Generating Discourse Inferences from Unscoped Episodic Logical Formulas

Gene Louis Kim, Benjamin Kane, Viet Duong, Muskaan Mendiratta, Graeme McGuire, Sophie Sackstein, Georgiy Platonov, and Lenhart Schubert

Presented by: Gene Louis Kim
August 2019

UNIVERSITY *of* ROCHESTER

Introduction

Unscoped episodic logical form (ULF) is an expressive initial representation of Episodic Logic, but inference with it has not been demonstrated with it.

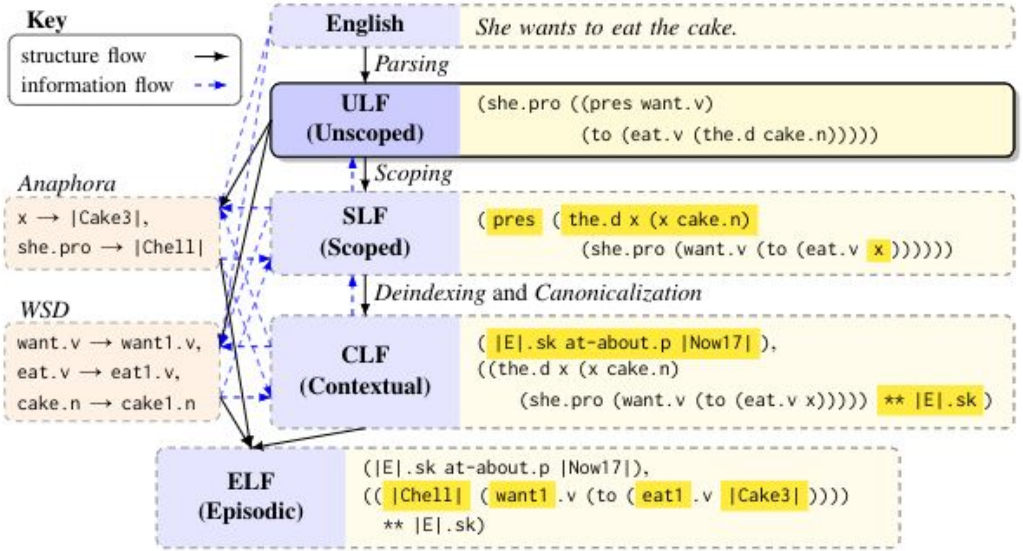
Introduction

Unscoped episodic logical form (ULF) is an expressive initial representation of Episodic Logic, but inference with it has not been demonstrated with it.

Unscoped {Episodic} Logical Form (ULF)

- An underspecified Episodic Logic (EL)
  - Extended FOL, closely matches expressivity of natural languages
    - modification, reification, generalized quantifiers, and more

Episodic Logic Pipeline



Key

structure flow →

information flow - -→

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 )
```

Anaphora

x → |Cake3|,
she.pro → |Chell|

WSD

want.v → want1.v,
eat.v → eat1.v,
cake.n → cake1.n

ELF (Episodic):

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

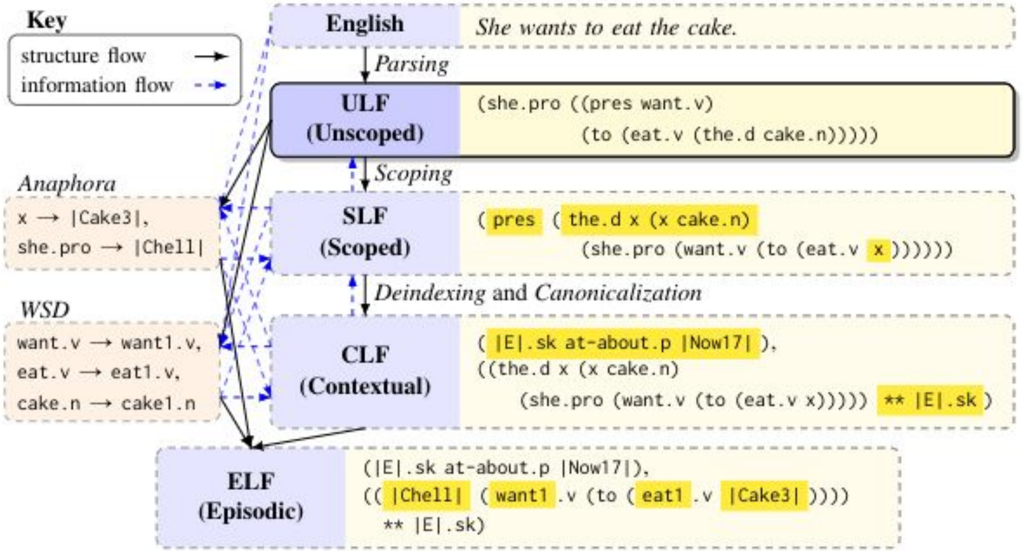
Introduction

Unscoped episodic logical form (ULF) is an expressive initial representation of Episodic Logic, but inference with it has not been demonstrated with it.

Unscoped {Episodic} Logical Form (ULF)

- An underspecified Episodic Logic (EL)
  - Extended FOL, closely matches expressivity of natural languages
    - modification, reification, generalized quantifiers, and more
- Starting point for EL parsing

Episodic Logic Pipeline



Key
structure flow →
information flow - - →
English
She wants to eat the cake.
Parsing
ULF
(Unscoped)
(she.pro ((pres want.v)
(to (eat.v (the.d cake.n)))))
Scoping
Anaphora
x → |Cake3|,
she.pro → |Chell|
SLF
(Scoped)
(pres (the.d x (x cake.n)
(she.pro (want.v (to (eat.v x))))))
Deindexing and Canonicalization
WSD
want.v → want1.v,
eat.v → eat1.v,
cake.n → cake1.n
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)

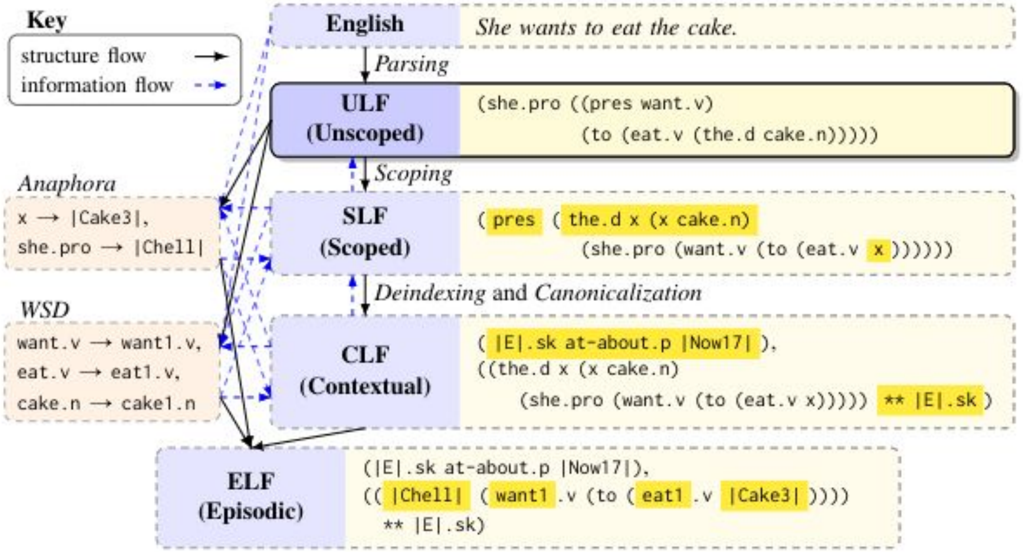
Introduction

Unscoped episodic logical form (ULF) is an expressive initial representation of Episodic Logic, but inference with it has not been demonstrated with it.

Unscoped {Episodic} Logical Form (ULF)

- An underspecified Episodic Logic (EL)
  - Extended FOL, closely matches expressivity of natural languages
    - modification, reification, generalized quantifiers, and more
- Starting point for EL parsing
- Enables situated inferences

Episodic Logic Pipeline



Key
structure flow →
information flow ⇢

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)

Anaphora
x → |Cake3|,
she.pro → |Chell|

WSD
want.v → want1.v,
eat.v → eat1.v,
cake.n → cake1.n

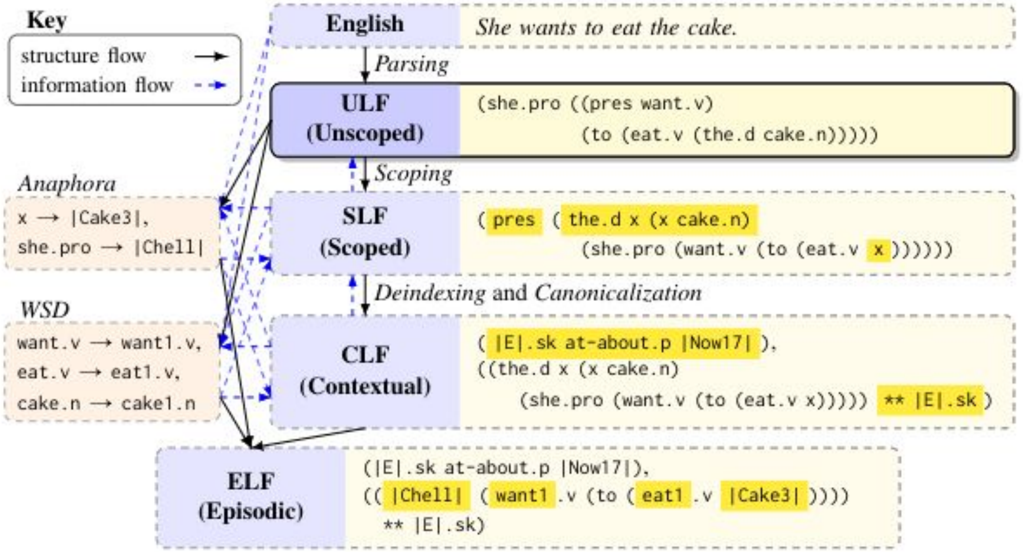
Introduction

Unscoped episodic logical form (ULF) is an expressive initial representation of Episodic Logic, but inference with it has not been demonstrated with it.

Unscoped {Episodic} Logical Form (ULF)

- An underspecified Episodic Logic (EL)
  - Extended FOL, closely matches expressivity of natural languages
    - modification, reification, generalized quantifiers, and more
- Starting point for EL parsing
- Enables situated inferences ?

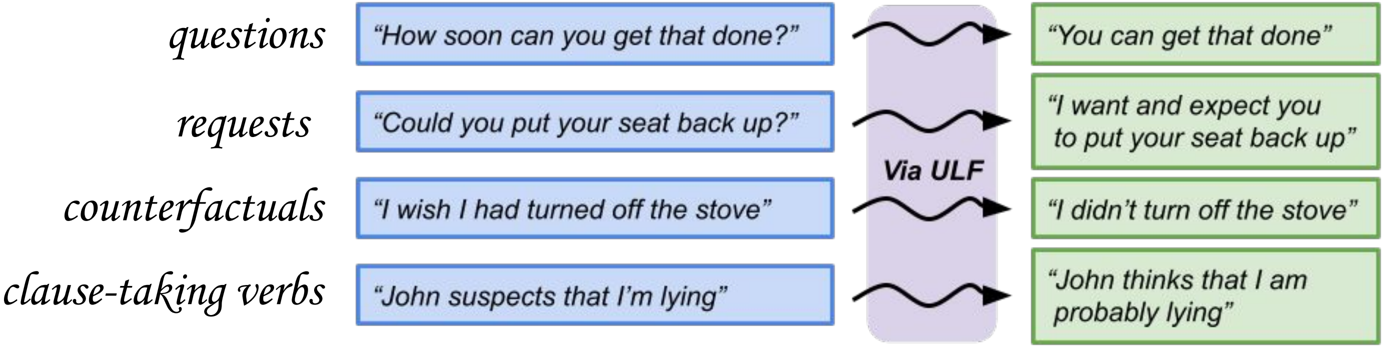
Episodic Logic Pipeline



Key
structure flow
information flow
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))))))
Anaphora
x → |Cake3|,
she.pro → |Chell|
WSD
want.v → want1.v,
eat.v → eat1.v,
cake.n → cake1.n
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)

Introduction

We select the following inference types for evaluation:



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”

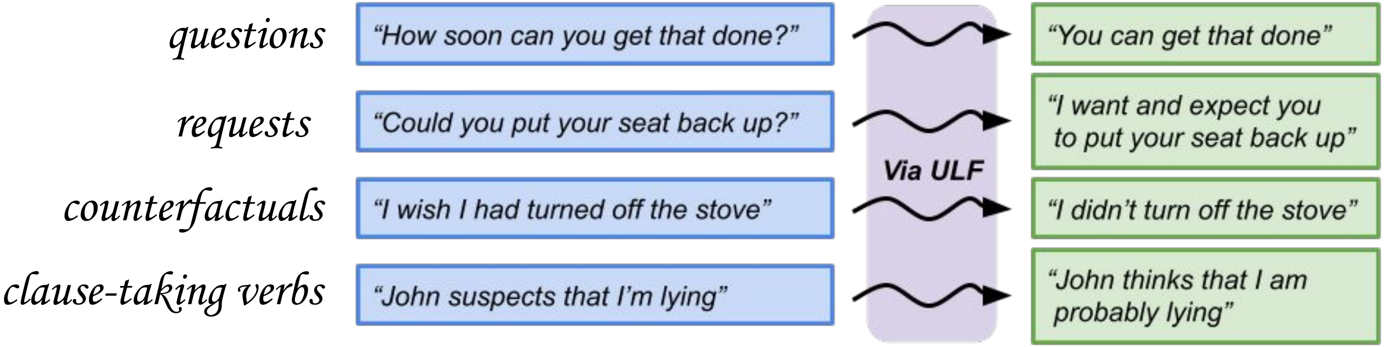
counterfactuals: “I wish I had turned off the stove” → “I didn’t turn off the stove”

clause-taking verbs: “John suspects that I’m lying” → “John thinks that I am probably lying”

Via ULF

Introduction

We select the following inference types for evaluation:

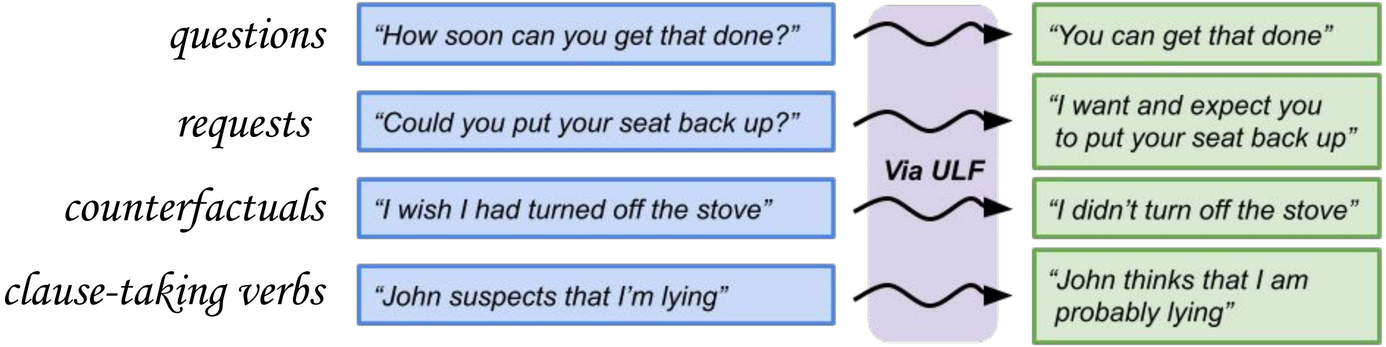

questions
"How soon can you get that done?"
"You can get that done"
requests
"Could you put your seat back up?"
"I want and expect you to put your seat back up"
counterfactuals
"I wish I had turned off the stove"
"I didn't turn off the stove"
clause-taking verbs
"John suspects that I'm lying"
"John thinks that I am probably lying"
Via ULF

Properties of Inferences

1. important for setting a natural discourse context

Introduction

We select the following inference types for evaluation:



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”

Via ULF

Properties of Inferences

1. important for setting a natural discourse context
2. structurally-oriented - we can avoid turning evaluation into a classification problem

ULF? (syntax)

A minimal step across from syntax to semantics in Episodic Logic

ULF? (syntax)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULF? (syntax)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULF

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

Syntax (simplified)

```
(S (NP Alice.nnp) (VP thinks.vbz
                 (SBAR that.rb (S (NP John.nnp) (ADVP nearly.rb) (VP fell.vbd)))))
```

ULF? (syntax)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULF

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

Syntax (simplified)

```
(S (NP Alice.nnp) (VP thinks.vbz
                      (SBAR that.rb (S (NP John.nnp) (ADVP nearly.rb) (VP fell.vbd)))))
```

Proper Nouns Verbs **Adverbs**

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULFs

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

Basic Ontological Types

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

Monadic Predicate $\mathcal{N} : \mathcal{D} \to (\mathcal{S} \to \mathbf{2})$

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULFs

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

Entity($\mathcal{D}$): |Alice|, |John|

Basic Ontological Types

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

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

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULFs

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

Basic Ontological Types

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

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

Entity($\mathcal{D}$): `|Alice|, |John|`

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): `think.v, fall.v`

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULFs

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

Basic Ontological Types

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

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

Entity($\mathcal{D}$): `|Alice|`, `|John|`

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): `think.v`, `fall.v`

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

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULFs

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

Basic Ontological Types

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

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

Entity($\mathcal{D}$): `|Alice|`, `|John|`

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): `think.v`, `fall.v`

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

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

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”

ULFs

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

Basic Ontological Types

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

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

Entity($\mathcal{D}$): `|Alice|`, `|John|`

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): `think.v`, `fall.v`

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

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

Also... *determiner, sentence modifier, connective, lambda abstract, predicate reifier*

Building ULF Inference Rules

1. Abstract away syntactic idiosyncrasies with interpretable predicates and functions

Building ULF Inference Rules

1. Abstract away syntactic idiosyncrasies with interpretable predicates and functions

Predicates

`verb-phrase?` - defined over the ULF semantic type system

`wh-word?` - defined as a list

Building ULF Inference Rules

1. Abstract away syntactic idiosyncrasies with interpretable predicates and functions

Predicates

`verb-phrase?` - defined over the ULF semantic type system

```
verb-phrase? :
    | lexical-verb?
```

`wh-word?` - defined as a list

Building ULF Inference Rules

1. Abstract away syntactic idiosyncrasies with interpretable predicates and functions

Predicates

`verb-phrase?` - defined over the ULF semantic type system

`wh-word?` - defined as a list

```
verb-phrase? :
     | lexical-verb?
     | (verb-phrase? term?)
```

Building ULF Inference Rules

1. Abstract away syntactic idiosyncrasies with interpretable predicates and functions

Predicates

`verb-phrase?` - defined over the ULF semantic type system

`wh-word?` - defined as a list

```
verb-phrase? :
     | lexical-verb?
     | (verb-phrase? term?)
     | (verb-modifier? verb-phrase?)
     | ...
```

Building ULF Inference Rules

1. Abstract away syntactic idiosyncrasies with interpretable predicates and functions

Predicates

`verb-phrase?` - defined over the ULF semantic type system

`wh-word?` - defined as a list

```
verb-phrase? :
     | lexical-verb?
     | (verb-phrase? term?)
     | (verb-modifier? verb-phrase?)
     | ...
```

Functions

`negate-verb-phrase!`

*“left the house” → “**did not** leave the house”*
*“could leave the house” → “could **not** leave the house”*

Building ULF Inference Rules

1. Abstract away syntactic idiosyncrasies with interpretable predicates and functions

Predicates

`verb-phrase?` - defined over the ULF semantic type system

`wh-word?` - defined as a list

```
verb-phrase? :
     | lexical-verb?
     | (verb-phrase? term?)
     | (verb-modifier? verb-phrase?)
     | ...
```

Functions

`negate-verb-phrase!`

*“left the house” → “**did not** leave the house”*
*“could leave the house” → “could **not** leave the house”*

`uninvert-sentence!`

*“did you leave already” → “**you did** leave already”*

Building ULF Inference Rules

2. Construct simple if-then rules

Building ULF Inference Rules

2. Construct simple if-then rules

if formula satisfies `contains-wh?` and ends with a question mark

“what did you buy?”

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

Building ULF Inference Rules

2. Construct simple if-then rules

if formula satisfies `contains-wh?` and ends with a question mark

strip the question mark

“what did you buy?”

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

↓

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

“what did you buy”

Building ULF Inference Rules

2. Construct simple if-then rules

if formula satisfies `contains-wh?` and ends with a question mark

strip the question mark

apply preprocessing markers

“what did you buy?”

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

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

↓

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

“did you buy what”

Building ULF Inference Rules

2. Construct simple if-then rules

if formula satisfies `contains-wh?` and ends with a question mark

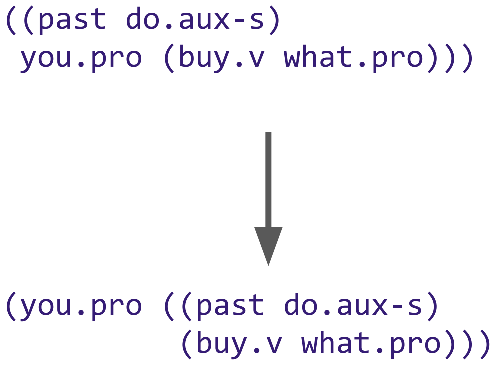
strip the question mark

apply preprocessing markers

apply `uninvert-sentence!`

“what did you buy?”

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



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

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

“you did buy what”

Building ULF Inference Rules

2. Construct simple if-then rules

if formula satisfies `contains-wh?` and ends with a question mark

strip the question mark

apply preprocessing markers

apply `uninvert-sentence!`

apply `wh2some!`

“what did you buy?”

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

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

↓

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

“you did buy something”

Building ULF Inference Rules

2. Construct simple if-then rules

if formula satisfies `contains-wh?` and ends with a question mark

strip the question mark

apply preprocessing markers

apply `uninvert-sentence!`

apply `wh2some!`

`infer-wh-question-presupposition`

“what did you buy?”

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

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

↓

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

“you did buy something”

Inference Evaluation

Inference Evaluation

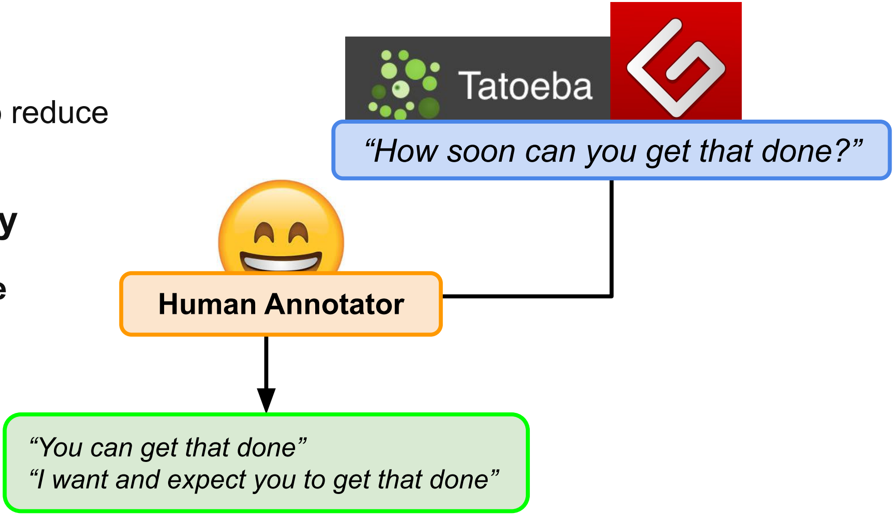
1. **Gather multi-genre dataset**
   a. **Sentences filtered by superficial patterns to reduce annotation overhead**

Tatoeba

“How soon can you get that done?”

Inference Evaluation

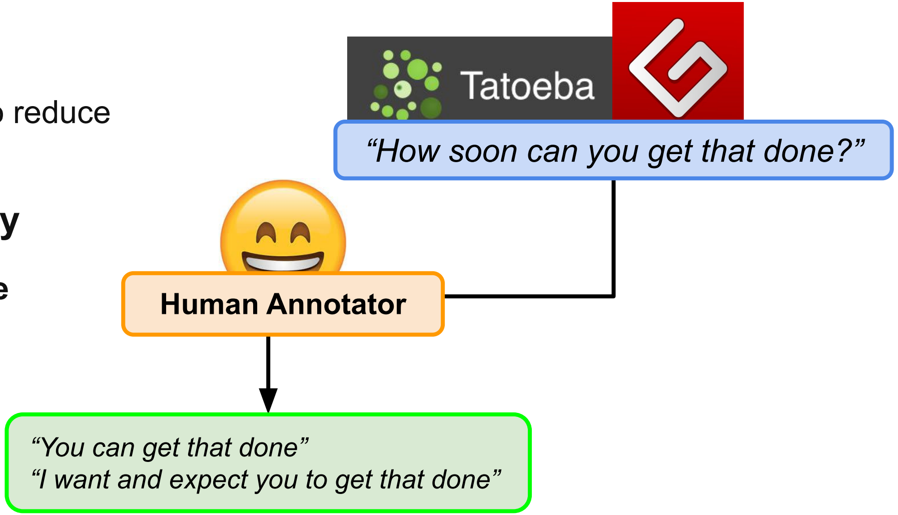
1. Gather multi-genre dataset
    a. Sentences filtered by superficial patterns to reduce annotation overhead
2. **Elicit human inferences in each category**
    a. **Reduced noise with structured guidance**



Tatoeba
"How soon can you get that done?"
Human Annotator
"You can get that done"
"I want and expect you to get that done"

Inference Evaluation

1. Gather multi-genre dataset
   a. Sentences filtered by superficial patterns to reduce annotation overhead
2. **Elicit human inferences in each category**
   a. Reduced noise with structured guidance



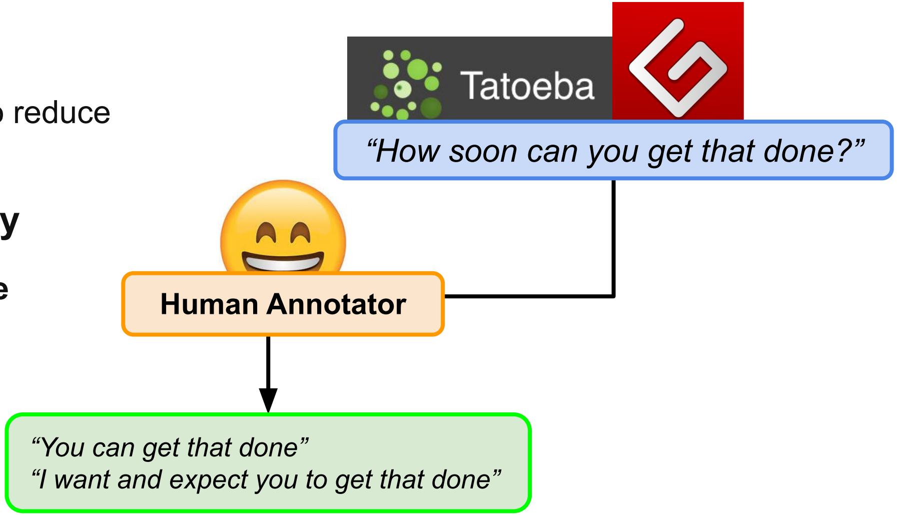
Tatoeba
"How soon can you get that done?"
Human Annotator
"You can get that done"
"I want and expect you to get that done"

<u>Human Inference Elicitation</u>

1. Select inference category
   a. *request, question, counterfactual, clause-taking*
2. Select inference structure
   a. e.g. (if `<x>` were `<pred>`, `<x>` would `<q>`)
      → (`<x>` is not `<pred>`)
3. Write fluent sentence corresponding to inference antecedent

Inference Evaluation

1. Gather multi-genre dataset
    a. Sentences filtered by superficial patterns to reduce annotation overhead
2. **Elicit human inferences in each category**
    a. **Reduced noise with structured guidance**



"How soon can you get that done?"

Human Annotator

"You can get that done"
"I want and expect you to get that done"

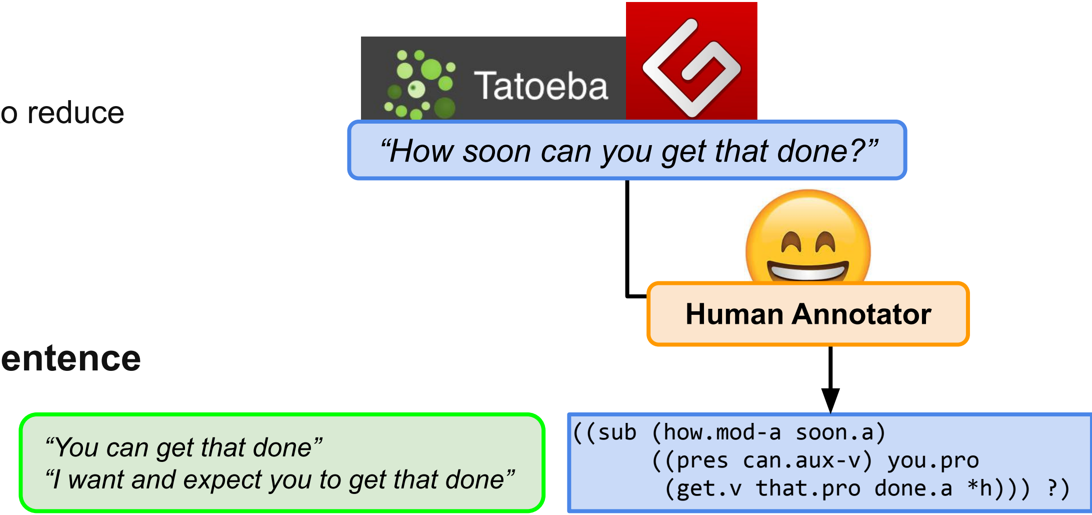
<u>Human Inference Elicitation</u>

1. Select inference category
    a. *request, question, counterfactual, clause-taking*
2. Select inference structure
    a. e.g. (if `<x>` were `<pred>`, `<x>` would `<q>`)
    → (`<x>` is not `<pred>`)
3. Write fluent sentence corresponding to inference antecedent

Dataset of 698 elicited inferences over 406 sentences.

Inference Evaluation

1. Gather multi-genre dataset
   a. Sentences filtered by superficial patterns to reduce annotation overhead
2. Elicit human inferences in each category
   a. Reduced noise with structured guidance
3. **Collect gold ULF annotations of each sentence**



"How soon can you get that done?"

Human Annotator

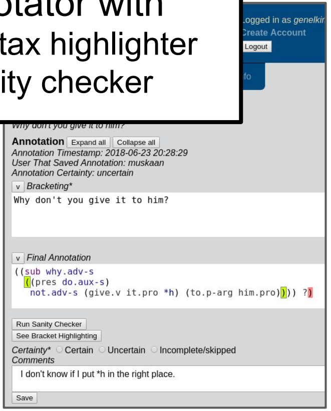
```
((sub (how.mod-a soon.a)
      ((pres can.aux-v) you.pro
       (get.v that.pro done.a *h))) ?)
```

"You can get that done"
"I want and expect you to get that done"

Inference Evaluation

1. Gather multi-genre dataset
   a. Sentences filtered by superficial patterns to reduce annotation overhead
2. Elicit human inferences in each category
   a. Reduced noise with structured guidance
3. **Collect gold ULF annotations of each sentence**

ULF Annotator with
- Syntax highlighter
- Sanity checker



Why don't you give it to him?

Annotation

Annotation Timestamp: 2018-06-23 20:28:29
User That Saved Annotation: muskaan
Annotation Certainty: uncertain

*Bracketing**

```
Why don't you give it to him?
```

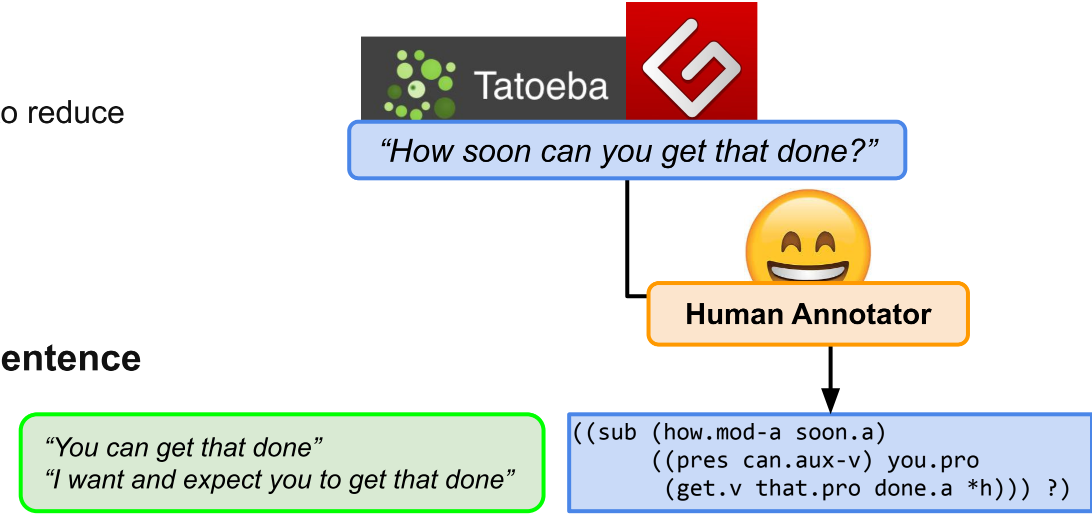
Final Annotation

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

*Certainty** ○ Certain ○ Uncertain ○ Incomplete/skipped

Comments

I don't know if I put *h in the right place.



“How soon can you get that done?”

Human Annotator

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

“You can get that done”
“I want and expect you to get that done”

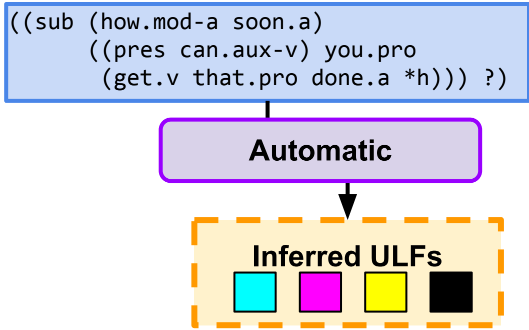
Inference Evaluation

1. Gather multi-genre dataset
    a. Sentences filtered by superficial patterns to reduce annotation overhead
2. Elicit human inferences in each category
    a. Reduced noise with structured guidance
3. Collect gold ULF annotations of each sentence
4. **Use inference rules to make conclusions**

Tatoeba

"How soon can you get that done?"

"You can get that done"
"I want and expect you to get that done"



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

Inference Evaluation

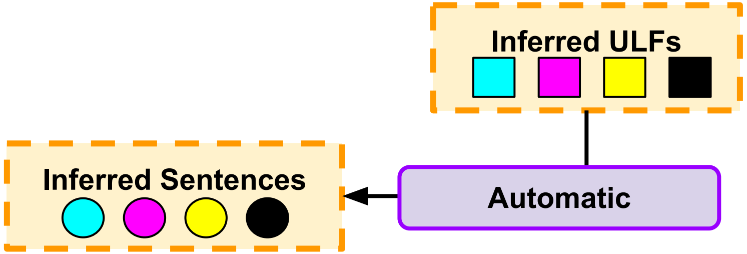
1. Gather multi-genre dataset
   a. Sentences filtered by superficial patterns to reduce annotation overhead
2. Elicit human inferences in each category
   a. Reduced noise with structured guidance
3. Collect gold ULF annotations of each sentence
4. Use inference rules to make conclusions
5. **Automatically generate English from ULF inferences**

Tatoeba

"How soon can you get that done?"

"You can get that done"
"I want and expect you to get that done"

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



Inference Evaluation

1. Gather multi-genre dataset
   a. Sentences filtered by superficial patterns to reduce annotation overhead
2. Elicit human inferences in each category
   a. Reduced noise with structured guidance
3. Collect gold ULF annotations of each sentence
4. Use inference rules to make conclusions
5. **Automatically generate English from ULF inferences**

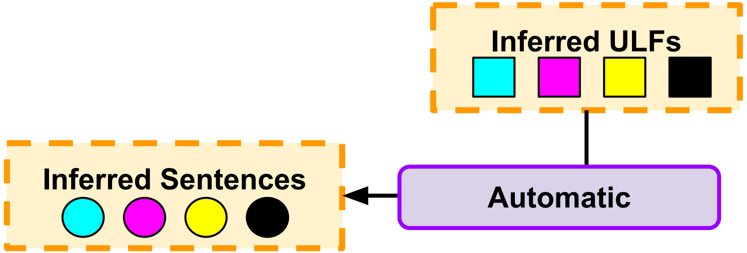
Tatoeba

“How soon can you get that done?”

“You can get that done”
“I want and expect you to get that done”

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

The ULF-to-English translation
1. Analyze the ULF type of each clause,
2. Incorporate morphological inflections based on the type analysis,
3. Filter out purely logical operators, and
4. Map logical symbols to surface form counterparts.

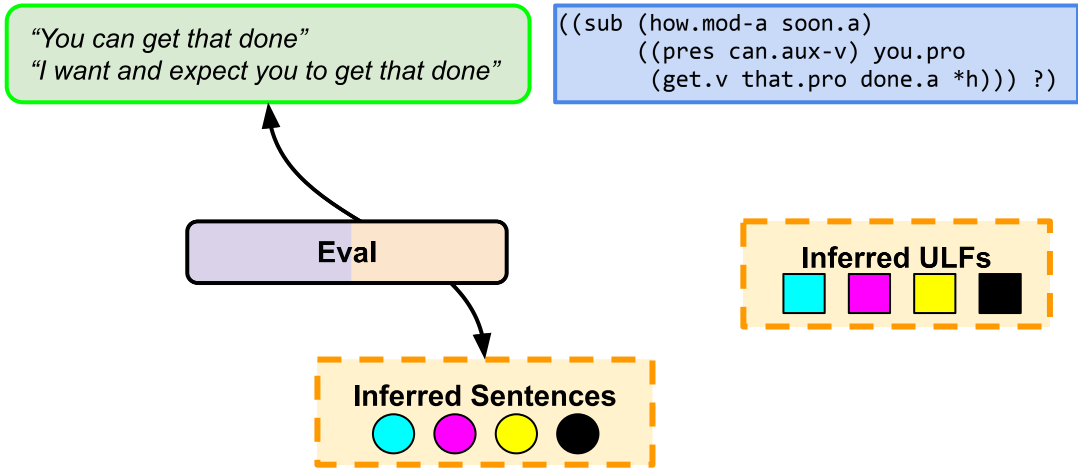


Inference Evaluation

1. Gather multi-genre dataset
    a. Sentences filtered by superficial patterns to reduce annotation overhead
2. Elicit human inferences in each category
    a. Reduced noise with structured guidance
3. Collect gold ULF annotations of each sentence
4. Use inference rules to make conclusions
5. Automatically generate English from ULF inferences
6. **Evaluate**
    a. **precision using human judgments**
    b. **recall using automatic matching**

Tatoeba

“How soon can you get that done?”

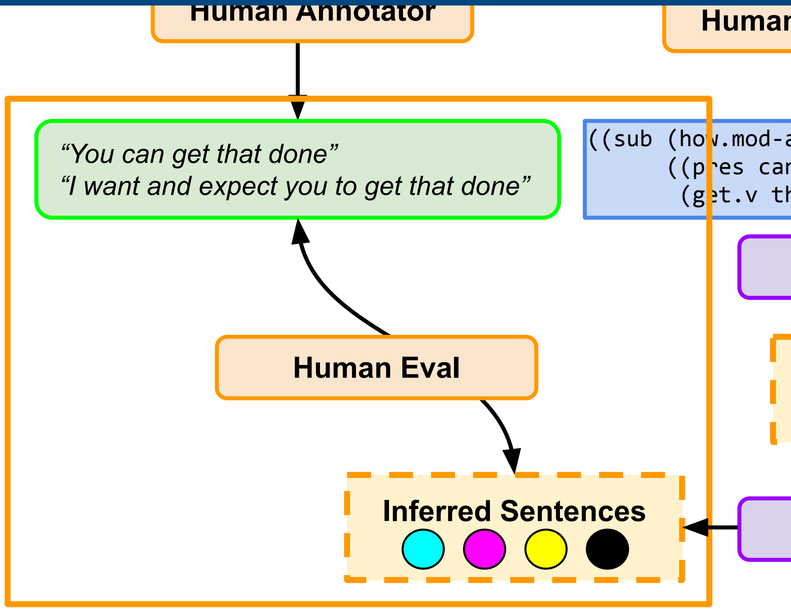


“You can get that done”
“I want and expect you to get that done”

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

Human Evaluation (Precision)

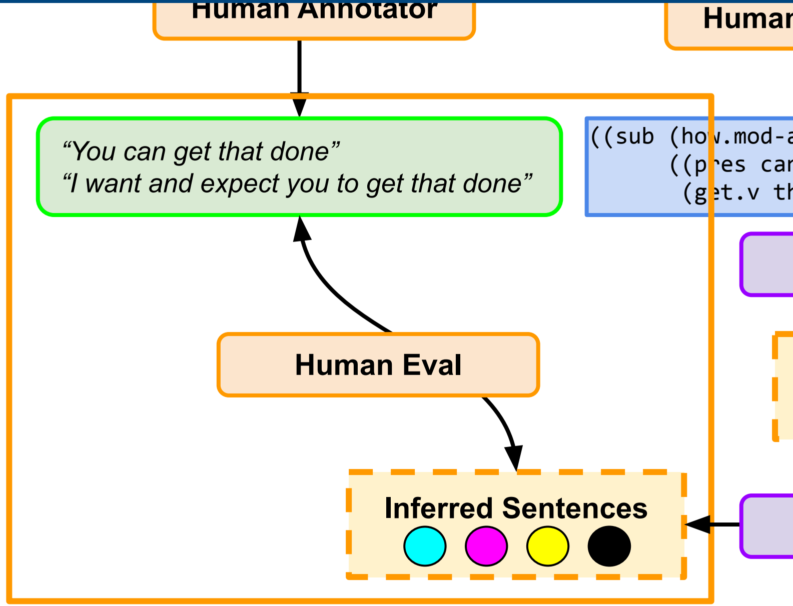
Human elicited inferences are incomplete, so precision is only measured with human evaluation.



Human Evaluation (Precision)

Human elicited inferences are incomplete, so precision is only measured with human evaluation.

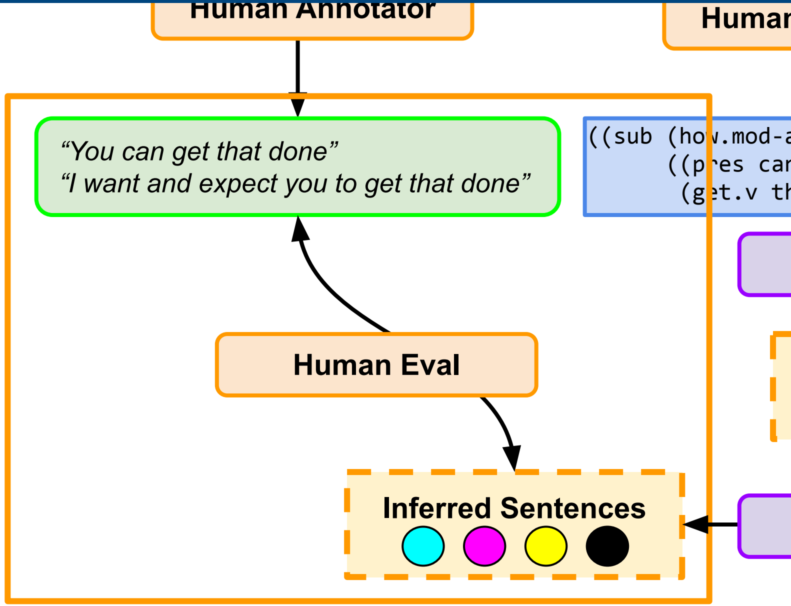
- Sample of 127 inferences
  - all *counterfactual* and *clause-taking* examples and random sampling of others



Human Evaluation (Precision)

Human elicited inferences are incomplete, so precision is only measured with human evaluation.

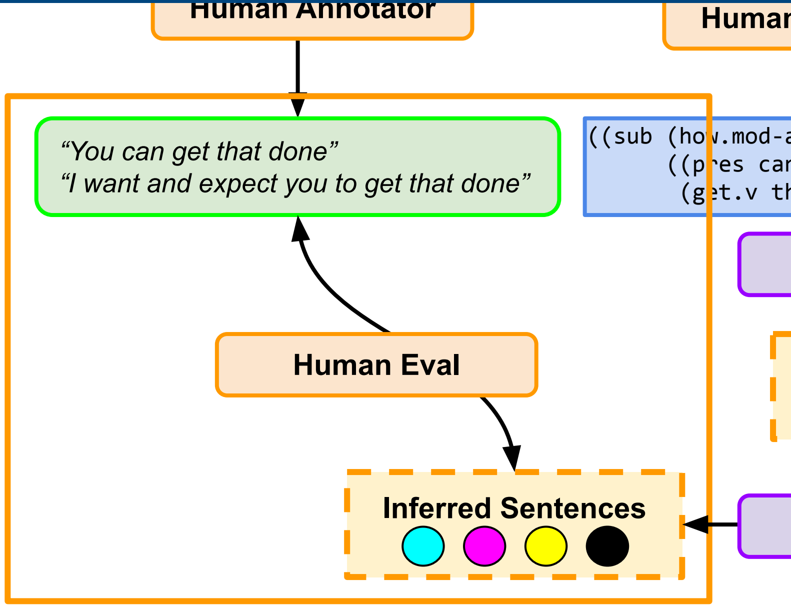
- Sample of 127 inferences
  - all *counterfactual* and *clause-taking* examples and random sampling of others
- 3 or 4 human judgments per inference



Human Evaluation (Precision)

Human elicited inferences are incomplete, so precision is only measured with human evaluation.

- Sample of 127 inferences
  - all *counterfactual* and *clause-taking* examples and random sampling of others
- 3 or 4 human judgments per inference
- 3 Categories: correct, incorrect, context-dependent



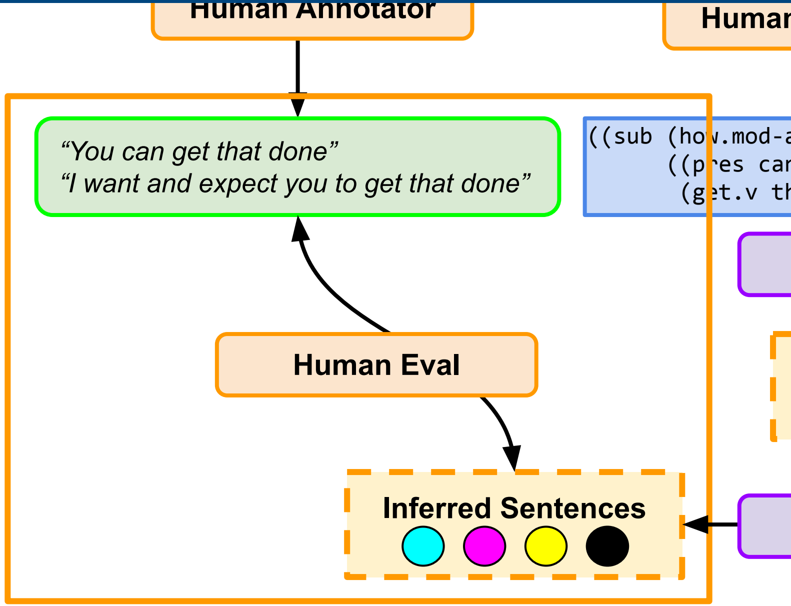
“You can get that done”
“I want and expect you to get that done”

```
((sub (how.mod-a
      ((pres can
       (get.v th
```

Human Evaluation (Precision)

Human elicited inferences are incomplete, so precision is only measured with human evaluation.

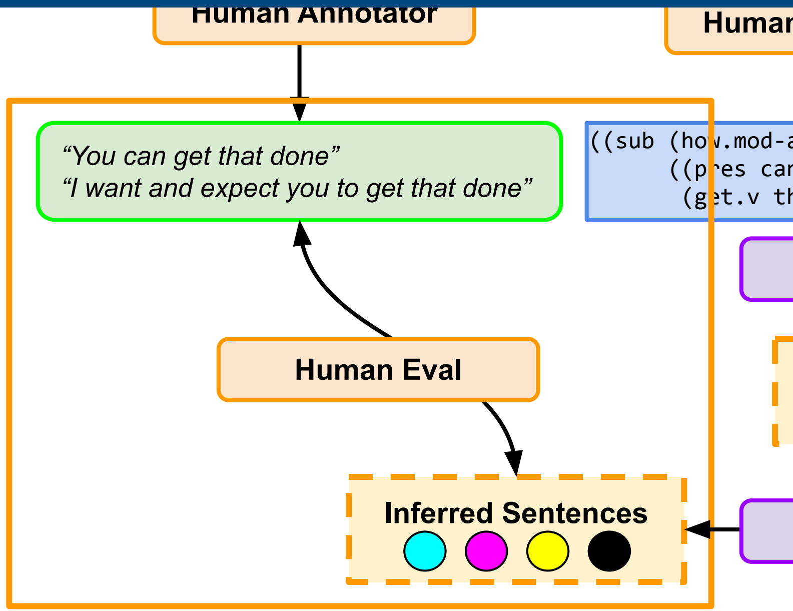
- Sample of 127 inferences
  - all *counterfactual* and *clause-taking* examples and random sampling of others
- 3 or 4 human judgments per inference
- 3 Categories: correct, incorrect, context-dependent
- Correctness(*) = reasonable inference in conversation, allowing for a bit of awkwardness in phrasing



Human Evaluation (Precision)

Human elicited inferences are incomplete, so precision is only measured with human evaluation.

- Sample of 127 inferences
  - all *counterfactual* and *clause-taking* examples and random sampling of others
- 3 or 4 human judgments per inference
- 3 Categories: correct, incorrect, context-dependent
- Correctness(*) = reasonable inference in conversation, allowing for a bit of awkwardness in phrasing
- Evaluated grammaticality



Human Annotator

Human

"You can get that done"
"I want and expect you to get that done"

```
((sub (how.mod-a
      ((pres car
       (get.v th
```

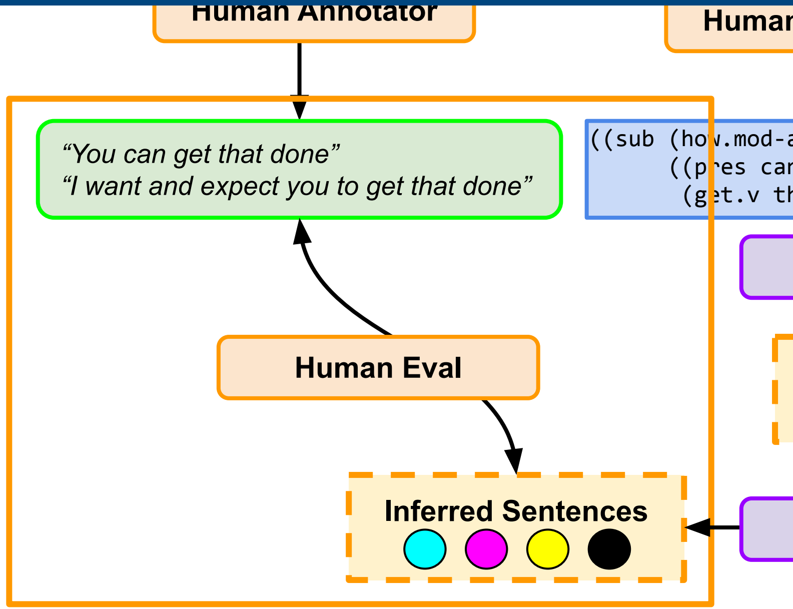
Human Eval

Inferred Sentences

Human Evaluation (Precision)

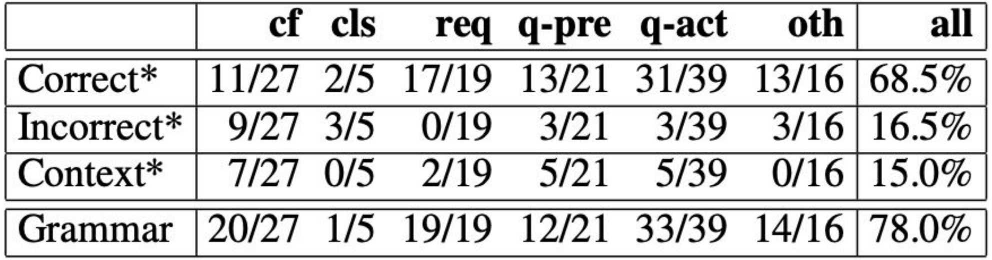
Human elicited inferences are incomplete, so precision is only measured with human evaluation.

- Sample of 127 inferences
  - all *counterfactual* and *clause-taking* examples and random sampling of others
- 3 or 4 human judgments per inference
- 3 Categories: correct, incorrect, context-dependent
- Correctness(*) = reasonable inference in conversation, allowing for a bit of awkwardness in phrasing
- Evaluated grammaticality



“You can get that done”
“I want and expect you to get that done”

```
((sub (how.mod-a
      ((pres can
       (get.v th
```



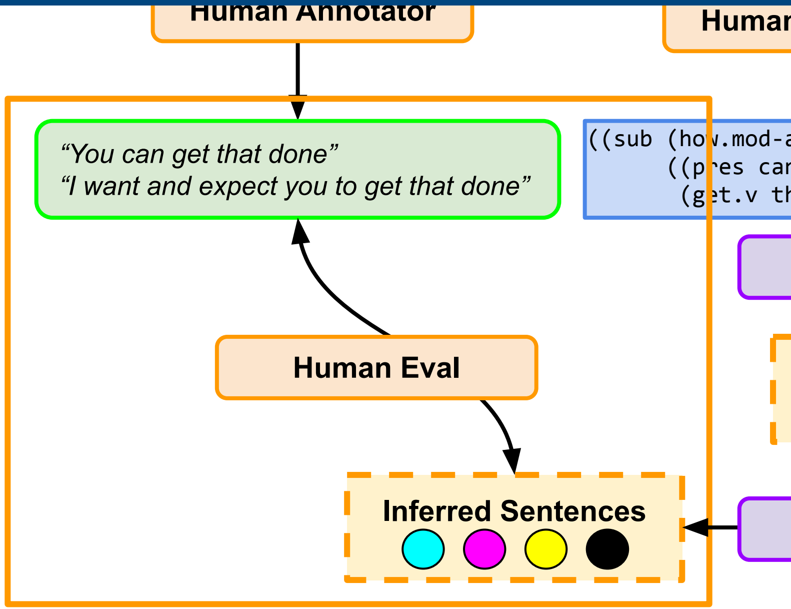
	cf	cls	req	q-pre	q-act	oth	all
Correct*	11/27	2/5	17/19	13/21	31/39	13/16	68.5%
Incorrect*	9/27	3/5	0/19	3/21	3/39	3/16	16.5%
Context*	7/27	0/5	2/19	5/21	5/39	0/16	15.0%
Grammar	20/27	1/5	19/19	12/21	33/39	14/16	78.0%

cf: *counterfactual*
cls: *clause-taking*
req: *request*
q-pre: *question presuppositional inferences*
q-act: *question act inferences*
oth: *other*

Human Evaluation (Precision)

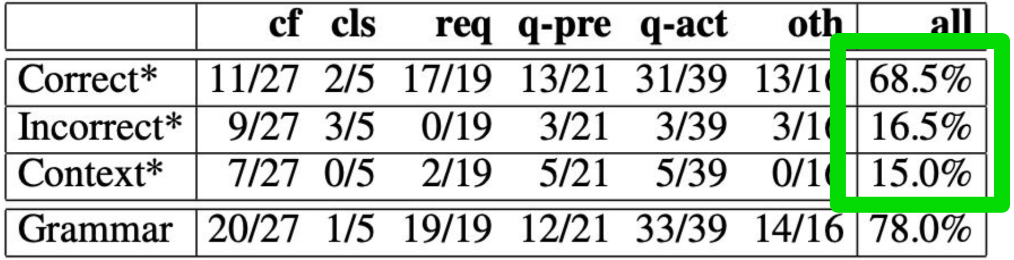
Human elicited inferences are incomplete, so precision is only measured with human evaluation.

- Sample of 127 inferences
  - all *counterfactual* and *clause-taking* examples and random sampling of others
- 3 or 4 human judgments per inference
- 3 Categories: correct, incorrect, context-dependent
- Correctness(*) = reasonable inference in conversation, allowing for a bit of awkwardness in phrasing
- Evaluated grammaticality



"You can get that done"
"I want and expect you to get that done"

```
((sub (ho .mod-a
      ((p es can
       (g t.v th
```



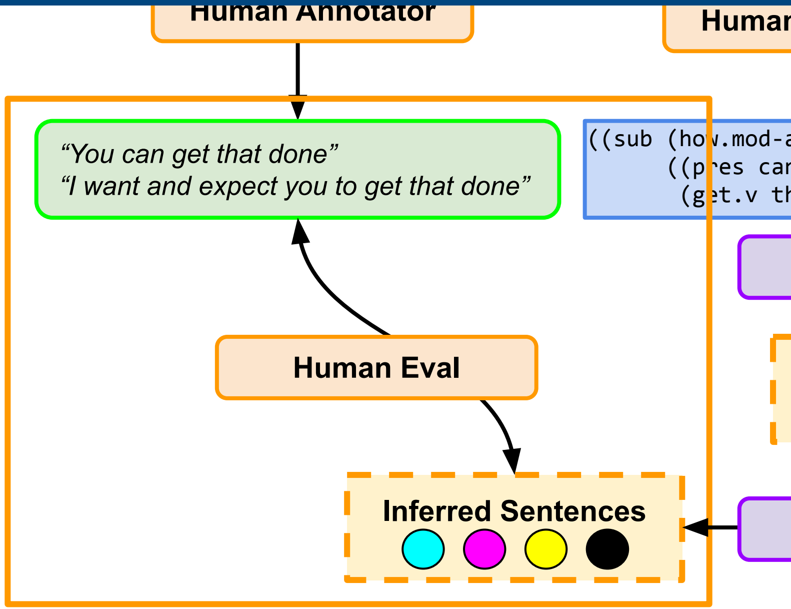
	cf	cls	req	q-pre	q-act	oth	all
Correct*	11/27	2/5	17/19	13/21	31/39	13/1	68.5%
Incorrect*	9/27	3/5	0/19	3/21	3/39	3/1	16.5%
Context*	7/27	0/5	2/19	5/21	5/39	0/1	15.0%
Grammar	20/27	1/5	19/19	12/21	33/39	14/16	78.0%

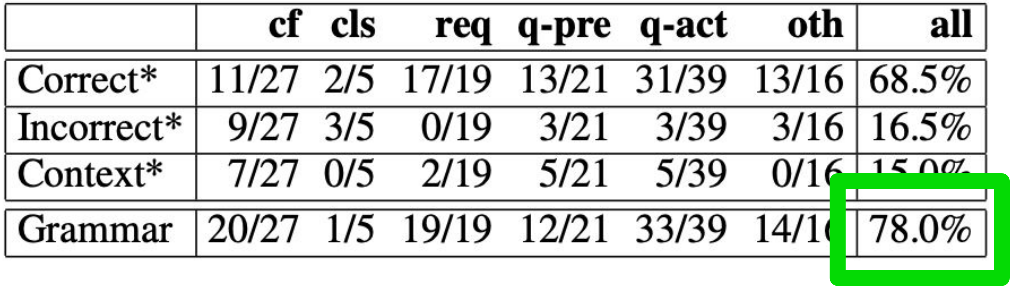
cf: *counterfactual*
cls: *clause-taking*
req: *request*
q-pre: *question presuppositional inferences*
q-act: *question act inferences*
oth: *other*

Human Evaluation (Precision)

Human elicited inferences are incomplete, so precision is only measured with human evaluation.

- Sample of 127 inferences
    - all *counterfactual* and *clause-taking* examples and random sampling of others
- 3 or 4 human judgments per inference
- 3 Categories: correct, incorrect, context-dependent
- Correctness(*) = reasonable inference in conversation, allowing for a bit of awkwardness in phrasing
- Evaluated grammaticality

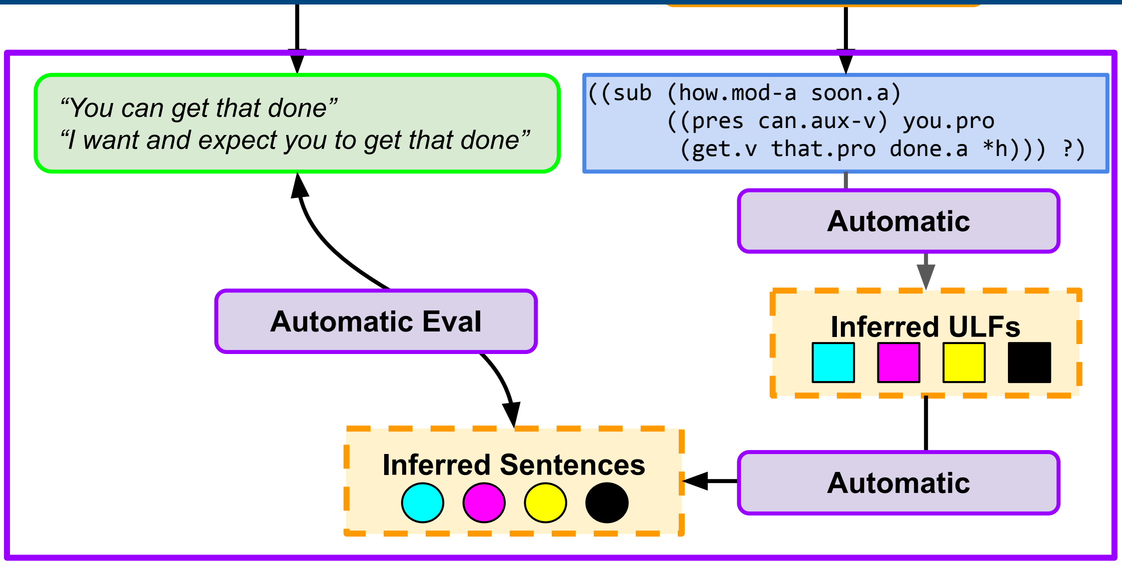




	cf	cls	req	q-pre	q-act	oth	all
Correct*	11/27	2/5	17/19	13/21	31/39	13/16	68.5%
Incorrect*	9/27	3/5	0/19	3/21	3/39	3/16	16.5%
Context*	7/27	0/5	2/19	5/21	5/39	0/16	15.0%
Grammar	20/27	1/5	19/19	12/21	33/39	14/16	78.0%

cf: *counterfactual*
cls: *clause-taking*
req: *request*
q-pre: *question presuppositional inferences*
q-act: *question act inferences*
oth: *other*

Automatic Evaluation (Recall)

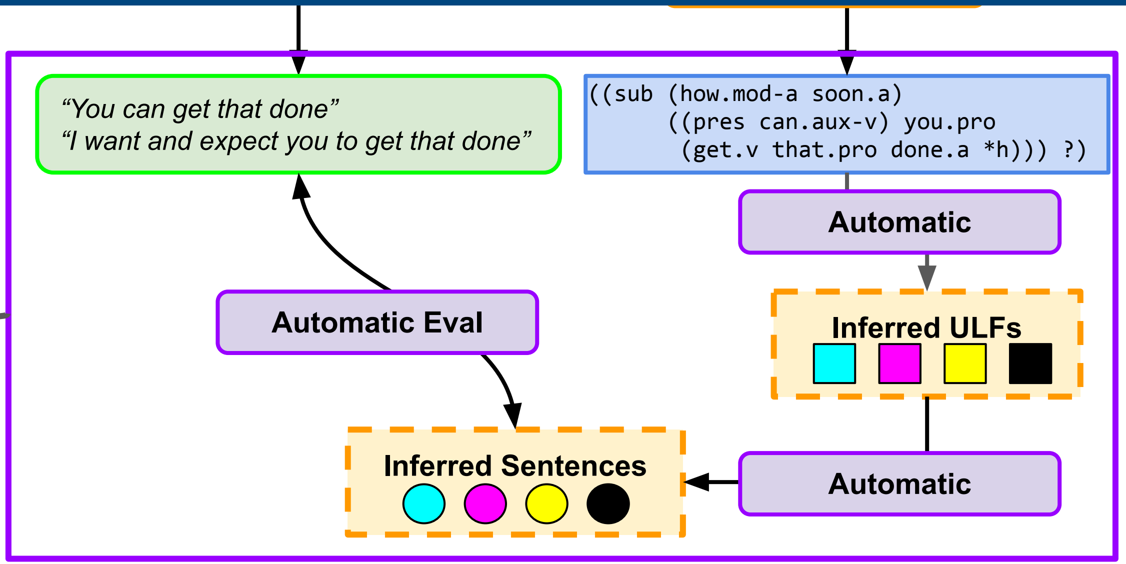


"You can get that done"
"I want and expect you to get that done"

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

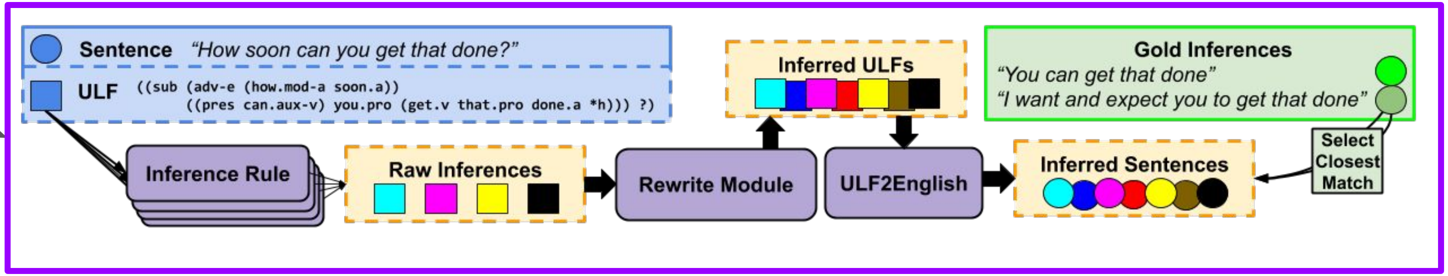
Automatic Evaluation (Recall)

The automatic evaluation has extra steps to deal with paraphrases



“You can get that done”
“I want and expect you to get that done”

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

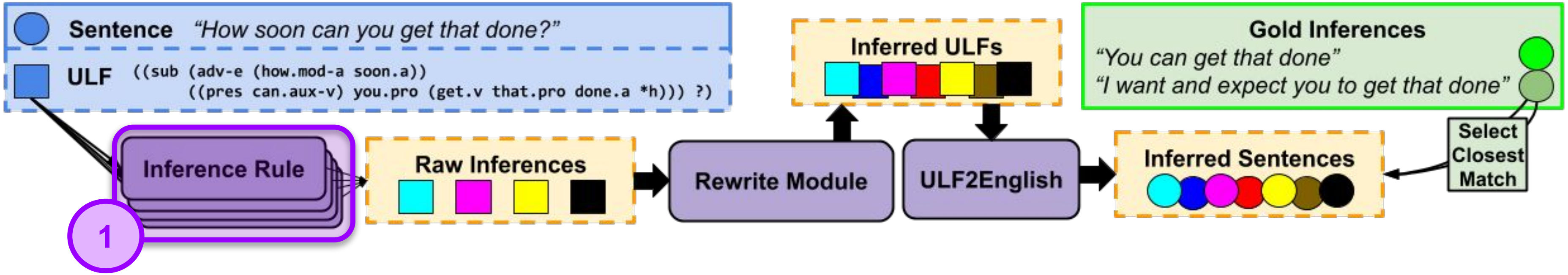


“How soon can you get that done?”

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

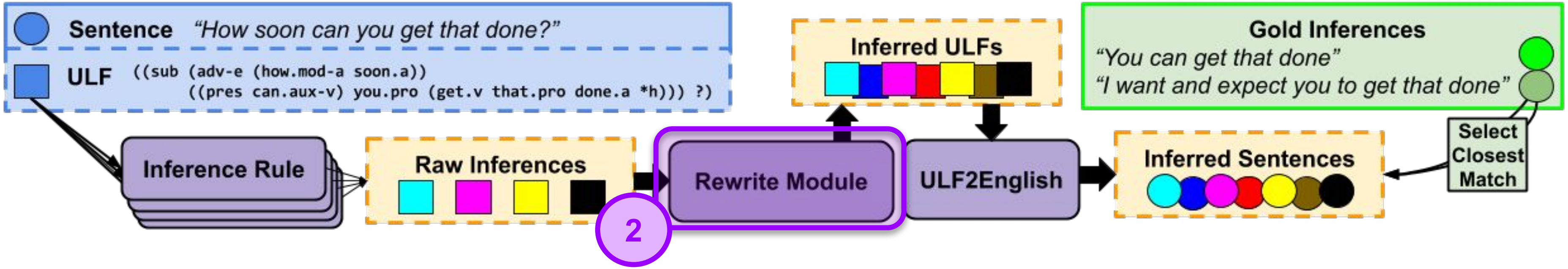
“You can get that done”
“I want and expect you to get that done”

Automatic Evaluation (Recall)


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
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

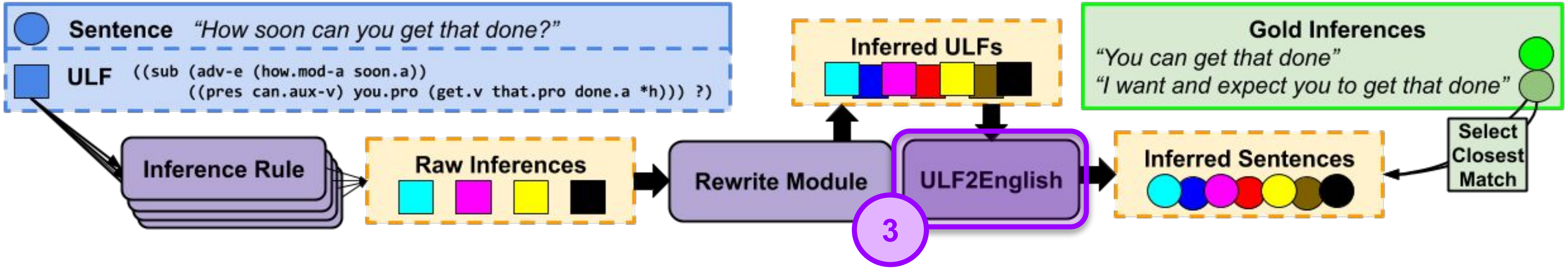
1. **Generate ULF inferences using generalized ULF predicates and transformations:**
   `infer-wh-question-presupposition`

Automatic Evaluation (Recall)


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
2
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

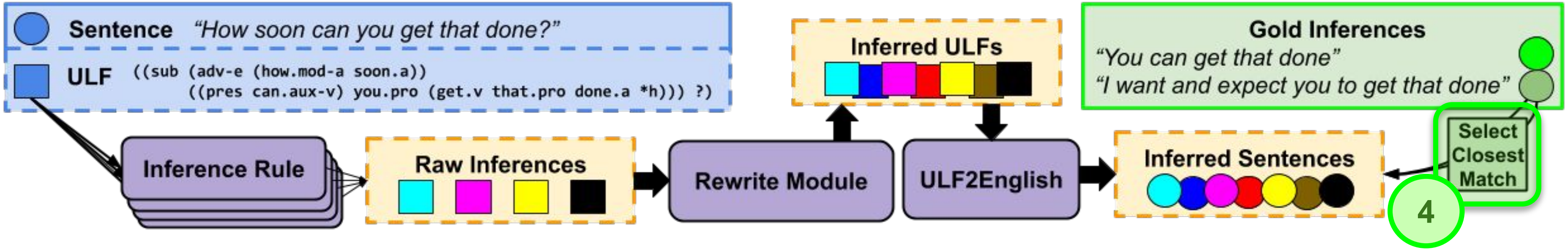
1. **Generate ULF inferences using generalized ULF predicates and transformations:**
   `infer-wh-question-presupposition`
2. **Rewrite inferences into possible other forms [In ULF]:**
   “I want you to get that done” + “I expect you to get that done” → “I want and expect you to get that done”

Automatic Evaluation (Recall)


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
3
Inferred Sentences
Gold Inferences
"You can get that done"
"I want and expect you to get that done"
Select Closest Match

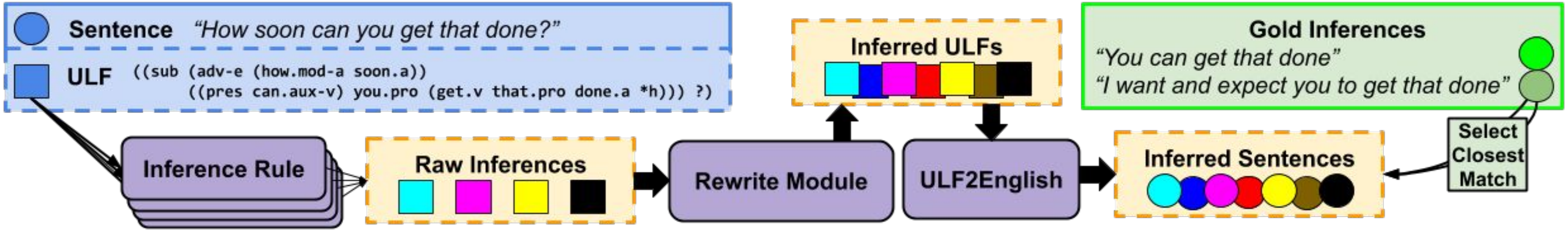
1. **Generate ULF inferences using generalized ULF predicates and transformations:**
   `infer-wh-question-presupposition`
2. **Rewrite inferences into possible other forms [In ULF]:**
   "I want you to get that done" + "I expect you to get that done" → *"I want and expect you to get that done"*
3. **Translate 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"*

Automatic Evaluation (Recall)


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
4

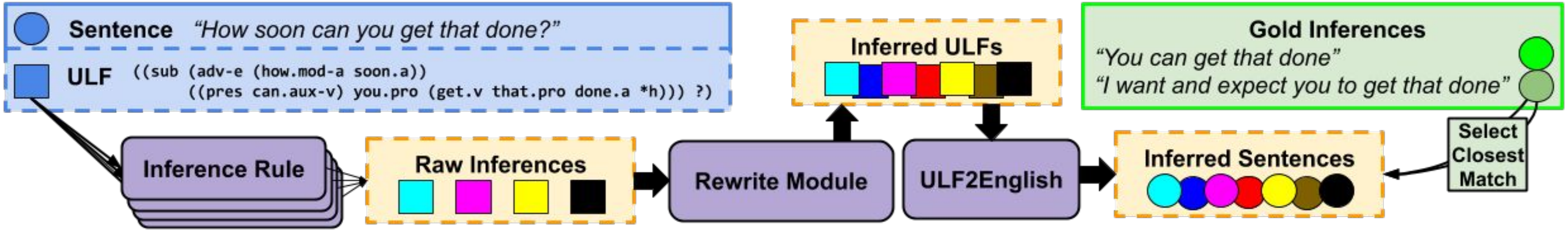
1. **Generate ULF inferences using generalized ULF predicates and transformations:**
   `infer-wh-question-presupposition`
2. **Rewrite inferences into possible other forms [In ULF]:**
   "I want you to get that done" + "I expect you to get that done" → *"I want and expect you to get that done"*
3. **Translate 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 between the gold inferences and the available rewrite sentences**
   a. Allow 3 character edit distance between gold inference and inferred sentence to allow minor English generation errors.

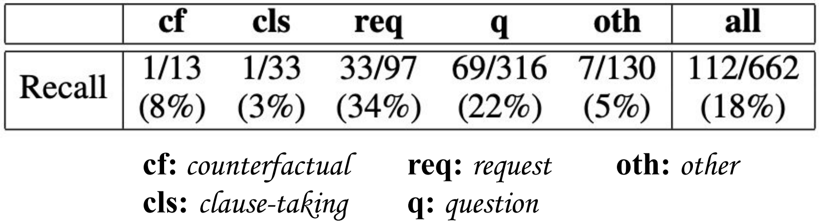
Automatic Evaluation (Recall)


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

1. **Generate ULF inferences using generalized ULF predicates and transformations:**
   `infer-wh-question-presupposition`
2. **Rewrite inferences into possible other forms [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 between the gold inferences and the available rewrite sentences**
   a. Allow 3 character edit distance between gold inference and inferred sentence to allow minor English generation errors.

Automatic Evaluation (Recall)

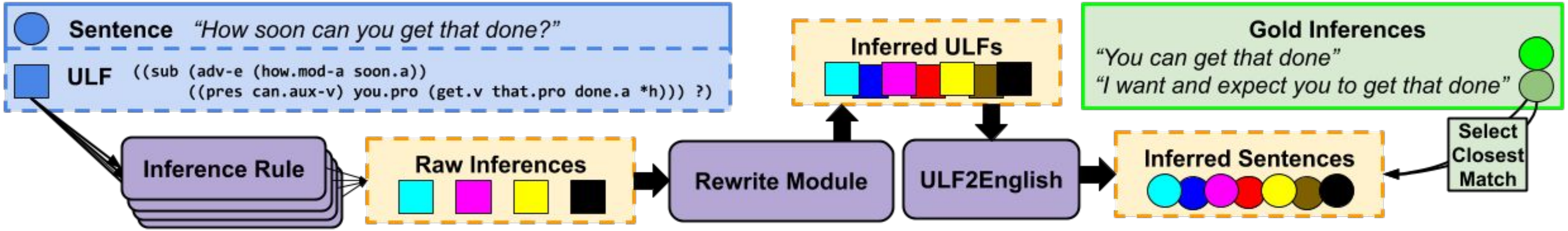

Sentence "How soon can you get that done?"
ULF ((sub (adv-e (how.mod-a soon.a))
((pres can.aux-v) you.pro (get.v that.pro done.a *h))) ?)
Inference Rule
Raw Inferences
Rewrite Module
Inferred ULFs
ULF2English
Inferred Sentences
Gold Inferences
"You can get that done"
"I want and expect you to get that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



	cf	cls	req	q	oth	all
Recall	1/13 (8%)	1/33 (3%)	33/97 (34%)	69/316 (22%)	7/130 (5%)	112/662 (18%)

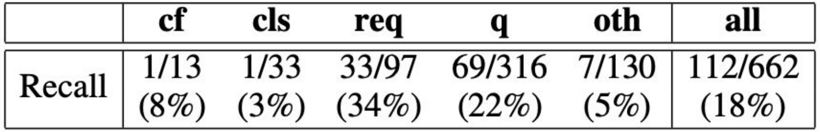
cf: *counterfactual* **req:** *request* **oth:** *other*
cls: *clause-taking* **q:** *question*

Automatic Evaluation (Recall)


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

Results are low…

but consider simple baseline’s performance



	cf	cls	req	q	oth	all
Recall	1/13 (8%)	1/33 (3%)	33/97 (34%)	69/316 (22%)	7/130 (5%)	112/662 (18%)

cf: *counterfactual* **req:** *request* **oth:** *other*
cls: *clause-taking* **q:** *question*

Takeaways

Takeaways

- The form close to syntax allowed evaluation over English using reliable generation.

Takeaways

- The form close to syntax allowed evaluation over English using reliable generation.

- The underlying semantic coherence allows the construction of inference rules, though with an additional interface to handle the syntax.

Conclusions

- We presented the first known method of generating inferences from ULF.

Conclusions

- We presented the first known method of generating inferences from ULF.
- We presented a method of collecting human elicitations of restricted categories of structural inferences, allowing a novel forward inference evaluation.

Conclusions

- We presented the first known method of generating inferences from ULF.
- We presented a method of collecting human elicitations of restricted categories of structural inferences, allowing a novel forward inference evaluation.
- 68.5% of the inferences were judged as reasonable by human judges.

Conclusions

- We presented the first known method of generating inferences from ULF.
- We presented a method of collecting human elicitations of restricted categories of structural inferences, allowing a novel forward inference evaluation.
- 68.5% of the inferences were judged as reasonable by human judges.
- Our experiments demonstrate some of the advantages of using a semantic representation closer to the syntactic form such as ULF—reliable translation to English and access to syntactic signals—though this comes at the cost of a more complicated interface with the semantic patterns.

Conclusions

- We presented the first known method of generating inferences from ULF.
- We presented a method of collecting human elicitations of restricted categories of structural inferences, allowing a novel forward inference evaluation.
- 68.5% of the inferences were judged as reasonable by human judges.
- Our experiments demonstrate some of the advantages of using a semantic representation closer to the syntactic form such as ULF—reliable translation to English and access to syntactic signals—though this comes at the cost of a more complicated interface with the semantic patterns.
- Improvements in the human elicitation procedure and implementation of the inference system (e.g. clause-taking verbs) are clear areas of future work. A larger and more refined dataset of inference elicitations will likely allow the development of a robust inference system.

Acknowledgements

We would like to thank the paper reviewers for their thoughtful feedback. This work was supported by DARPA CwC subcontract W911NF-15-1-0542.

Analysis & Discussion

Poor performance on counterfactual and clause-taking categories due to few examples

“he said he would give a ruble to anyone who found a hare” → ***“A hare”***

Needs improved sampling and larger dataset

Analysis & Discussion

Poor performance on counterfactual and clause-taking categories due to few examples

“he said he would give a ruble to anyone who found a hare” → ***“A hare”***

Needs improved sampling and larger dataset

Annotator disagreements on usage of certainty words

probably, likely, [absence of any], etc.

Move this to a separate likelihood metric of inferences

Analysis & Discussion

Disagreements on the boundary of request and questions

“Could you open the door?” ?→? “You know whether you could open the door”

Analysis & Discussion

Disagreements on the boundary of request and questions

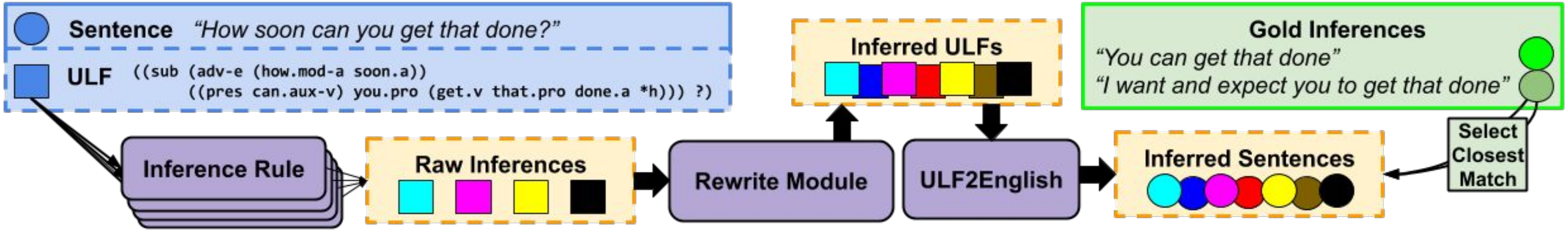
"Could you open the door?" ?→? "You know whether you could open the door"

Some remaining ULF to English errors

Me have a wife (subject/object pronouns)

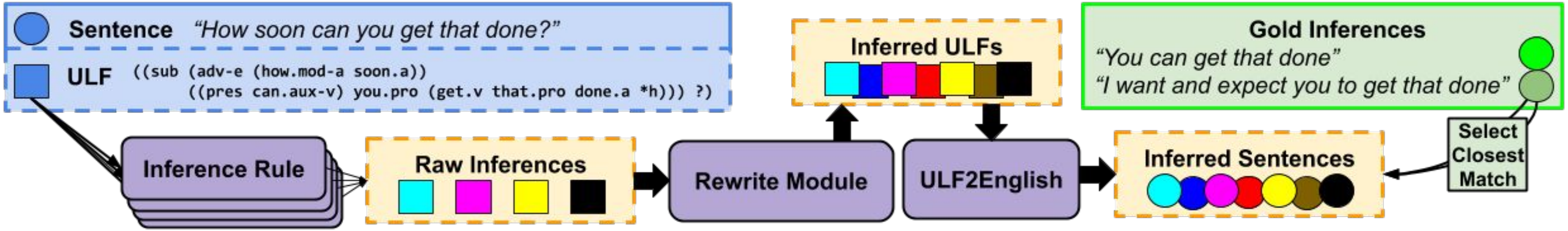
It will entail a radical departure from current policys. (certain pluralizations)

Automatic Evaluation (Recall)


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

Evaluation of rewrite module:

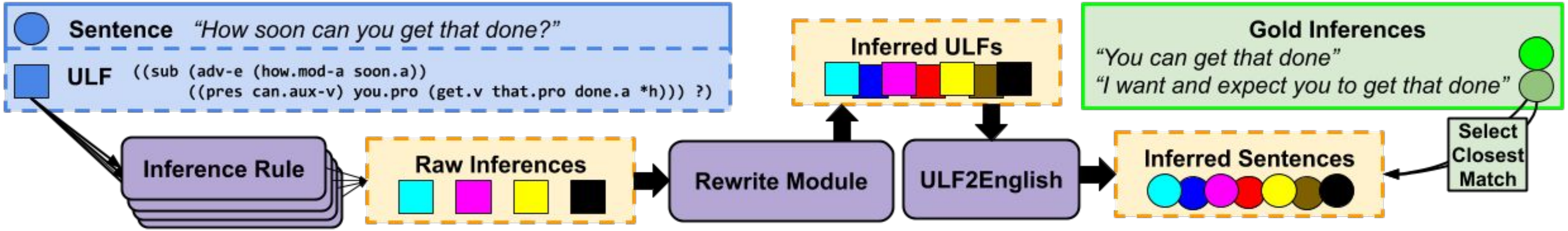
Automatic Evaluation (Recall)


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

<u>Evaluation of rewrite module:</u>

- Sampled 100 sentences from final inferred sentences that were closest to gold inferences

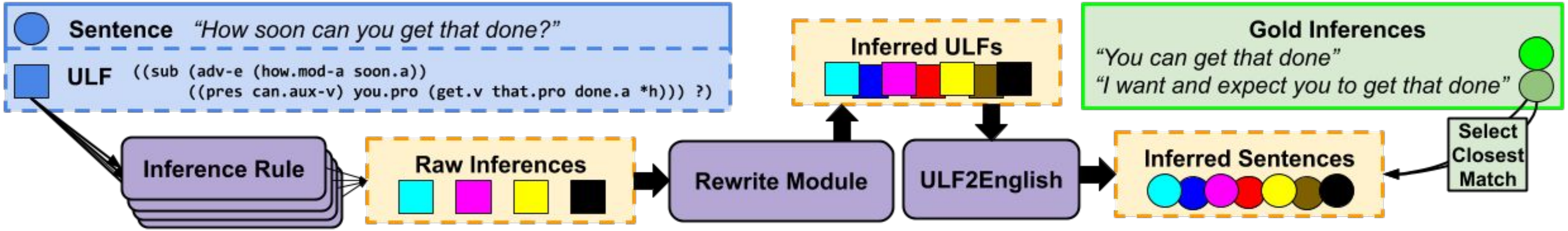
Automatic Evaluation (Recall)


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

<u>Evaluation of rewrite module:</u>

- Sampled 100 sentences from final inferred sentences that were closest to gold inferences
- Evaluated whether the sentence is a *valid rewriting* of the original inference(s)

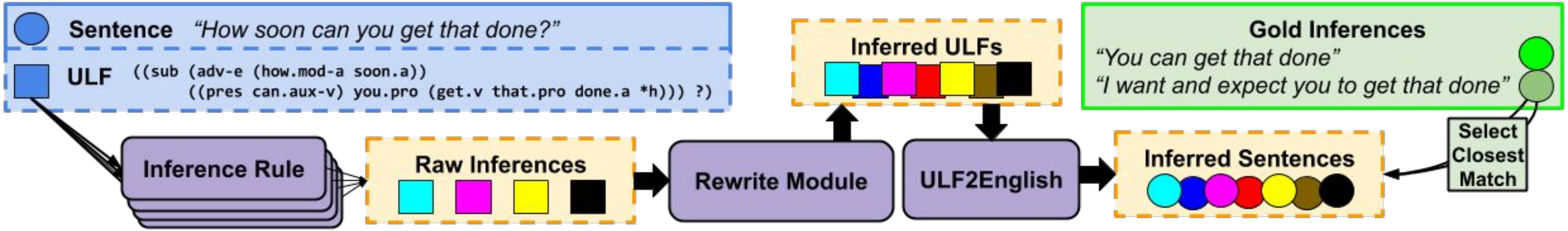
Automatic Evaluation (Recall)


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

<u>Evaluation of rewrite module:</u>

- Sampled 100 sentences from final inferred sentences that were closest to gold inferences
- Evaluated whether the sentence is a *valid rewriting* of the original inference(s)
- A *valid rewriting* does not introduce any new semantic information

Automatic Evaluation (Recall)


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

<u>Evaluation of rewrite module:</u>

- Sampled 100 sentences from final inferred sentences that were closest to gold inferences
- Evaluated whether the sentence is a *valid rewriting* of the original inference(s)
- A *valid rewriting* does not introduce any new semantic information

91/100 sentences were valid rewritings

Inference Evaluation

1. **Gather multi-genre dataset**
   a. **Sentences filtered by superficial patterns to reduce annotation overhead**

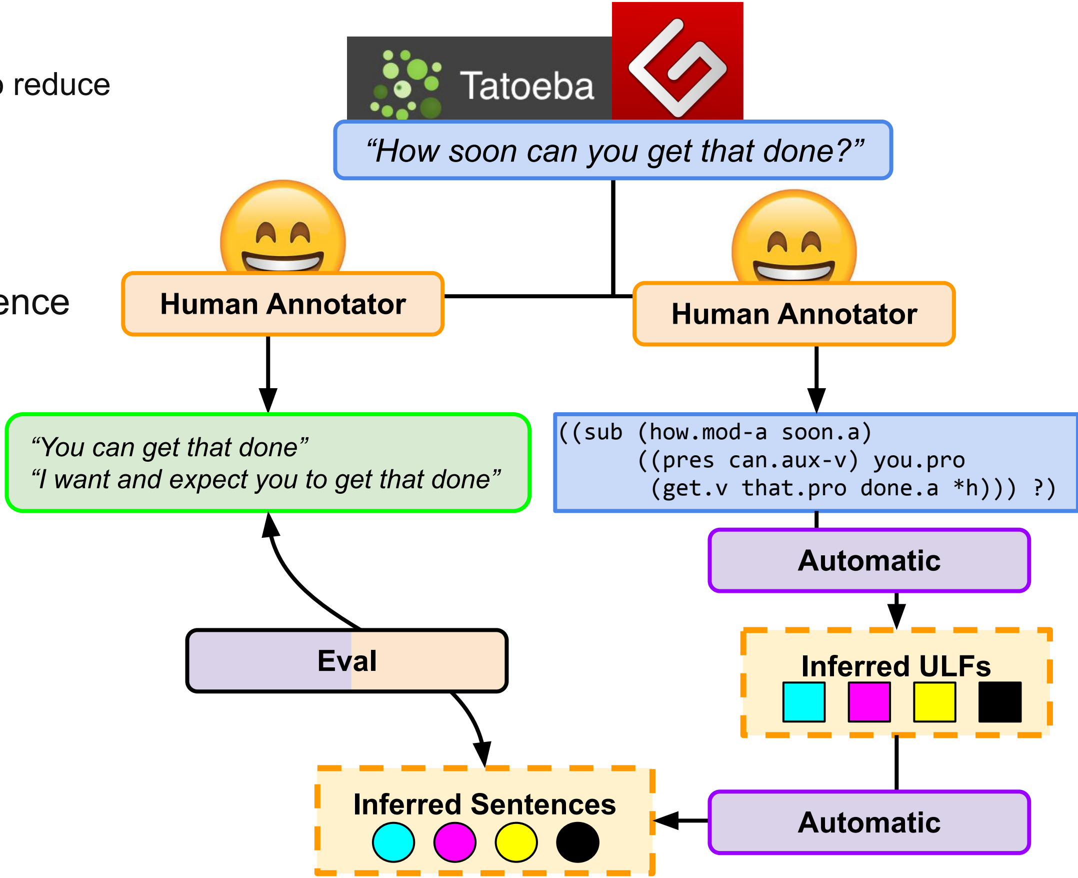
Tatoeba

“How soon can you get that done?”

- Filtering patterns
  - *(if-then)* “if something <past tense/participle> something <future marking word> something”
  - *(inverted if-then)* “something <future marking word> something if something <past tense/participle>”

Inference Evaluation

1. Gather multi-genre dataset
    a. Sentences filtered by superficial patterns to reduce annotation overhead
2. Elicit human inferences in each category
    a. Reduced noise with structured guidance
3. Collect gold ULF annotations of each sentence
4. Use inference rules to make conclusions
5. Automatically generate English from ULF inferences
6. Evaluate
    a. precision using human judgments
    b. recall using automatic matching



Tatoeba

"How soon can you get that done?"

Human Annotator

Human Annotator

"You can get that done"
"I want and expect you to get that done"

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

Automatic

Inferred ULFs
🟦 🟪 🟨 ⬛

Eval

Automatic

Inferred Sentences
🔵 🟣 🟡 ⚫

ULF? (Episodic Logic)

Episodic Logic

- Extended FOL
- Closely matches expressivity of natural languages
  - Predicates, connectives, quantifiers, equality → FOL
  - Predicate and sentence modification ***(e.g. very, gracefully, nearly, possibly)***
  - Predicate and sentence reification ***(e.g. <u>Beauty</u> is subjective, <u>That exoplanets exist</u> is now certain)***
  - Generalized quantifiers ***(e.g. most men who smoke)***
  - Intensional predicates ***(e.g. believe, intend, resemble)***
  - Reference to events and situations ***(Many children had not been vaccinated against measles; <u>this situation</u> caused sporadic outbreaks of the disease)***
- Suitable for deductive, uncertain, and Natural-Logic-like inference
- A fast and comprehensive theorem prover, EPILOG, is already available.

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”, “Could you dial for me?”

ULFs

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

(((pres could.aux-v) you.pro
  (dial.v {ref1}.pro (adv-a (for.p me.pro)))) ?)
```

Entity($\mathcal{D}$): |Alice|, |John|, you.pro, {ref1}.pro, me.pro

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): think.v, fall.v, dial.v, for.p

Basic Ontological Types

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

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

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”, “Could you dial for me?”

ULFs

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

(((pres could.aux-v) you.pro
  (dial.v {ref1}.pro (adv-a (for.p me.pro)))) ?)
```

Entity($\mathcal{D}$): `|Alice|`, `|John|`, `you.pro`, `{ref1}.pro`, `me.pro`

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): `think.v`, `fall.v`, `dial.v`, `for.p`

Predicate modifier($\mathcal{N} \rightarrow \mathcal{N}$): `nearly.adv-a`, `(adv-a (for.p me.pro))`

Basic Ontological Types

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

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

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”, “Could you dial for me?”

ULFs

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

(((pres could.aux-v) you.pro
  (dial.v {ref1}.pro (adv-a (for.p me.pro)))) ?)
```

Entity($\mathcal{D}$): |Alice|, |John|, you.pro, {ref1}.pro, me.pro

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): think.v, fall.v, dial.v, for.p

Predicate modifier($\mathcal{N} \rightarrow \mathcal{N}$): nearly.adv-a, (adv-a (for.p me.pro))

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

Basic Ontological Types

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

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

ULF? (semantics)

A minimal step across from syntax to semantics in Episodic Logic

“Alice thinks that John nearly fell”, “Could you dial for me?”

ULFs

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

(((pres could.aux-v) you.pro
  (dial.v {ref1}.pro (adv-a (for.p me.pro)))) ?)
```

Entity($\mathcal{D}$): |Alice|, |John|, you.pro, {ref1}.pro, me.pro

n-ary predicate($\mathcal{D}^n \rightarrow (\mathcal{S} \rightarrow \mathbf{2})$): think.v, fall.v, dial.v, for.p

Predicate modifier($\mathcal{N} \rightarrow \mathcal{N}$): nearly.adv-a, (adv-a (for.p me.pro))

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

Basic Ontological Types

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

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

Also... *determiner, sentence modifier, connective, lambda abstract, predicate reifier*