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?
To understand language
1. parse the structure
2. relate to world knowledge
3. consider the participants

“Adam placed John under arrest.”
Distributional or Symbolic

\[ \text{Abc} \]
\[ \Omega \]
\[ x = y \]
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
- predicates
- identity
- generalized quantifiers
- modification
- reification
- event reference
- comparatives

\[ \text{FOL} \]

\{ \]

- most, few, many, no, at most 10
- very, gracefully, nearly, possibly

- Beauty is subjective. That exoplanets exist is now certain.
- Many children had not been vaccinated against measles; this situation caused sporadic outbreaks of the disease.
- Doorways are taller than most people
Proposal

Bridge the gap with a type system + ambiguity

\[ D + (D \rightarrow T) = T \]

“Spot runs”

“Spot” “runs”

“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

\[
(\exists e: [e \text{ at-about Now}]
\quad [[\text{Gene want1.v}]
\quad (ka (\lambda x: [[x \text{ dance1.v}]) \land
\quad (\forall y: [[y \text{ shoes.n}]) \land
\quad [y \text{ poss-by Gene} \land
\quad [x \text{ in-wear y}])]])]) \ast e]\]

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

\[
(\exists e: [e \text{ at-about Now}]
[[\text{Gene want1.v}]
(ka (\lambda x: [[x \text{ dance1.v}] \land
(\nu y: [[y \text{ shoes.n}] \land
[y \text{ poss-by Gene}] \land
[x \text{ in-wear y}])))]) ** e])
\]

### Unscoped Logical Form

\[
(\text{i.pro ((pres want.v)}
\text{ (to (dance.v)}
\text{(adv-a (in.p (my.d ((mod-n new.a)}
\text{(plur shoe.n))})})})
\]

1. **Retain ambiguity of**
   - a. *scopes*
   - b. *word sense*
   - c. *anaphora*
   - d. *event relations*

2. **Maintain semantic coherence**

3. **Reflect syntactic structure**
ULF & Syntax

Word Order

Grammatical Structure

Part-of-Speech

“Adam placed John under arrest.”

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

verb

preposition

noun
ULF & Semantics

Semantic Types

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

Basic Ontological Types

- \(D\) Domain
- \(S\) Situations
- \(2\) Truth-value

Monadic Predicate

\(N : (D \rightarrow (S \rightarrow 2))\)
ULF & Semantics

“*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** \((N \rightarrow D): \text{the}.d\)

**Modifier Constructor** \((N \rightarrow (N \rightarrow N)): \text{adv-a}\)

**Predicate modifier** \((N \rightarrow N): \text{nearly}.adv-a\)

**Sentence reifier** \(((S \rightarrow 2) \rightarrow D): \text{that}\)

**Basic Ontological Types**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(D)</td>
<td>Domain</td>
</tr>
<tr>
<td>(S)</td>
<td>Situations</td>
</tr>
<tr>
<td>(2)</td>
<td>Truth-value</td>
</tr>
</tbody>
</table>

**Monadic Predicate**

\(N: (D \rightarrow (S \rightarrow 2))\)
"I want to dance in my new shoes"

Episodic Logic

(∃e: [e at-about Now]
  [[Gene want1.v
    (ka (λx: [[x dance1.v] ∧
       (iy: [[y shoes.n] ∧
        [y poss-by Gene] ∧
        [x in-wear y]]]))]] ** e])

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

\[
(\exists e: [e \text{ at-about Now}]
[[\text{Gene want1.v}
(\text{ka (\lambda x: [[x \text{ dance1.v}] \land
(\text{ny: [[y \text{ shoes.n}] \land
[y \text{ poss-by Gene}] \land
[x \text{ in-wear y]])}])}] \land
[x \text{ in-wear y}]])]) \land
[e])
\]

Unscoped Logical Form

\[
(i.pro ((\text{pres want.v})
(to (dance.v
(\text{adv-a (in.p (my.d ((\text{mod-n new.a})
(plur shoe.n))}})
\]

Dataset & Parser
Human ULF annotations

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

"She wants to eat the cake"

(she.pro ((pres want.v) (to (eat.v (the.d cake.n)))))
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**
- k
- ka
- that
- adv-a

**Operand**
- noun predicates (k gold.n)
- verb predicates (ka (run.v quickly.adv-a))
- sentences (that (i.pro (past win.v)))
- any predicates (adv-a (for.p you.pro))
“ran” \rightarrow \text{run}.v \\
“refreshed” \rightarrow \text{refreshed}.a \\
“Coke” \rightarrow |\text{Coke}|.n \\

dog.n \rightarrow (k \text{dog}.n) \\
quick.a \rightarrow (\text{adv}-a \text{quick}.a) \\
(i.\text{pro (past win}.v)) \rightarrow \text{that} (\text{that} (i.\text{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?
"Adam placed John under arrest"

Labeled ULF + Alignment

Oracle

Parsing Action Sequence

WordGen → Name → Suffix(null) → Push(1) → NoArc → NoPromote → NoPop → WordGen → Lemma → Suffix(v) → ...
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
Word Sequence
GloVe + RoBERTa + CharCNN + lemmas + POS + NER
Symbol Sequence
Symbol + CharCNN (of aligned word)
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
**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)))
“what did you buy?”

“did you buy what”

“you did buy something”

(you.pro ((past do.aux-s) (buy.v something.pro)))
"what did you buy?"

Un-inversion

"you did 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?”

“did you buy what?”

De-questioning

“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

**Recall**
Get human inferences for a sample of sentences and check coverage that the automatic inferences achieve
- Annotators are trained for these phenomena

127 inferences
698 inferences
406 sentences
Precision Evaluation

“How soon can you get that done?”

“Human”

“Automatic”

Inferred Sentences

Correct 68.5%

Contextual 15.0%

Incorrect 16.5%

Grammatical 78.0%
Recall Evaluation

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**

   \[
   (i.\text{pro} ((\text{pres} \text{ want.v}) \text{ and} \text{ cc} (\text{pres} \text{ expect.v})) \text{ you.pro} (\text{to} (\text{get.v that.pro done.a})))
   \]

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

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

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 (finished the survey on time)\footnote{\textcolor{red}{↑}}
⇒ Some delegates finished the survey

I never had a (girlfriend)\footnote{\textcolor{blue}{▼}} before
⇒ I never had a girlfriend taller than me before

Exactly 12 aliens read (magazines)\footnote{\textcolor{green}{●}}
⇔ Exactly 12 aliens read (news magazines)\footnote{\textcolor{purple}{●}}
“abelard sees a carp”
“every carp is a fish”

Lambek Derivations
Tableau-style proofs

Replace Lambek derivations
and sentences with ULFs

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

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

ULF
Sánchez Valencia’s System

Monotonicity

\[(\text{every } x)^\# \text{ is a } y, \, F(x^+), \, X \leftrightarrow Y\]

\[(\text{every } x)^\# \text{ is a } y, \, F(y), \, X \leftrightarrow Y\]

Inference 1  \( \text{abelard sees a carp, every carp is a fish} / \text{abelard sees a fish} \)

\[ \text{abe see a carp, every carp is a fish } \quad \text{abe see a fish} \]

\[ \text{abe sees } (\text{a carp})^\#, \, (\text{every carp})^\# \text{ is a fish } \quad \text{abe sees } (\text{a fish})^\# \]

\[ \text{abe sees } (\text{a carp}^+)\#, \, (\text{every carp})^\# \text{ is a fish } \quad \text{abe sees } (\text{a fish})^\# \]

\[ \text{abe sees } (\text{a fish})^\#, \, (\text{every carp})^\# \text{ is a fish } \quad \text{abe sees } (\text{a fish})^\# \]
Natural Logic with ULFs

“Abelard sees a carp”  1. $\langle \text{Abelardl (see.v (a.d carp.n))} \rangle$

“Every carp is a fish”  2. $\langle (\text{every.d carp.n}) (\text{be.v (= (a.d fish.n)))} \rangle$

“Abelard sees a fish”  3. $\langle \text{a.d x: (x carp.n)}^+ \rangle$

  $\langle \text{Abelardl (see.v x)}^+ \rangle^+$

  4. $\langle \text{Abelardl (see.v (a.d carp.n))} \rangle$

  5. $\langle \text{Abelardl (see.v (a.d fish.n))} \rangle$

Assumption
Assumption
SLF of 1.
Pol marking
UMI 2..4.

Monotonicity (UMI)

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

$$\phi[(\delta P2)]$$

where $\delta$ is a determiner.
Data
Some delegates finished the survey on time

Some delegates finished the survey

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

Why not a trained ULF parser?
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^-))^+)^+
```

Initial/Backup Polarity Marker

Stanford CoreNLP

ULF2English
Polarity Propagation

“Abe saw a dog without a tail”

“Every dog without a tail is a dog”

“Abe saw a dog”

“Mismatch!”

“Abe saw a dog without a tail”

“Mismatch!”

“Abe saw a dog without a tail”
1. **Monotonicity Substitution**

   *Every A is a B* \( \Rightarrow \) *S[B]*

2. **Conversion**

   *Some A is a B* \( \Leftrightarrow \) *Some B is an A*

3. **Conservativity**

   *DET As are Bs* \( \Leftrightarrow \) *DET As are As that/who are Bs*

4. **Equivalences**

   e.g., *Every dog is happy* \( \Leftrightarrow \) *All dogs are happy*
**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

- **Single-premise**: Baseline - 40, Us - 70
- **Multi-premise**: Baseline - 50, Us - 70
- **Overall**: Baseline - 45, Us - 75
Wider Use of ULF
Spatial Reasoning

David

```
<table>
<thead>
<tr>
<th>Time</th>
<th>Scene</th>
<th>Memory</th>
<th>Facts (query)</th>
<th>Facts (embed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Now0</td>
<td>D</td>
<td>(you) (past ask.v) ...</td>
<td>(B) touching.p (A)</td>
<td>None</td>
</tr>
<tr>
<td>Now1</td>
<td>D</td>
<td>(past move.v) (from-p-arg (5 loc 1 1)) (to-p-arg (5 loc 2 1))</td>
<td>(B) touching.p (A)</td>
<td>None</td>
</tr>
<tr>
<td>Now2</td>
<td>D</td>
<td>(you) (past ask.v) ...</td>
<td>(B) touching.p (A)</td>
<td>None</td>
</tr>
<tr>
<td>Now3</td>
<td>D</td>
<td>(B) touching.p (A) (B) touching.p (C)</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Now4</td>
<td>D</td>
<td>(you) (past ask.v) ...</td>
<td>(B) touching.p (A)</td>
<td>None</td>
</tr>
<tr>
<td>Now5</td>
<td>D</td>
<td>(B) touching.p (A)</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Perceive-world.v:

```

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

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

```
 adel (past aux-s) (what.d plur block.n)) (before.ps (l.pro (past move.v) [B])))
```

```
 adel (before.p [Now3])
```

"B touched A and C"
Schema Learning

Stories

Schemas

Proto-Schemas

(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 COCONUT.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