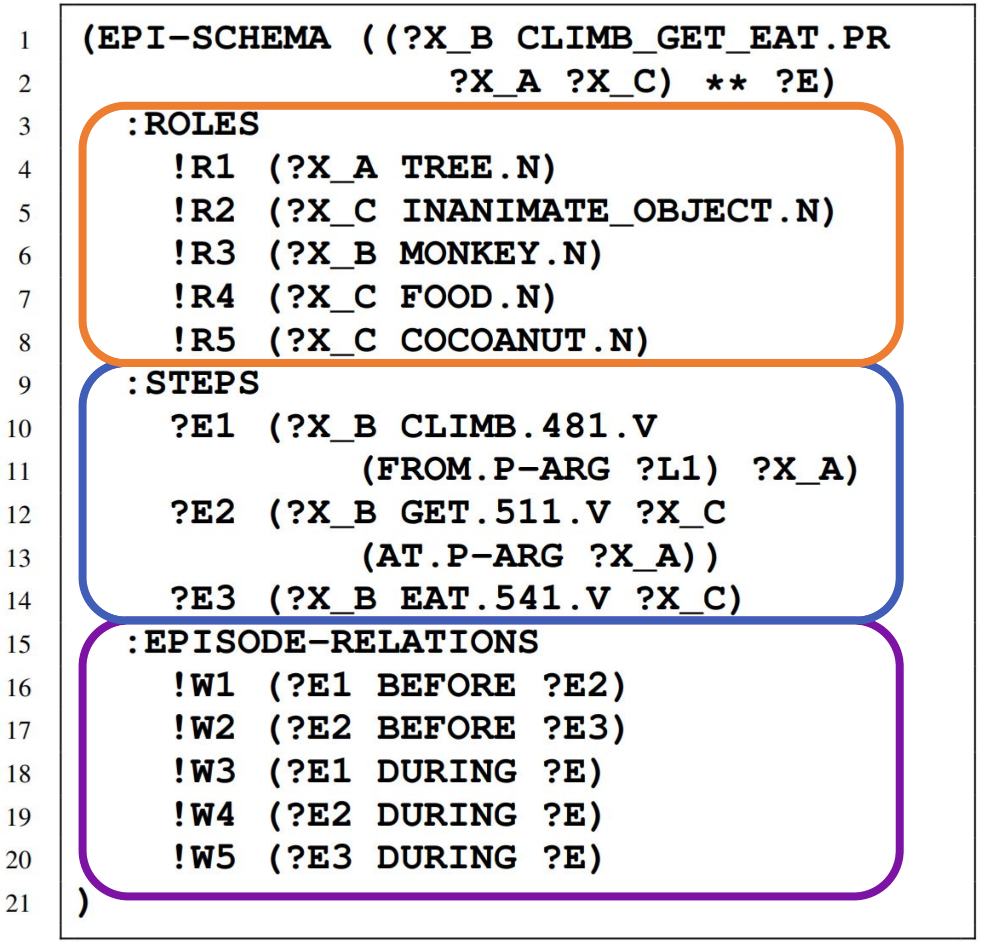
~~The Starbucks block is next to the Target block~~

There are no blocks next to the Starbucks block



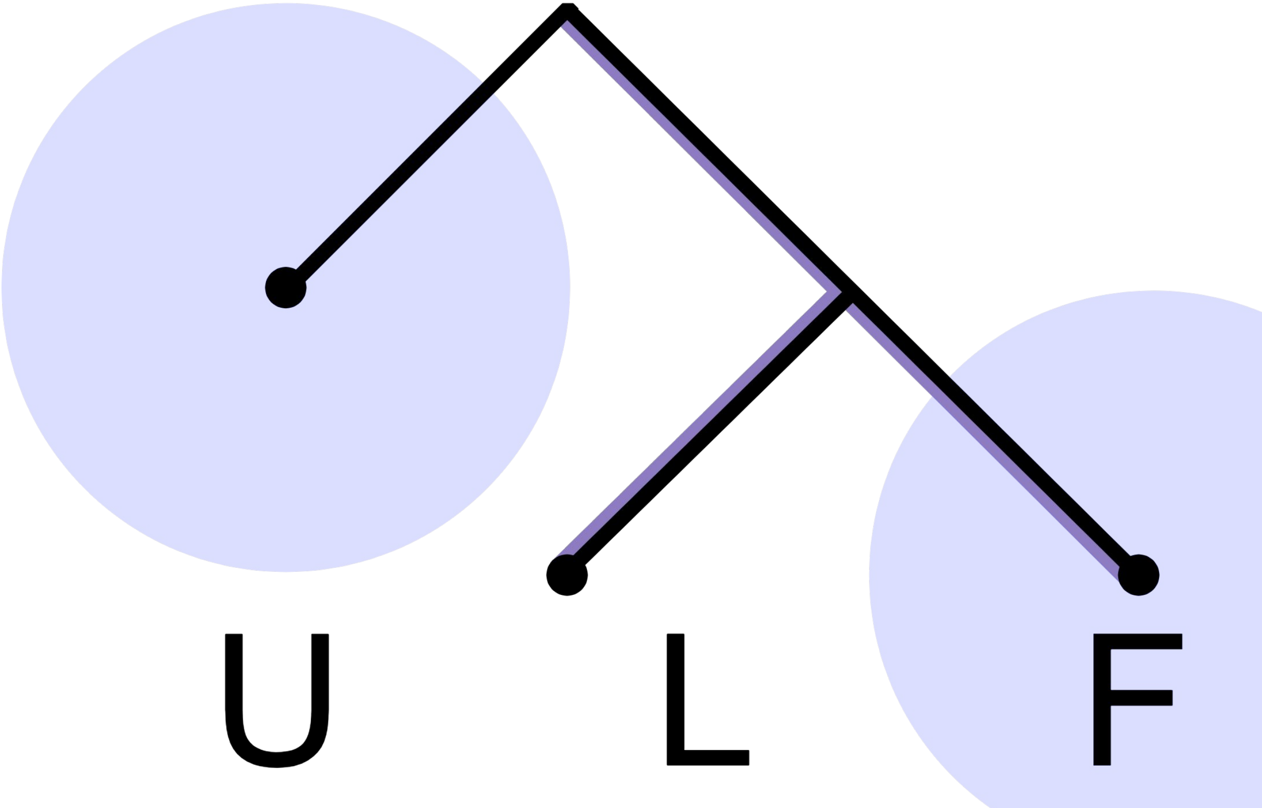
Schema Learning





```
(EPI-SCHEMA ((?X_B CLIMB_GET_EAT.PR
                    ?X_A ?X_C) ** ?E)
  :ROLES
    !R1 (?X_A TREE.N)
    !R2 (?X_C INANIMATE_OBJECT.N)
    !R3 (?X_B MONKEY.N)
    !R4 (?X_C FOOD.N)
    !R5 (?X_C COCOANUT.N)
  :STEPS
    ?E1 (?X_B CLIMB.481.V
              (FROM.P-ARG ?L1) ?X_A)
    ?E2 (?X_B GET.511.V ?X_C
              (AT.P-ARG ?X_A))
    ?E3 (?X_B EAT.541.V ?X_C)
  :EPISODE-RELATIONS
    !W1 (?E1 BEFORE ?E2)
    !W2 (?E2 BEFORE ?E3)
    !W3 (?E1 DURING ?E)
    !W4 (?E2 DURING ?E)
    !W5 (?E3 DURING ?E)
)
```

Conclusion

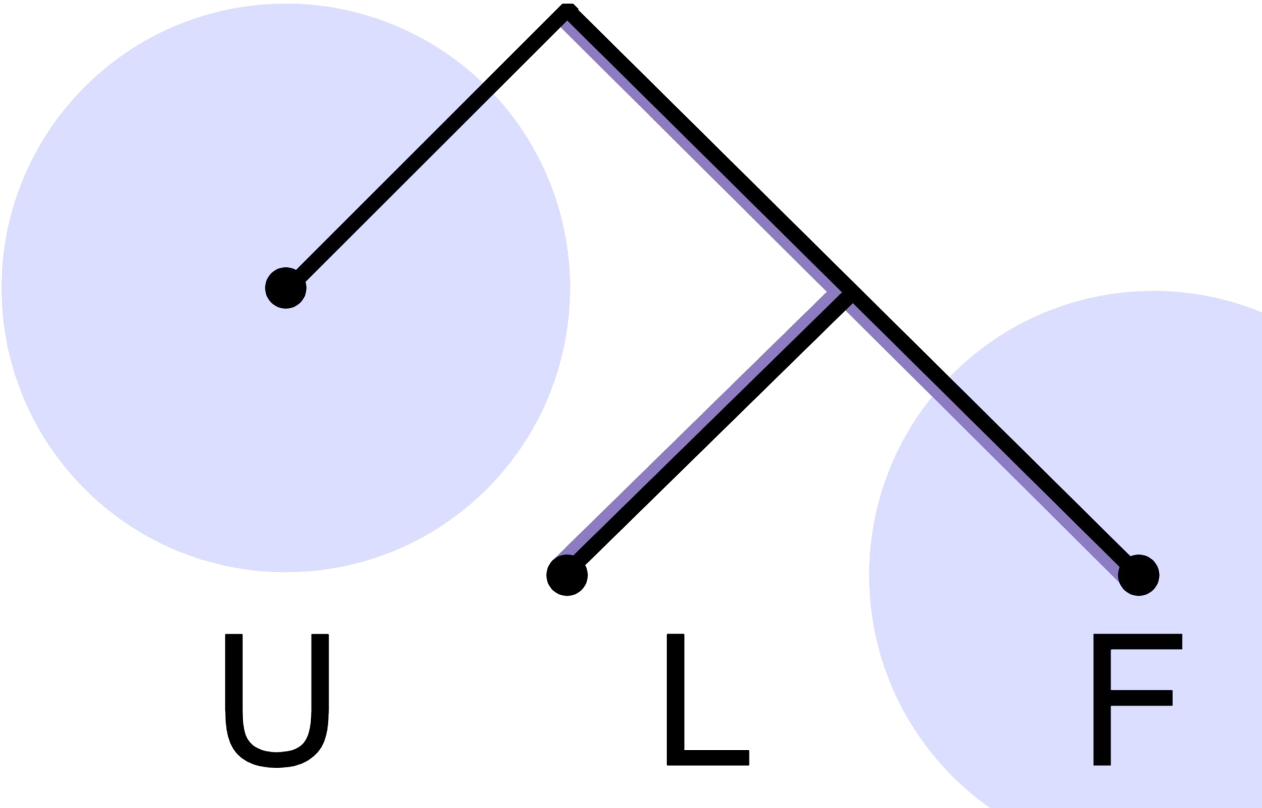


ULF Summarized

Type system + syntax for easy access expressive semantics. This enables

- Sufficient data collection *speed* and *consistency*
- *Parsability* with modest data size
- Syntax-related *inferences*
- Use in larger language interfacing systems

Thanks!



Funded in part by multiple NSF grants and a University of Rochester Sproull Fellowship