Learning in the Rational Speech Acts Model

Christopher Potts

Stanford Linguistics

Paper: http://arxiv.org/abs/1510.06807

Will Monroe

Bringing machine learning and pragmatics together

Bringing machine learning and pragmatics together

Bayesian pragmatic models

Bringing machine learning and pragmatics together

Bayesian pragmatic models

- Analytic insights
- Small problems
- Idealized agents
- Constrained by model design

Bringing machine learning and pragmatics together

Bayesian pragmatic models

- Analytic insights
- Small problems
- Idealized agents
- Constrained by model design

Machine learning

Bringing machine learning and pragmatics together

Bayesian pragmatic models

- Analytic insights
- Small problems
- Idealized agents
- Constrained by model design

Machine learning

- Coverage
- Huge, unruly problems
- Fallible agents
- Constrained by experiences in training

Bringing machine learning and pragmatics together

Bayesian pragmatic models

- Analytic insights
- Small problems
- Idealized agents
- Constrained by model design

Machine learning

- Coverage
- Huge, unruly problems
- Fallible agents
- Constrained by experiences in training

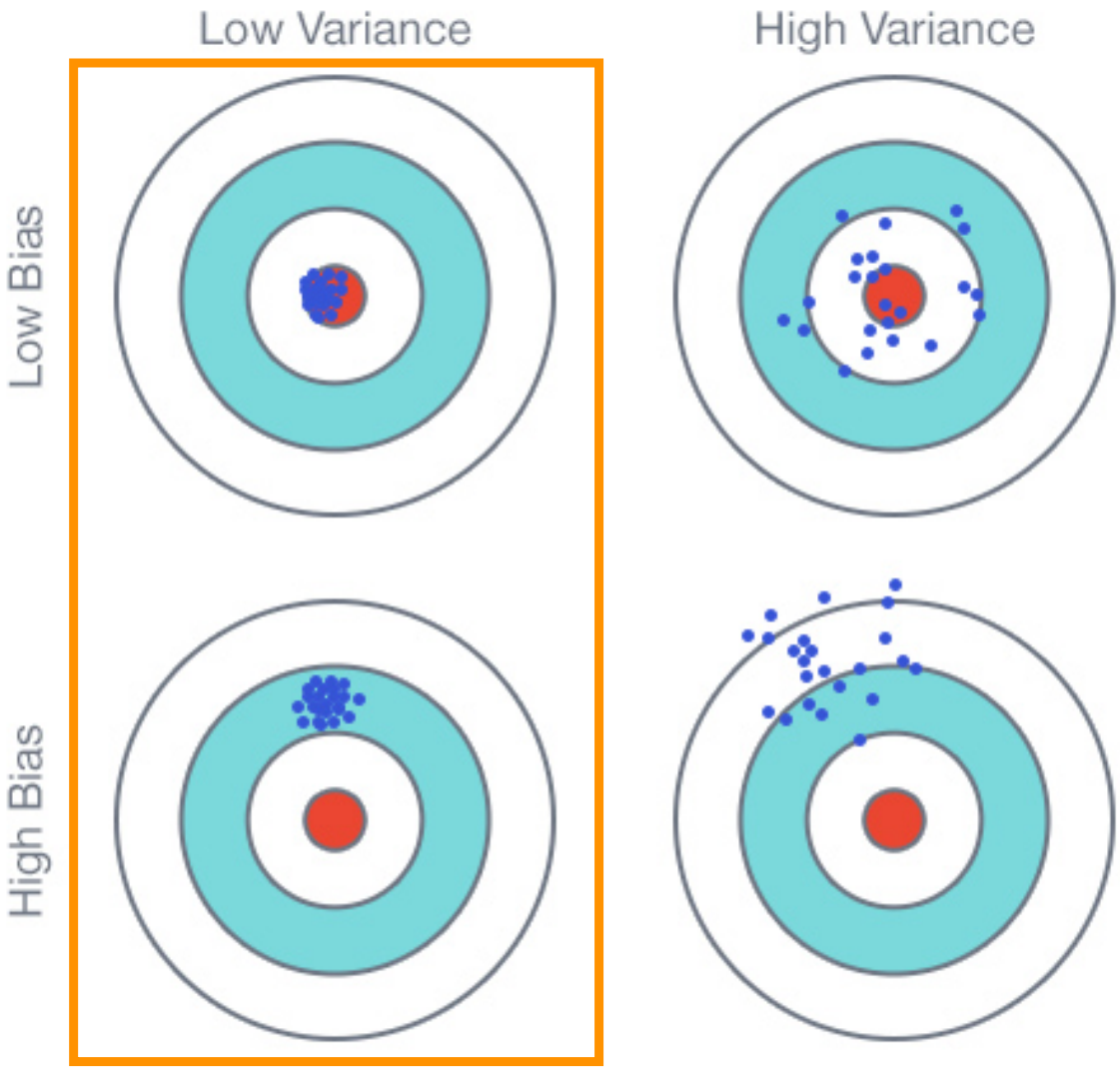
Goal: combine the best aspects of both to achieve broader coverage and novel pragmatic insights

Plan

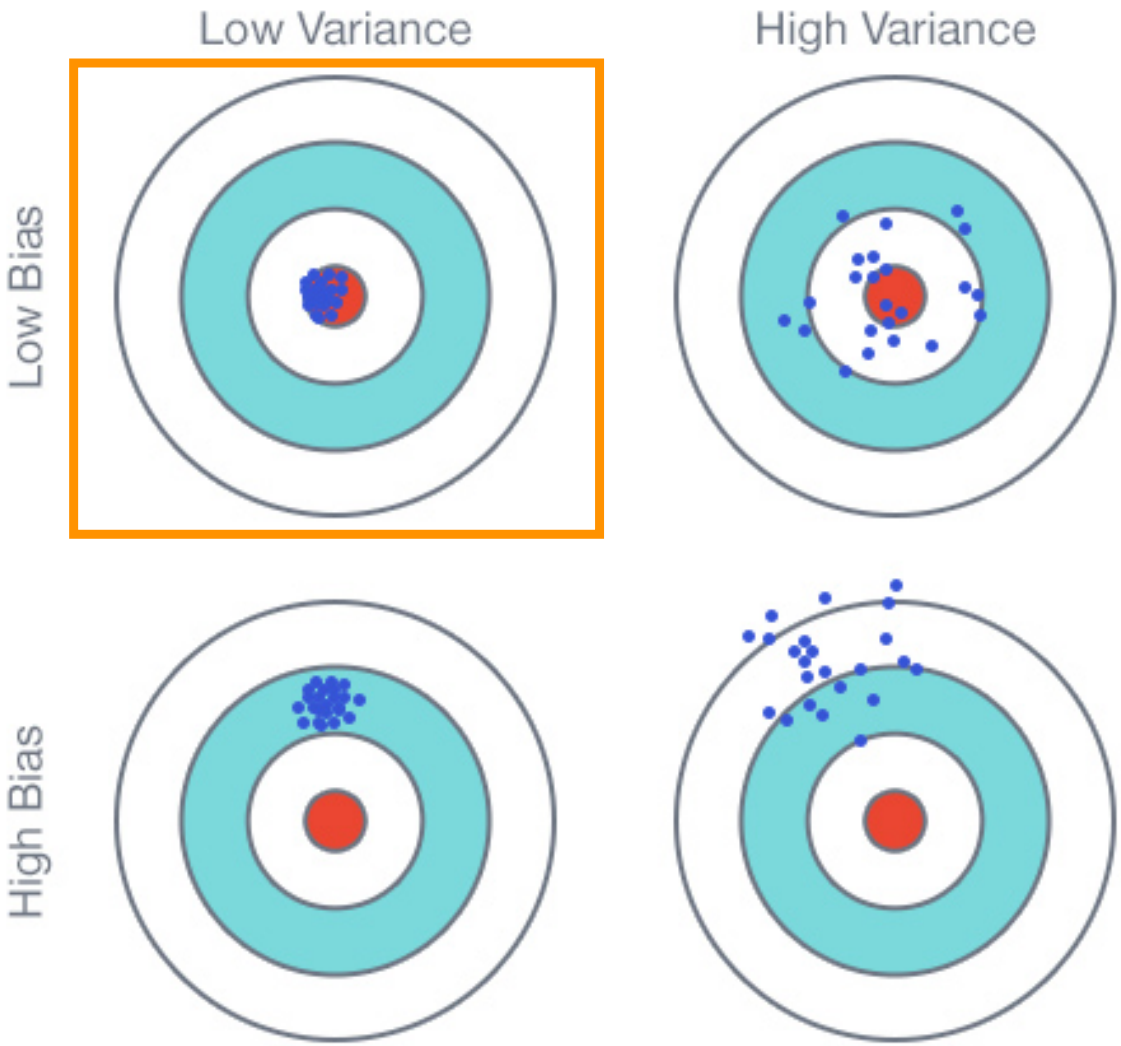
1. The Rational Speech Acts (RSA) model
2. TUNA
3. Learned RSA
4. Experiments

Bias and variance

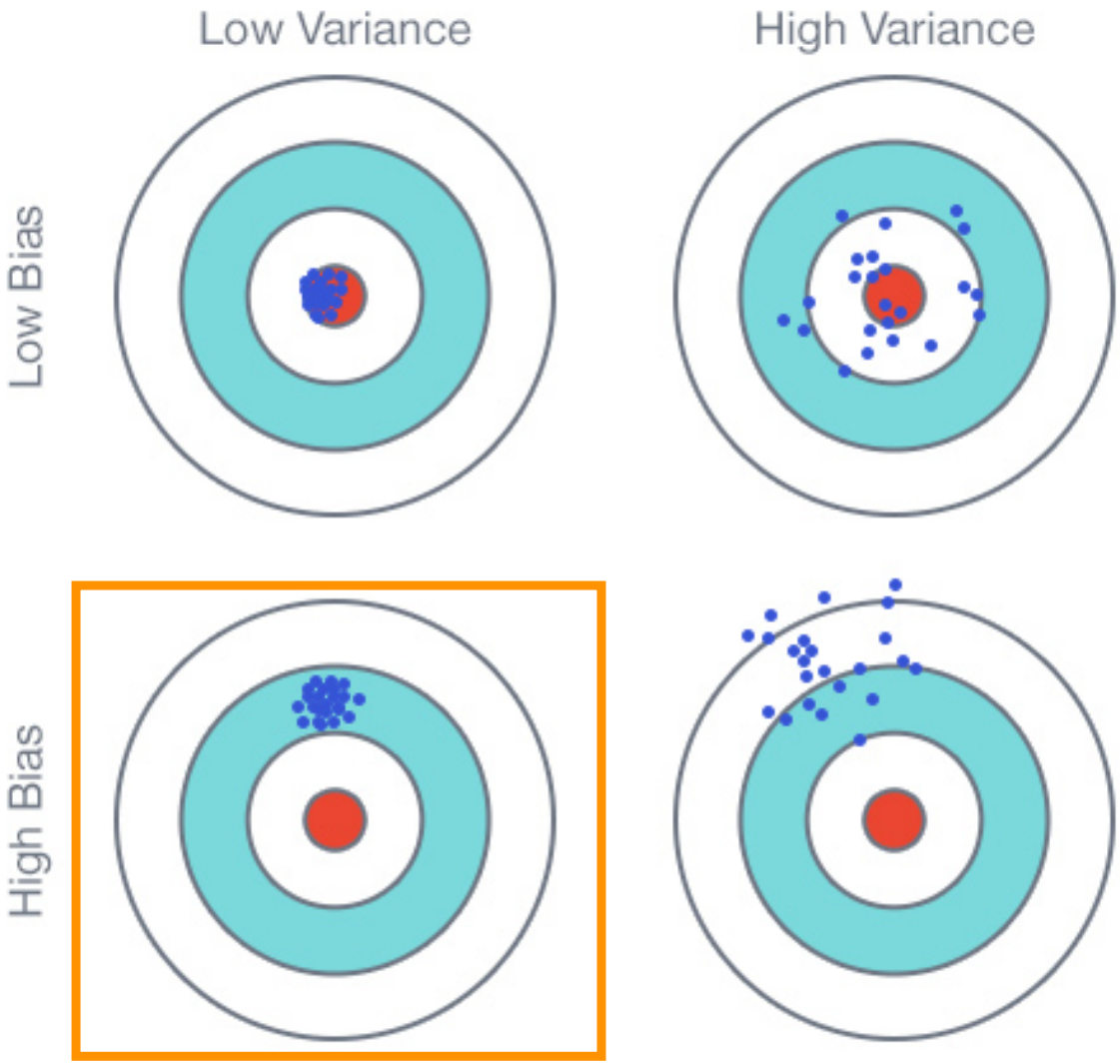
Bias and variance



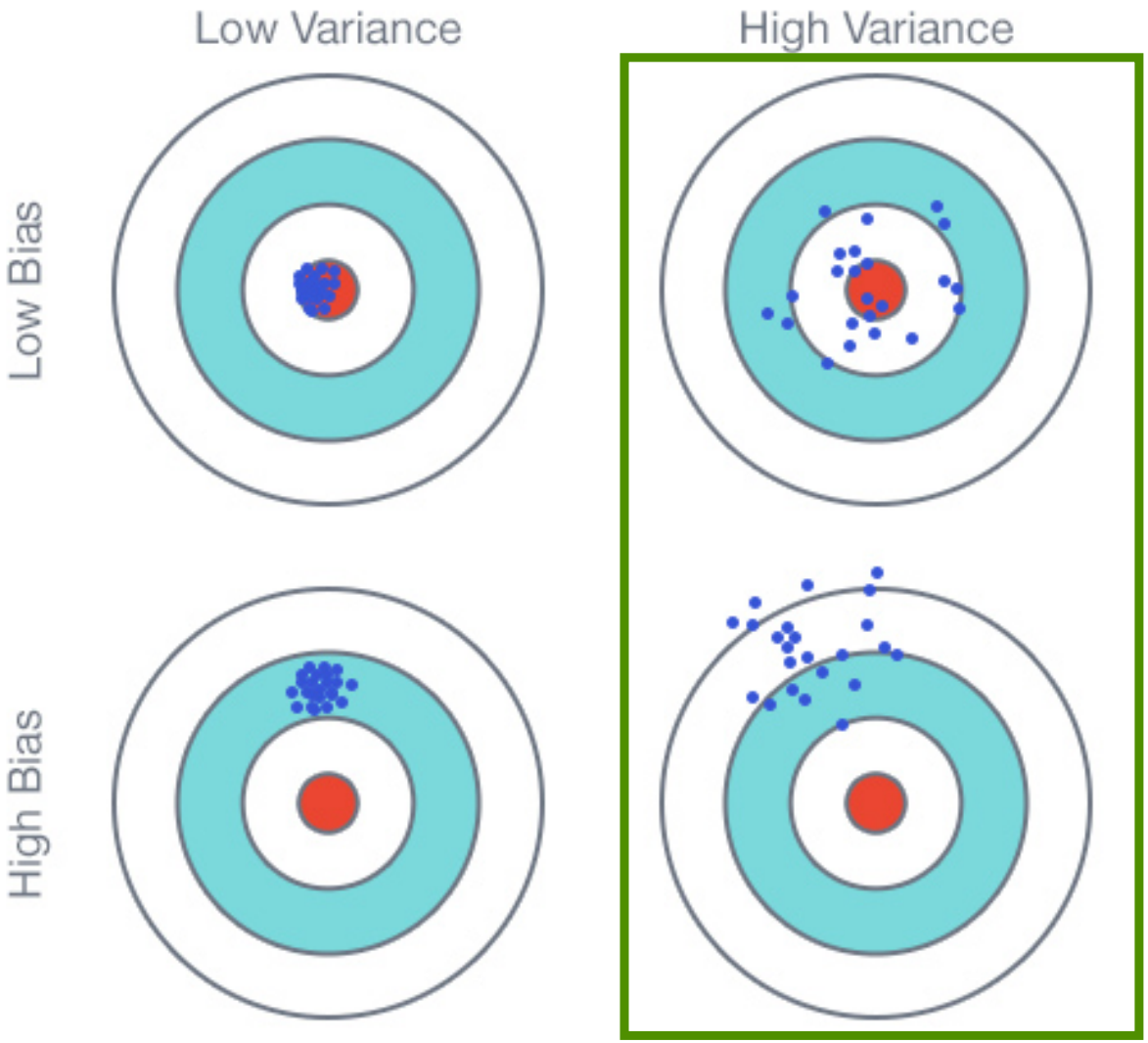
Bias and variance



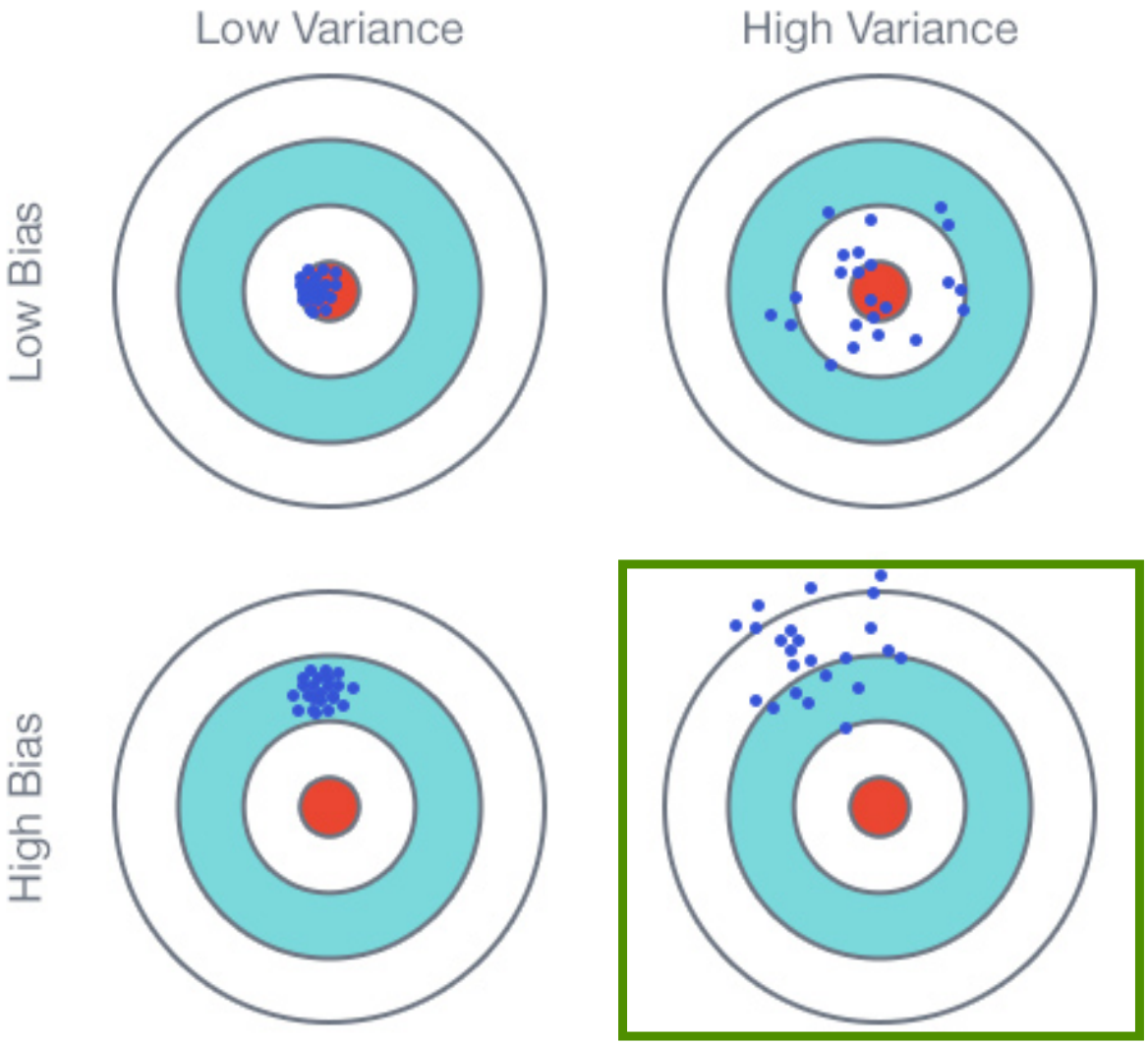
Bias and variance



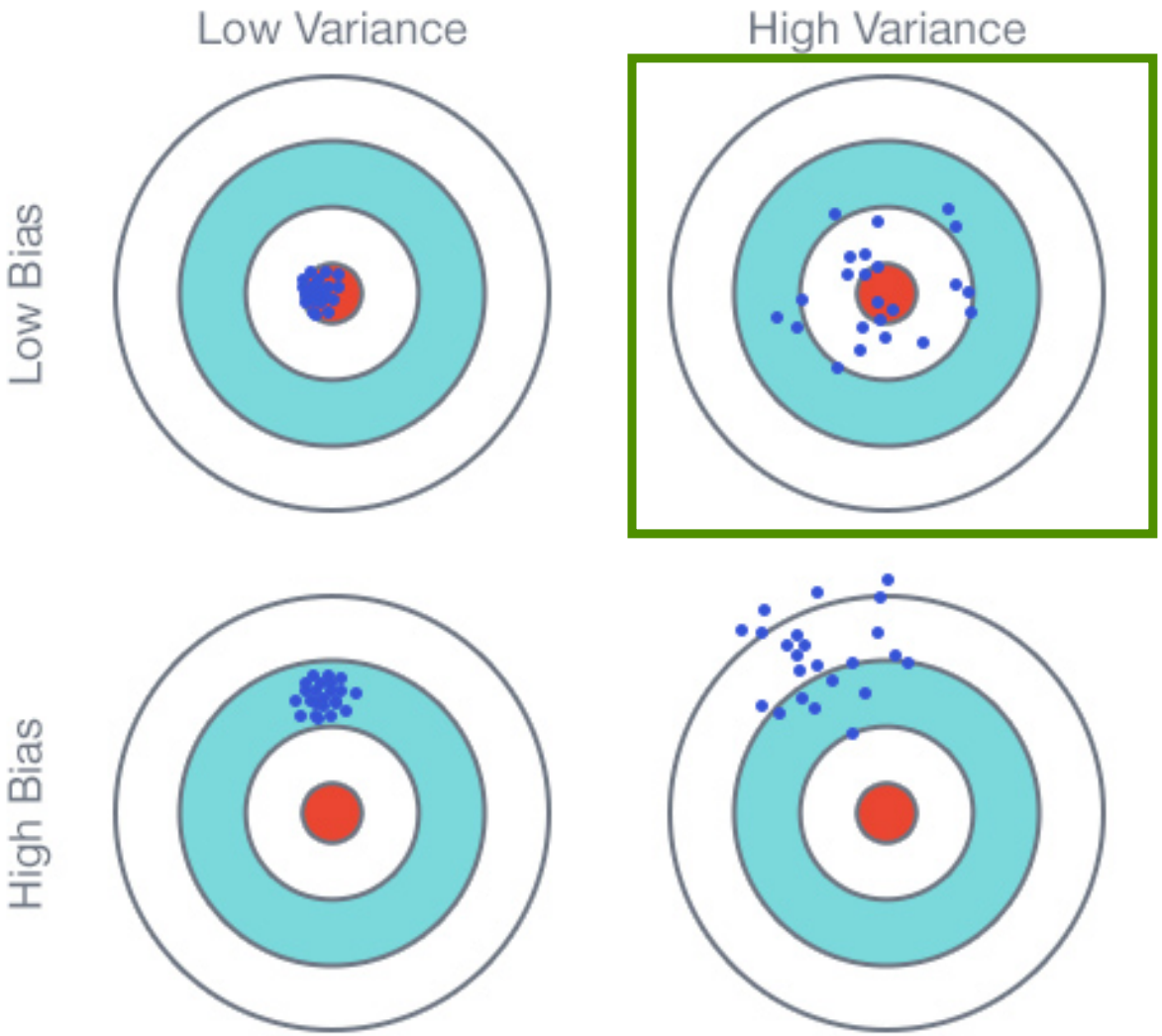
Bias and variance



Bias and variance



Bias and variance



Semantic parsing

Semantic parsing

Chat80:

```
/* Sentences */
sentence(S) --> declarative(S), terminator(.) .
sentence(S) --> wh_question(S), terminator(?) .
sentence(S) --> yn_question(S), terminator(?) .
sentence(S) --> imperative(S), terminator(!) .

/* Noun Phrase */
np(np(Agmt,Pronoun,[]),Agmt,NPCase,def,_,Set,Nil) -->
  {is_pp(Set)},
  pers_pron(Pronoun,Agmt,Case),
  {empty(Nil), role(Case,decl,NPCase)}.

/* Prepositional Phrase */
pp(pp(Prep,Arg),Case,Set,Mask) -->
  prep(Prep),
  {prep_case(NPCase)},
  np(Arg,_,NPCase,_,Case,Set,Mask).
```

Semantic parsing

```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

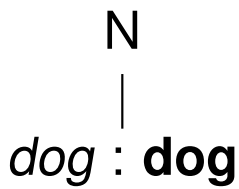
Semantic parsing

1 **for** $w \in$ *Words*
2 **for** $X \in$ *Categories*
3 **for** $d \in$ *Domain*
4 **yield** '$X \rightarrow w : d$'

0 N $\rightarrow$ *dog* : **dog**
0 V $\rightarrow$ *dog* : $\mathbf{dog}_v$
0 N $\rightarrow$ *dog* : **cat**
0 N $\rightarrow$ *cat* : **cat**
0 N $\rightarrow$ *cat* : **dog**
0 V $\rightarrow$ *jump* : **dog**
0 V $\rightarrow$ *jump* : **jump**

Semantic parsing

```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

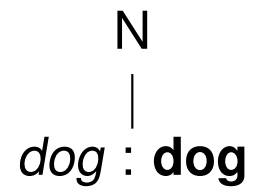


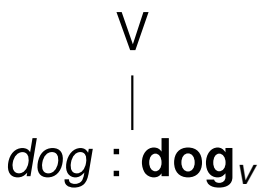
1	N	→	*dog*	:	**dog**
0	V	→	*dog*	:	$\mathbf{dog}_v$
0	N	→	*dog*	:	**cat**
0	N	→	*cat*	:	**cat**
0	N	→	*cat*	:	**dog**
0	V	→	*jump*	:	**dog**
0	V	→	*jump*	:	**jump**

Semantic parsing

```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

1	N	→	*dog*	:	**dog**
1	V	→	*dog*	:	**dog**$_v$
0	N	→	*dog*	:	**cat**
0	N	→	*cat*	:	**cat**
0	N	→	*cat*	:	**dog**
0	V	→	*jump*	:	**dog**
0	V	→	*jump*	:	**jump**

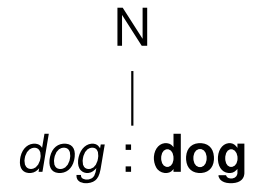


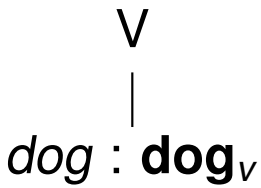


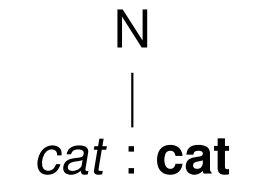
Semantic parsing

1 **for** $w \in$ *Words*
2 **for** $X \in$ *Categories*
3 **for** $d \in$ *Domain*
4 **yield** '$X \rightarrow w : d$'

1	N	$\rightarrow$	*dog*	:	**dog**
1	V	$\rightarrow$	*dog*	:	$\mathbf{dog}_v$
0	N	$\rightarrow$	*dog*	:	**cat**
1	N	$\rightarrow$	*cat*	:	**cat**
0	N	$\rightarrow$	*cat*	:	**dog**
0	V	$\rightarrow$	*jump*	:	**dog**
0	V	$\rightarrow$	*jump*	:	**jump**



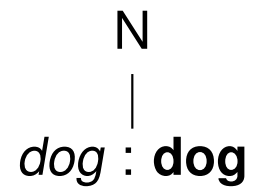


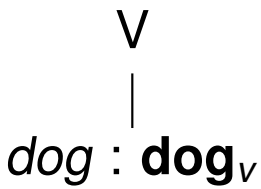


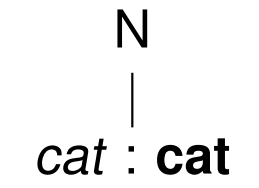
Semantic parsing

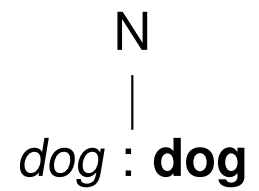
```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

2	N	→	*dog*	:	**dog**
1	V	→	*dog*	:	$\mathbf{dog}_v$
0	N	→	*dog*	:	**cat**
1	N	→	*cat*	:	**cat**
0	N	→	*cat*	:	**dog**
0	V	→	*jump*	:	**dog**
0	V	→	*jump*	:	**jump**





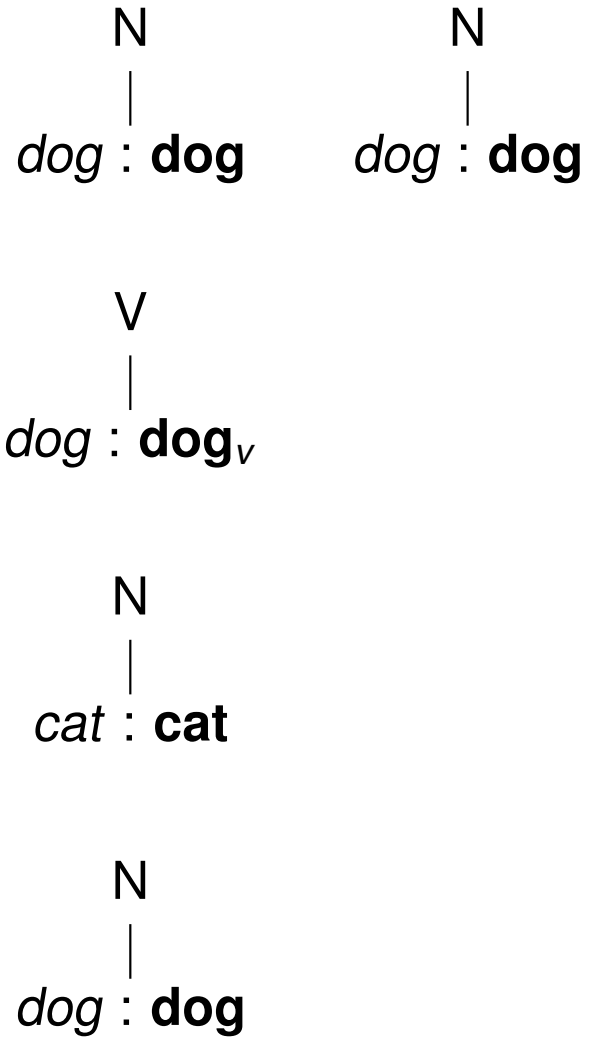




Semantic parsing

```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

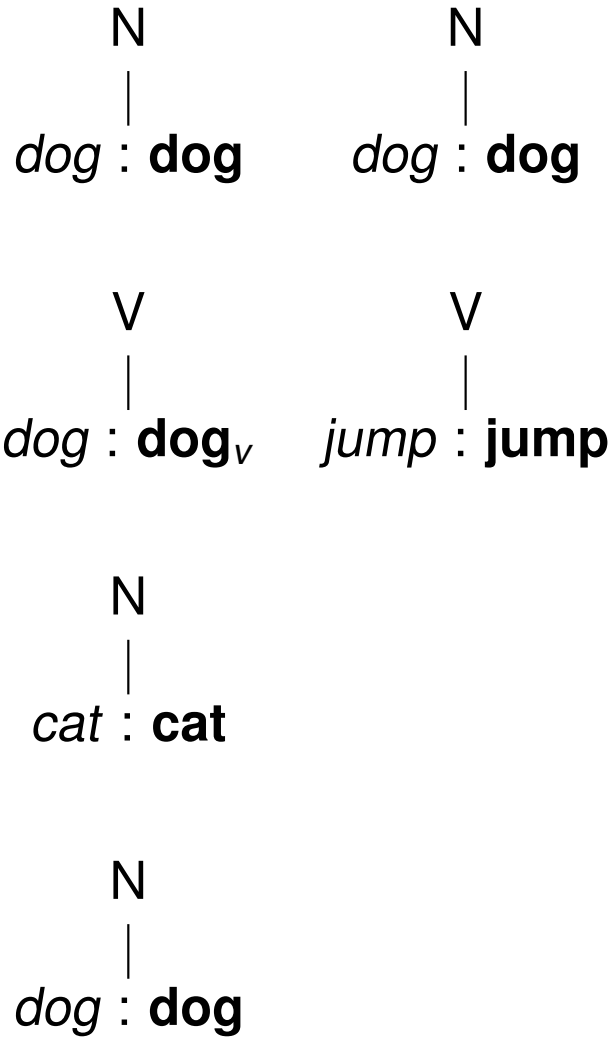
3	N	→	*dog*	:	**dog**
1	V	→	*dog*	:	$\mathbf{dog}_v$
0	N	→	*dog*	:	**cat**
1	N	→	*cat*	:	**cat**
0	N	→	*cat*	:	**dog**
0	V	→	*jump*	:	**dog**
0	V	→	*jump*	:	**jump**



Semantic parsing

```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

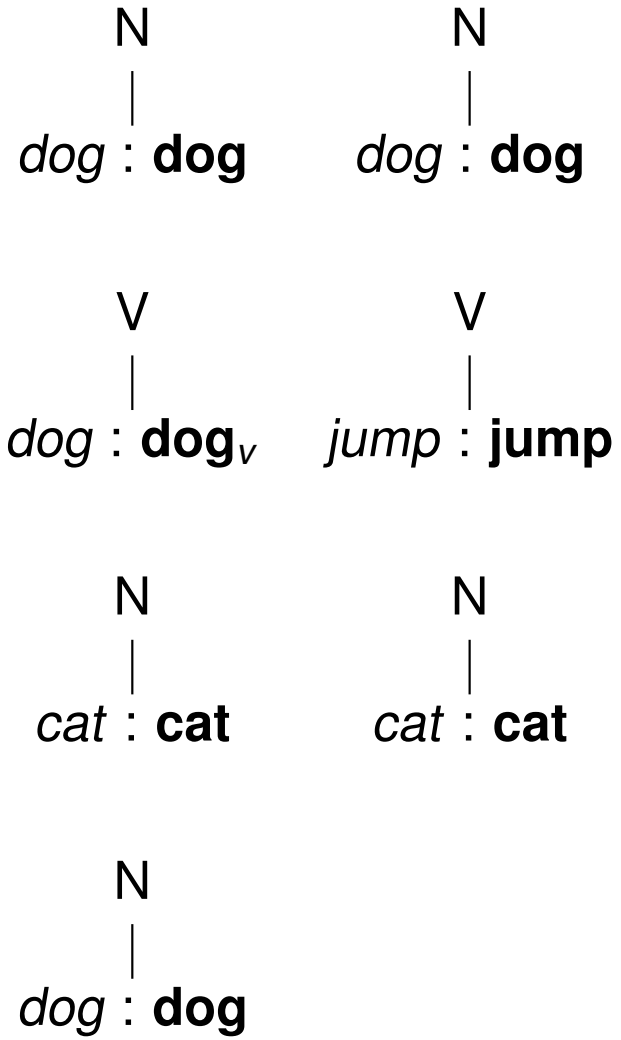
3	N	→	*dog*	:	**dog**
1	V	→	*dog*	:	$\mathbf{dog}_v$
0	N	→	*dog*	:	**cat**
1	N	→	*cat*	:	**cat**
0	N	→	*cat*	:	**dog**
0	V	→	*jump*	:	**dog**
1	V	→	*jump*	:	**jump**



Semantic parsing

```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

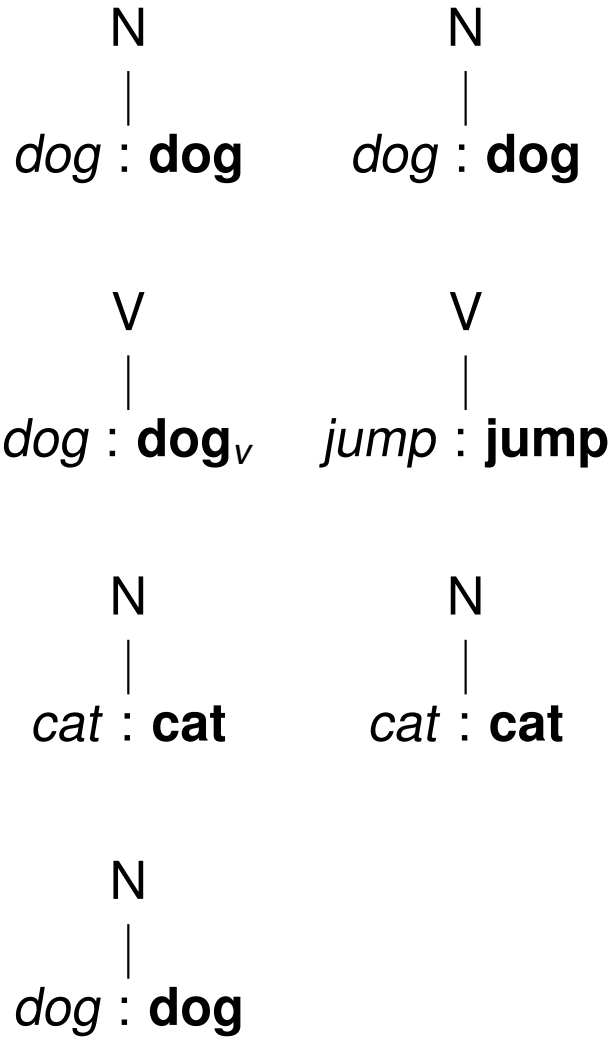
3	N	→	*dog*	:	**dog**
1	V	→	*dog*	:	$\mathbf{dog}_v$
0	N	→	*dog*	:	**cat**
2	N	→	*cat*	:	**cat**
0	N	→	*cat*	:	**dog**
0	V	→	*jump*	:	**jump**
1	V	→	*jump*	:	**jump**



Semantic parsing

```
1  for w ∈ Words
2      for X ∈ Categories
3          for d ∈ Domain
4              yield 'X → w : d'
```

3	N	→	*dog*	:	**dog**
1	V	→	*dog*	:	**dog**$_v$
0	N	→	*dog*	:	**cat**
2	N	→	*cat*	:	**cat**
0	N	→	*cat*	:	**dog**
0	V	→	*jump*	:	**dog**
1	V	→	*jump*	:	**jump**



Semantic parsing

for $w \in$ *Words*
  for $X \in$ *Categories*
    for $d \in$ *Domain*
      yield '$X \rightarrow w : d$'

```
/* Sentences */
sentence(S) --> declarative(S), terminator(.) .
sentence(S) --> wh_question(S), terminator(?) .
sentence(S) --> yn_question(S), terminator(?) .
sentence(S) --> imperative(S), terminator(!) .

/* Noun Phrase */
np(np(Agmt,Pronoun,[]),Agmt,NPCase,def,_,Set,Nil) -->
  {is_pp(Set)},
  pers_pron(Pronoun,Agmt,Case),
  {empty(Nil), role(Case,decl,NPCase)}.

/* Prepositional Phrase */
pp(pp(Prep,Arg),Case,Set,Mask) -->
  prep(Prep),
  {prep_case(NPCase)},
  np(Arg,_,NPCase,_,Case,Set,Mask).
```

Semantic parsing

Zettlemoyer & Collins 2005:

Rules		Categories produced from logical form
Input Trigger	Output Category	$\arg\max(\lambda x.state(x) \wedge borders(x, texas), \lambda x.size(x))$
constant c	$NP : c$	$NP : texas$
arity one predicate p_1	$N : \lambda x.p_1(x)$	$N : \lambda x.state(x)$
arity one predicate p_1	$S\backslash NP : \lambda x.p_1(x)$	$S\backslash NP : \lambda x.state(x)$
arity two predicate p_2	$(S\backslash NP)/NP : \lambda x.\lambda y.p_2(y, x)$	$(S\backslash NP)/NP : \lambda x.\lambda y.borders(y, x)$
arity two predicate p_2	$(S\backslash NP)/NP : \lambda x.\lambda y.p_2(x, y)$	$(S\backslash NP)/NP : \lambda x.\lambda y.borders(x, y)$
arity one predicate p_1	$N/N : \lambda g.\lambda x.p_1(x) \wedge g(x)$	$N/N : \lambda g.\lambda x.state(x) \wedge g(x)$
literal with arity two predicate p_2 and constant second argument c	$N/N : \lambda g.\lambda x.p_2(x, c) \wedge g(x)$	$N/N : \lambda g.\lambda x.borders(x, texas) \wedge g(x)$
arity two predicate p_2	$(N\backslash N)/NP : \lambda x.\lambda g.\lambda y.p_2(x, y) \wedge g(x)$	$(N\backslash N)/NP : \lambda g.\lambda x.\lambda y.borders(x, y) \wedge g(x)$
an $\arg\max$ / $\min$ with second argument arity one function f	$NP/N : \lambda g.\arg\max / \min(g, \lambda x.f(x))$	$NP/N : \lambda g.\arg\max(g, \lambda x.size(x))$
an arity one numeric-ranged function f	$S/NP : \lambda x.f(x)$	$S/NP : \lambda x.size(x)$

Semantic parsing

Zettlemoyer & Collins 2005:

Rules		Categories produced from logical form $\arg\max(\lambda x.state(x) \wedge borders(x, texas), \lambda x.size(x))$
Input Trigger	Output Category	
constant c	$NP : c$	$NP : texas$
arity one predicate p_1	$N : \lambda x.p_1(x)$	$N : \lambda x.state(x)$
arity one predicate p_1	$S\backslash NP : \lambda x.p_1(x)$	$S\backslash NP : \lambda x.state(x)$
arity two predicate p_2	$(S\backslash NP)/NP : \lambda x.\lambda y.p_2(y, x)$	$(S\backslash NP)/NP : \lambda x.\lambda y.borders(y, x)$
arity two predicate p_2	$(S\backslash NP)/NP : \lambda x.\lambda y.p_2(x, y)$	$(S\backslash NP)/NP : \lambda x.\lambda y.borders(x, y)$
arity one predicate p_1	$N/N : \lambda g.\lambda x.p_1(x) \wedge g(x)$	$N/N : \lambda g.\lambda x.state(x) \wedge g(x)$
literal with arity two predicate p_2 and constant second argument c	$N/N : \lambda g.\lambda x.p_2(x, c) \wedge g(x)$	$N/N : \lambda g.\lambda x.borders(x, texas) \wedge g(x)$
arity two predicate p_2	$(N\backslash N)/NP : \lambda x.\lambda g.\lambda y.p_2(x, y) \wedge g(x)$	$(N\backslash N)/NP : \lambda g.\lambda x.\lambda y.borders(x, y) \wedge g(x)$
an $\arg\max$ / $\min$ with second argument arity one function f	$NP/N : \lambda g.\arg\max / \min(g, \lambda x.f(x))$	$NP/N : \lambda g.\arg\max(g, \lambda x.size(x))$
an arity one numeric-ranged function f	$S/NP : \lambda x.f(x)$	$S/NP : \lambda x.size(x)$

constant c	$NP : c$

Semantic parsing

Zettlemoyer & Collins 2005:

Rules		Categories produced from logical form
Input Trigger	Output Category	$\arg\max(\lambda x.state(x) \wedge borders(x, texas), \lambda x.size(x))$
constant c	$NP : c$	$NP : texas$
arity one predicate p_1	$N : \lambda x.p_1(x)$	$N : \lambda x.state(x)$
arity one predicate p_1	$S\backslash NP : \lambda x.p_1(x)$	$S\backslash NP : \lambda x.state(x)$
arity two predicate p_2	$(S\backslash NP)/NP : \lambda x.\lambda y.p_2(y, x)$	$(S\backslash NP)/NP : \lambda x.\lambda y.borders(y, x)$
arity two predicate p_2	$(S\backslash NP)/NP : \lambda x.\lambda y.p_2(x, y)$	$(S\backslash NP)/NP : \lambda x.\lambda y.borders(x, y)$
arity one predicate p_1	$N/N : \lambda g.\lambda x.p_1(x) \wedge g(x)$	$N/N : \lambda g.\lambda x.state(x) \wedge g(x)$
literal with arity two predicate p_2 and constant second argument c	$N/N : \lambda g.\lambda x.p_2(x, c) \wedge g(x)$	$N/N : \lambda g.\lambda x.borders(x, texas) \wedge g(x)$
arity two predicate p_2	$(N\backslash N)/NP : \lambda x.\lambda g.\lambda y.p_2(x, y) \wedge g(x)$	$(N\backslash N)/NP : \lambda g.\lambda x.\lambda y.borders(x, y) \wedge g(x)$
an $\arg\max$ / $\min$ with second argument arity one function f	$NP/N : \lambda g.\arg\max / \min(g, \lambda x.f(x))$	$NP/N : \lambda g.\arg\max(g, \lambda x.size(x))$
an arity one numeric-ranged function f	$S/NP : \lambda x.f(x)$	$S/NP : \lambda x.size(x)$

arity one predicate p_1	$N : \lambda x.p_1(x)$
arity one predicate p_1	$S\backslash NP : \lambda x.p_1(x)$

The Rational Speech Acts Model (RSA)

1. The Rational Speech Acts (RSA) model
2. TUNA
3. Learned RSA
4. Experiments

Grice

Cooperative principle: Make your contribution as is required, when it is required, by the conversation in which you are engaged.

Grice

Cooperative principle: Make your contribution as is required, when it is required, by the conversation in which you are engaged.

- Quantity
- Quality
- Relevance
- Manner

Grice

Cooperative principle: Make your contribution as is required, when it is required, by the conversation in which you are engaged.

- Quantity
- Quality
- Relevance
- Manner
- Politeness

Grice

Cooperative principle: Make your contribution as is required, when it is required, by the conversation in which you are engaged.

- Quantity
- Quality
- Relevance
- Manner
- Politeness
- Stylishness

Grice

Cooperative principle: Make your contribution as is required, when it is required, by the conversation in which you are engaged.

- Quantity
- Quality
- Relevance
- Manner
- Politeness
- Stylishness

- Q principle
- R principle

Grice

Cooperative principle: Make your contribution as is required, when it is required, by the conversation in which you are engaged.

- Quantity
- Quality
- Relevance
- Manner
- Politeness
- Stylishness

- Q principle
- R principle

- Q heuristic
- I heuristic
- M heuristic

Grice

Cooperative principle: Make your contribution as is required, when it is required, by the conversation in which you are engaged.

- Quantity
- Quality
- Relevance
- Manner
- Politeness
- Stylishness

- Q principle
- R principle

- Q heuristic
- I heuristic
- M heuristic

Conversational implicature

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Example

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Example

Ann: What city does Paul live in?
Bob: Hmm . . . he lives in California.

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Example

Ann: What city does Paul live in?

Bob: Hmm . . . he lives in California.

(A) Assume Bob is cooperative.

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Example

Ann: What city does Paul live in?

Bob: Hmm . . . he lives in California.

(A) Assume Bob is cooperative.

(B) Bob supplied less information than required; clash with (A).

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Example

Ann: What city does Paul live in?

Bob: Hmm . . . he lives in California.

(A) Assume Bob is cooperative.

(B) Bob supplied less information than required; clash with (A).

(C) **Assume Bob does not know which city Paul lives in.**

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Example

Ann: What city does Paul live in?
Bob: Hmm . . . he lives in California.

(A) Assume Bob is cooperative.
(B) Bob supplied less information than required; clash with (A).
(C) **Assume Bob does not know which city Paul lives in.**
(D) Then Bob's answer is optimal given his evidence.

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Implicature as

- Rooted in cooperativity
- Social, interactional
- Cognitively complex
- Error-driven

Conversational implicature

Definition

Speaker S saying U to listener L conversationally implicates q iff

1. S and L mutually, publicly presume that S is cooperative.
2. To maintain 1 given U, it must be supposed that S thinks q.
3. S thinks that both S and L mutually, publicly presume that L is willing and able to work out that 2 holds.

Implicature as

- Rooted in cooperativity
- Social, interactional
- Cognitively complex
- **Error-driven**

Pragmatic listeners

Pragmatic listeners

Definition (Literal listener)

$$l_0(w \mid msg, Lex) \propto Lex(msg, w)P(w)$$

Pragmatic listeners

Definition (Pragmatic speaker)

$$s_1(msg \mid w, Lex) \propto \exp \lambda (\log l_0(w \mid msg, Lex) - C(msg))$$

Definition (Literal listener)

$$l_0(w \mid msg, Lex) \propto Lex(msg, w)P(w)$$

Pragmatic listeners

Definition (Pragmatic listener)

$$l_1(w \mid msg, Lex) \propto s_1(msg \mid w, Lex)P(w)$$

Definition (Pragmatic speaker)

$$s_1(msg \mid w, Lex) \propto \exp \lambda (\log l_0(w \mid msg, Lex) - C(msg))$$

Definition (Literal listener)

$$l_0(w \mid msg, Lex) \propto Lex(msg, w)P(w)$$

Pragmatic listeners

Definition (Pragmatic listener)

$$l_1(w \mid msg, Lex) = \textbf{pragmatic speaker} \times \text{state prior}$$

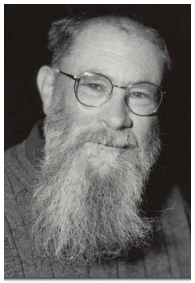
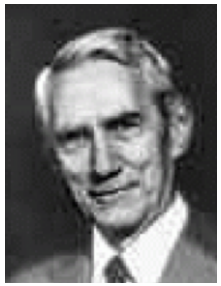
Definition (Pragmatic speaker)

$$s_1(msg \mid w, Lex) = \textbf{literal listener} - \text{message costs}$$

Definition (Literal listener)

$$l_0(w \mid msg, Lex) = \textbf{lexicon} \times \text{state prior}$$

RSA listener example

			
beard	1	0	0
glasses	1	1	0
tie	0	1	1

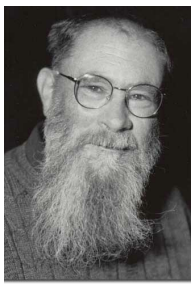
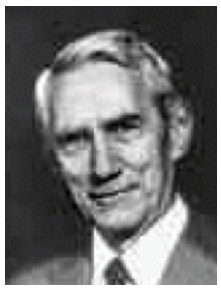
l_1

s_1

l_0

Lex

RSA listener example

			
beard	**1**	0	0
glasses	.5	.5	0
tie	0	.5	.5

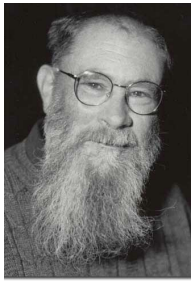
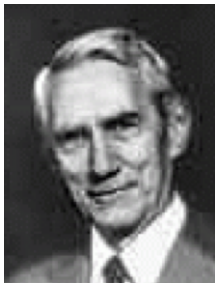
l_1

s_1

l_0

Lex

RSA listener example

	beard	*glasses*	*tie*
	**.67**	.33	0
	0	**1**	0
	0	0	**1**

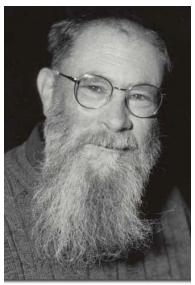
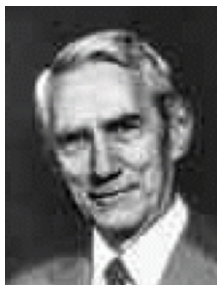
l_1

s_1

l_0

Lex

RSA listener example

			
beard	**1**	0	0
glasses	.25	**.75**	0
tie	0	0	**1**

l_1

s_1

l_0

Lex

More Gricean terrain

More Gricean terrain

- Lexical uncertainty and the division of pragmatic labor

  Bergen, Levy, Goodman, ‘Pragmatic reasoning through semantic inference’

More Gricean terrain

- Lexical uncertainty and the division of pragmatic labor

  Bergen, Levy, Goodman, 'Pragmatic reasoning through semantic inference'

- Lexical uncertainty and embedded implicatures

  Potts, Lassiter, Levy, Frank, 'Embedded implicatures as pragmatic inferences under compositional lexical uncertainty'

More Gricean terrain

- Lexical uncertainty and the division of pragmatic labor

  Bergen, Levy, Goodman, 'Pragmatic reasoning through semantic inference'

- Lexical uncertainty and embedded implicatures

  Potts, Lassiter, Levy, Frank, 'Embedded implicatures as pragmatic inferences under compositional lexical uncertainty'

- Pragmatic free variables and non-literal language

  Kao, Wu, Bergen, Goodman, 'Nonliteral understanding of number words'

More Gricean terrain

- Lexical uncertainty and the division of pragmatic labor

  Bergen, Levy, Goodman, 'Pragmatic reasoning through semantic inference'

- Lexical uncertainty and embedded implicatures

  Potts, Lassiter, Levy, Frank, 'Embedded implicatures as pragmatic inferences under compositional lexical uncertainty'

- Pragmatic free variables and non-literal language

  Kao, Wu, Bergen, Goodman, 'Nonliteral understanding of number words'

- Implicature blocking by higher-level agents

  Potts & Levy, 'Negotiating lexical uncertainty and speaker expertise with disjunction'

Pragmatic speakers

Pragmatic speakers

Definition (Literal speaker)

$$s_0(msg \mid w, Lex) \propto \exp \lambda \left(\log Lex(msg, w) - C(msg)\right)$$

Pragmatic speakers

Definition (Pragmatic listener)

$$l_1(w \mid msg, Lex) \propto s_0(msg \mid w, Lex)P(w)$$

Definition (Literal speaker)

$$s_0(msg \mid w, Lex) \propto \exp \lambda (\log Lex(msg, w) - C(msg))$$

Pragmatic speakers

Definition (Pragmatic speaker)

$$s_1(msg \mid w, Lex) \propto \exp \lambda (\log l_1(w \mid msg, Lex) - C(msg))$$

Definition (Pragmatic listener)

$$l_1(w \mid msg, Lex) \propto s_0(msg \mid w, Lex) P(w)$$

Definition (Literal speaker)

$$s_0(msg \mid w, Lex) \propto \exp \lambda (\log Lex(msg, w) - C(msg))$$

Pragmatic speakers

Definition (Pragmatic speaker)

$$s_1(msg \mid w, Lex) = \textbf{pragmatic listener} - \text{message costs}$$

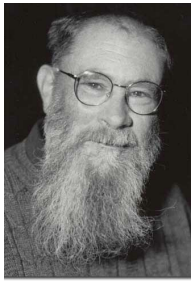
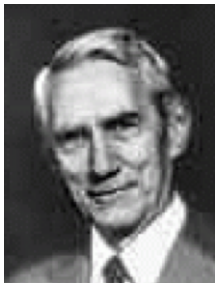
Definition (Pragmatic listener)

$$l_1(w \mid msg, Lex) = \textbf{literal speaker} \times \text{state prior}$$

Definition (Literal speaker)

$$s_0(msg \mid w, Lex) = \textbf{lexicon} - \text{message costs}$$

RSA speaker example

	beard	*glasses*	*tie*
	1	1	0
	0	1	1
	0	0	1

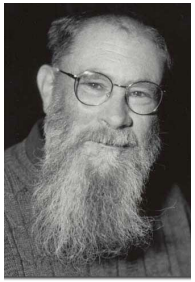
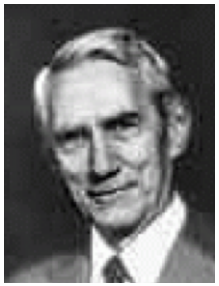
s_1

l_1

s_0

Lex

RSA speaker example

	beard	*glasses*	*tie*
	.5	.5	0
	0	.5	.5
	0	0	**1**

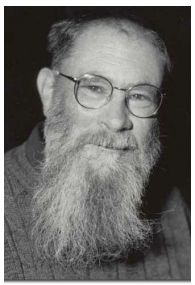
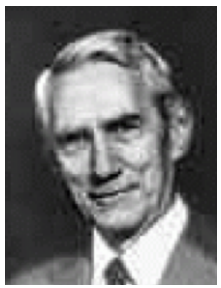
s_1

l_1

s_0

Lex

RSA speaker example

			
beard	**1**	0	0
glasses	.5	.5	0
tie	0	.33	**.67**

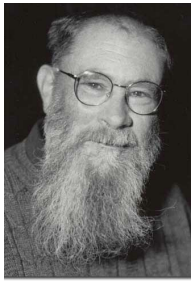
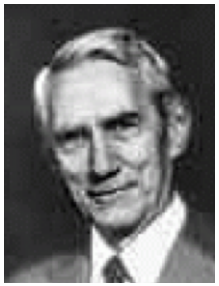
s_1

l_1

s_0

Lex

RSA speaker example

	beard	*glasses*	*tie*
	**.67**	.33	0
	0	**.6**	.4
	0	0	**1**

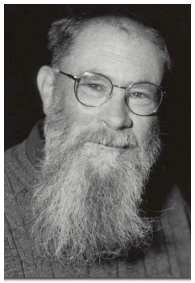
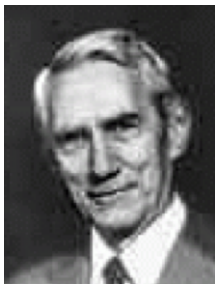
s_1

l_1

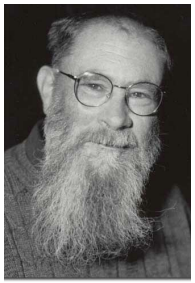
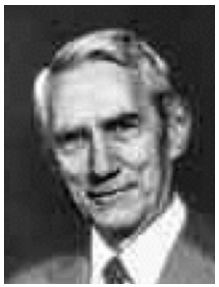
s_0

Lex

Achievements and drawbacks

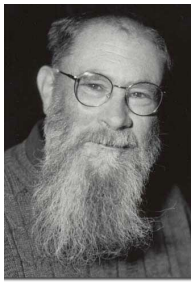
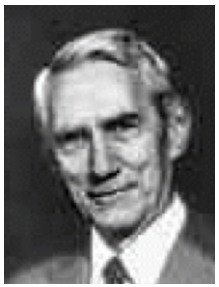
	beard	*glasses*	*tie*
	**.67**	.33	0
	0	**.6**	.4
	0	0	**1**

Achievements and drawbacks

	beard	*glasses*	*tie*
	**.67**	.33	0
	0	**.6**	.4
	0	0	**1**

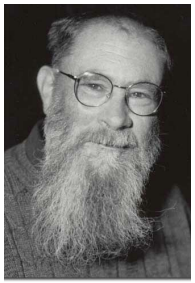
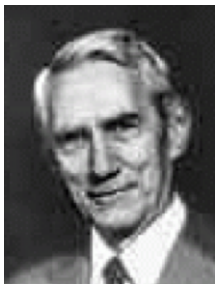
- Cognitive demands limit speaker rationality

Achievements and drawbacks

	beard	*glasses*	*tie*
	**.67**	.33	0
	0	**.6**	.4
	0	0	**1**

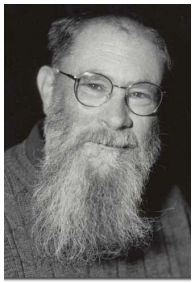
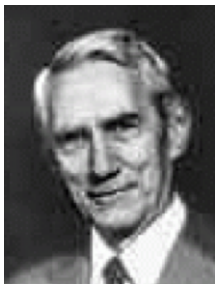
- Cognitive demands limit speaker rationality
- Speaker preferences

Achievements and drawbacks

	beard	*glasses*	*tie*
	**.67**	.33	0
	0	**.6**	.4
	0	0	**1**

- Cognitive demands limit speaker rationality
- Speaker preferences
- Hand-specified lexicon

Achievements and drawbacks

	beard	*glasses*	*tie*
	**.67**	.33	0
	0	**.6**	.4
	0	0	**1**

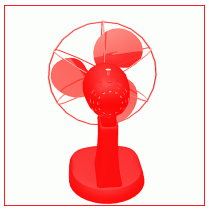
- Cognitive demands limit speaker rationality
- Speaker preferences
- Hand-specified lexicon
- High-bias model; few chances to learn from data

TUNA

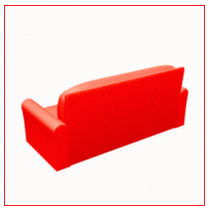
Furniture example

COLOUR:GREEN
ORIENTATION:LEFT
SIZE:SMALL
TYPE:FAN
X-DIMENSION:1
Y-DIMENSION:1

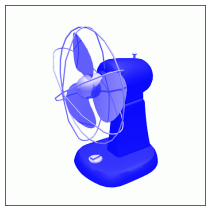
COLOUR:GREEN
ORIENTATION:LEFT
SIZE:SMALL
TYPE:SOFA
X-DIMENSION:1
Y-DIMENSION:2



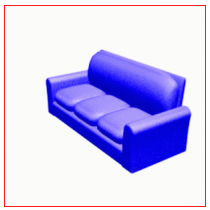
COLOUR:RED
ORIENTATION:BACK
SIZE:LARGE
TYPE:FAN
X-DIMENSION:1
Y-DIMENSION:3



COLOUR:RED
ORIENTATION:BACK
SIZE:LARGE
TYPE:SOFA
X-DIMENSION:2
Y-DIMENSION:1



COLOUR:BLUE
ORIENTATION:LEFT
SIZE:LARGE
TYPE:FAN
X-DIMENSION:2
Y-DIMENSION:2



COLOUR:BLUE
ORIENTATION:LEFT
SIZE:LARGE
TYPE:SOFA
X-DIMENSION:3
Y-DIMENSION:1

COLOUR:BLUE
ORIENTATION:LEFT
SIZE:SMALL
TYPE:FAN
X-DIMENSION:3
Y-DIMENSION:3

Utterance: “blue fan small”
Utterance attributes: [*colour:blue*]; [*size:small*]; [*type:fan*]

Furniture example

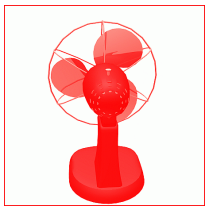
Utterance: “blue fan small”

Utterance attributes: [*colour:blue*]; [*size:small*]; [*type:fan*]

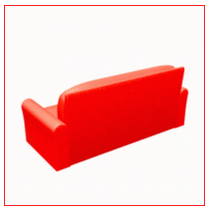
Furniture example

COLOUR:GREEN
ORIENTATION:LEFT
SIZE:SMALL
TYPE:FAN
X-DIMENSION:1
Y-DIMENSION:1

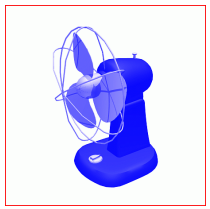
COLOUR:GREEN
ORIENTATION:LEFT
SIZE:SMALL
TYPE:SOFA
X-DIMENSION:1
Y-DIMENSION:2



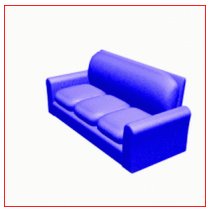
COLOUR:RED
ORIENTATION:BACK
SIZE:LARGE
TYPE:FAN
X-DIMENSION:1
Y-DIMENSION:3



COLOUR:RED
ORIENTATION:BACK
SIZE:LARGE
TYPE:SOFA
X-DIMENSION:2
Y-DIMENSION:1



COLOUR:BLUE
ORIENTATION:LEFT
SIZE:LARGE
TYPE:FAN
X-DIMENSION:2
Y-DIMENSION:2



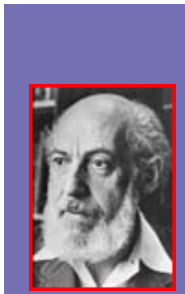
COLOUR:BLUE
ORIENTATION:LEFT
SIZE:LARGE
TYPE:SOFA
X-DIMENSION:3
Y-DIMENSION:1

COLOUR:BLUE
ORIENTATION:LEFT
SIZE:SMALL
TYPE:FAN
X-DIMENSION:3
Y-DIMENSION:3

Utterance: “blue fan small”

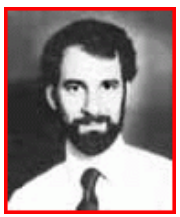
Utterance attributes: [*colour:blue*]; [*size:small*]; [*type:fan*]

People example



AGE:OLD
HAIRCOLOUR:LIGHT
HASBEARD:1
HASGLASSES:0
HASHAIR:0
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:LEFT
TYPE:PERSON
X-DIMENSION:1
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:1
Y-DIMENSION:2



AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:2
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:2
Y-DIMENSION:2

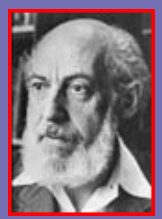
AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:2

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:3

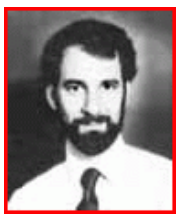
Utterance: The bald man with a beard
[*hasBeard:1*]; [*hasHair:0*]; [*type:person*]

People example



AGE:OLD
HAIRCOLOUR:LIGHT
HASBEARD:1
HASGLASSES:0
HASHAIR:0
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:LEFT
TYPE:PERSON
X-DIMENSION:1
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:1
Y-DIMENSION:2



AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:2
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:2
Y-DIMENSION:2

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:1

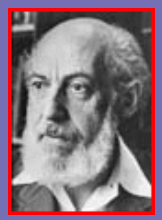
AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:2

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:3

Utterance: The bald man with a beard

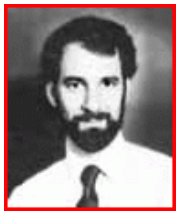
[*hasBeard:1*]; [*hasHair:0*]; [*type:person*]

People example



AGE:OLD
HAIRCOLOUR:LIGHT
HASBEARD:1
HASGLASSES:0
HASHAIR:0
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:LEFT
TYPE:PERSON
X-DIMENSION:1
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:1
Y-DIMENSION:2



AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:2
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:2
Y-DIMENSION:2

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:1

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:1
HASGLASSES:0
HASHAIR:1
HASSHIRT:1
HASSUIT:0
HASTIE:0
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:2

AGE:YOUNG
HAIRCOLOUR:DARK
HASBEARD:0
HASGLASSES:0
HASHAIR:1
HASSHIRT:0
HASSUIT:1
HASTIE:1
ORIENTATION:FRONT
TYPE:PERSON
X-DIMENSION:3
Y-DIMENSION:3

Utterance: The bald man with a beard

[*hasBeard:1*]; [*hasHair:0*]; [*type:person*]

Multiset Dice coefficient

Definition

$$\frac{2 \sum_{x \in D} \min \left[\mathbb{Z}_{a(msg_i)}(x), \mathbb{Z}_{a(msg_j)}(x) \right]}{|a(msg_i)| + |a(msg_j)|}$$

Multiset Dice coefficient

Definition

$$\frac{\textbf{Multiset intersection cardinality}}{\textbf{Multiset union cardinality}}$$

Multiset Dice coefficient

Definition

$$\frac{\textbf{Multiset intersection cardinality}}{\textbf{Multiset union cardinality}}$$

- $\begin{matrix} \textit{predicted}: [a & b & c & a] \\ \textit{actual}: [a & b & c & a] \end{matrix} = 1$

Multiset Dice coefficient

Definition

$$\frac{\textbf{Multiset intersection cardinality}}{\textbf{Multiset union cardinality}}$$

- $\begin{matrix} \textit{predicted}: [a & b & c & a] \\ \textit{actual}: [a & b & c & a] \end{matrix} = 1$
- $\begin{matrix} \textit{predicted}: [a & b & c & a] \\ \textit{actual}: [a & b & c & \ ] \end{matrix} = .86$

Multiset Dice coefficient

Definition

$$\frac{\textbf{Multiset intersection cardinality}}{\textbf{Multiset union cardinality}}$$

- $\begin{matrix} \textit{predicted}: [a & b & c & a] \\ \textit{actual}: [a & b & c & a] \end{matrix} = 1$
- $\begin{matrix} \textit{predicted}: [a & b & c & a] \\ \textit{actual}: [a & b & c & \ ] \end{matrix} = .86$
- $\begin{matrix} \textit{predicted}: [a & b & c & \ ] \\ \textit{actual}: [a & b & c & a] \end{matrix} = .86$

Multiset Dice coefficient

Definition

$$\frac{\textbf{Multiset intersection cardinality}}{\textbf{Multiset union cardinality}}$$

- $\begin{array}{rl} \textit{predicted}: & [a \quad b \quad c \quad a] \\ \textit{actual}: & [a \quad b \quad c \quad a] \end{array} = 1$
- $\begin{array}{rl} \textit{predicted}: & [a \quad b \quad c \quad a] \\ \textit{actual}: & [a \quad b \quad c \quad\ \ ] \end{array} = .86$
- $\begin{array}{rl} \textit{predicted}: & [a \quad b \quad c \quad\ \ ] \\ \textit{actual}: & [a \quad b \quad c \quad a] \end{array} = .86$
- $\begin{array}{rl} \textit{predicted}: & [a \quad\quad\quad\ \ ] \\ \textit{actual}: & [a \quad b \quad c \quad a] \end{array} = .4$

Learned RSA

Feature representations

Feature representations

Target	Utterance attributes	Features
	[*colour:blue*] [*size:small*] [*type:fan*]	

Feature representations

Target	Utterance attributes	Features
 COLOUR:BLUE ORIENTATION:LEFT SIZE:SMALL TYPE:FAN X-DIMENSION:3 Y-DIMENSION:3	[*colour:blue*] [*size:small*] [*type:fan*]	COLOUR:BLUE $\wedge$ [*colour:blue*] COLOUR:BLUE $\wedge$ [*size:small*] COLOUR:BLUE $\wedge$ [*type:fan*] ORIENTATION:LEFT $\wedge$ [*colour:blue*] ORIENTATION:LEFT $\wedge$ [*size:small*] ORIENTATION:LEFT $\wedge$ [*type:fan*] $\vdots$

Feature representations

Target	Utterance attributes	Features
 COLOUR:BLUE ORIENTATION:LEFT SIZE:SMALL TYPE:FAN X-DIMENSION:3 Y-DIMENSION:3	[*colour:blue*] [*size:small*] [*type:fan*]	COLOUR:BLUE ∧ [*colour:blue*] COLOUR:BLUE ∧ [*size:small*] COLOUR:BLUE ∧ [*type:fan*] ORIENTATION:LEFT ∧ [*colour:blue*] ORIENTATION:LEFT ∧ [*size:small*] ORIENTATION:LEFT ∧ [*type:fan*] ⋮
		color
	Generation features	

Feature representations

Target	Utterance attributes	Features
 COLOUR:BLUE ORIENTATION:LEFT SIZE:SMALL TYPE:FAN X-DIMENSION:3 Y-DIMENSION:3	[*colour:blue*] [*size:small*] [*type:fan*]	COLOUR:BLUE ∧ [*colour:blue*] COLOUR:BLUE ∧ [*size:small*] COLOUR:BLUE ∧ [*type:fan*] ORIENTATION:LEFT ∧ [*colour:blue*] ORIENTATION:LEFT ∧ [*size:small*] ORIENTATION:LEFT ∧ [*type:fan*] ⋮
	Generation features	**color** **type** + **color** **color** + ¬**size**

type ≫ **color** ≫ **size**

type ≫ **orientation** ≫ **color** ≫ **size**

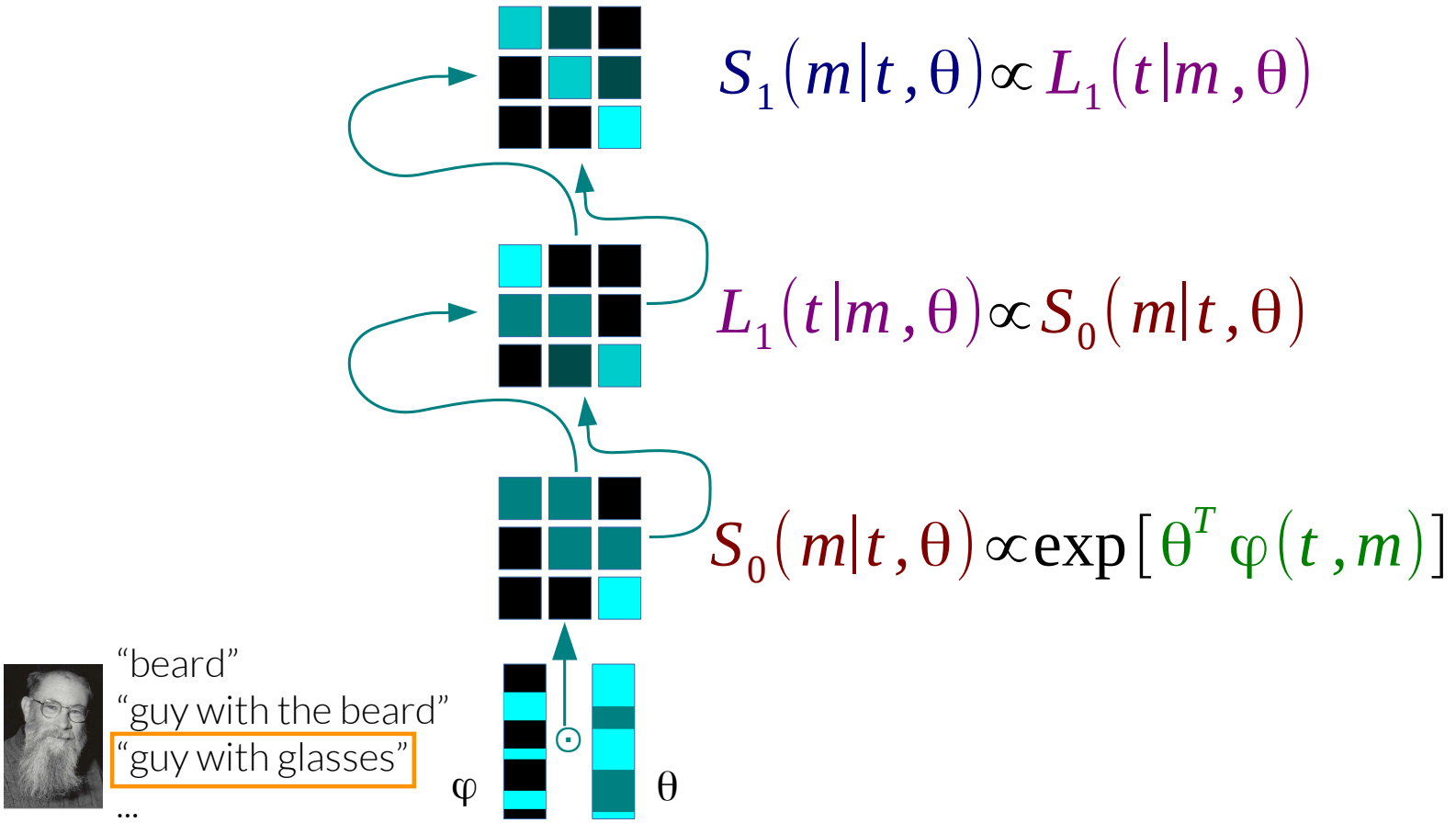
Feature representations

Target	Utterance attributes	Features
	[*colour:blue*] [*size:small*] [*type:fan*]	COLOUR:BLUE $\wedge$ [*colour:blue*] COLOUR:BLUE $\wedge$ [*size:small*] COLOUR:BLUE $\wedge$ [*type:fan*] ORIENTATION:LEFT $\wedge$ [*colour:blue*] ORIENTATION:LEFT $\wedge$ [*size:small*] ORIENTATION:LEFT $\wedge$ [*type:fan*] $\vdots$
	Generation features	**color** **type** $+$ **color** **color** $+ \neg$**size** **attribute-count** $= 3$

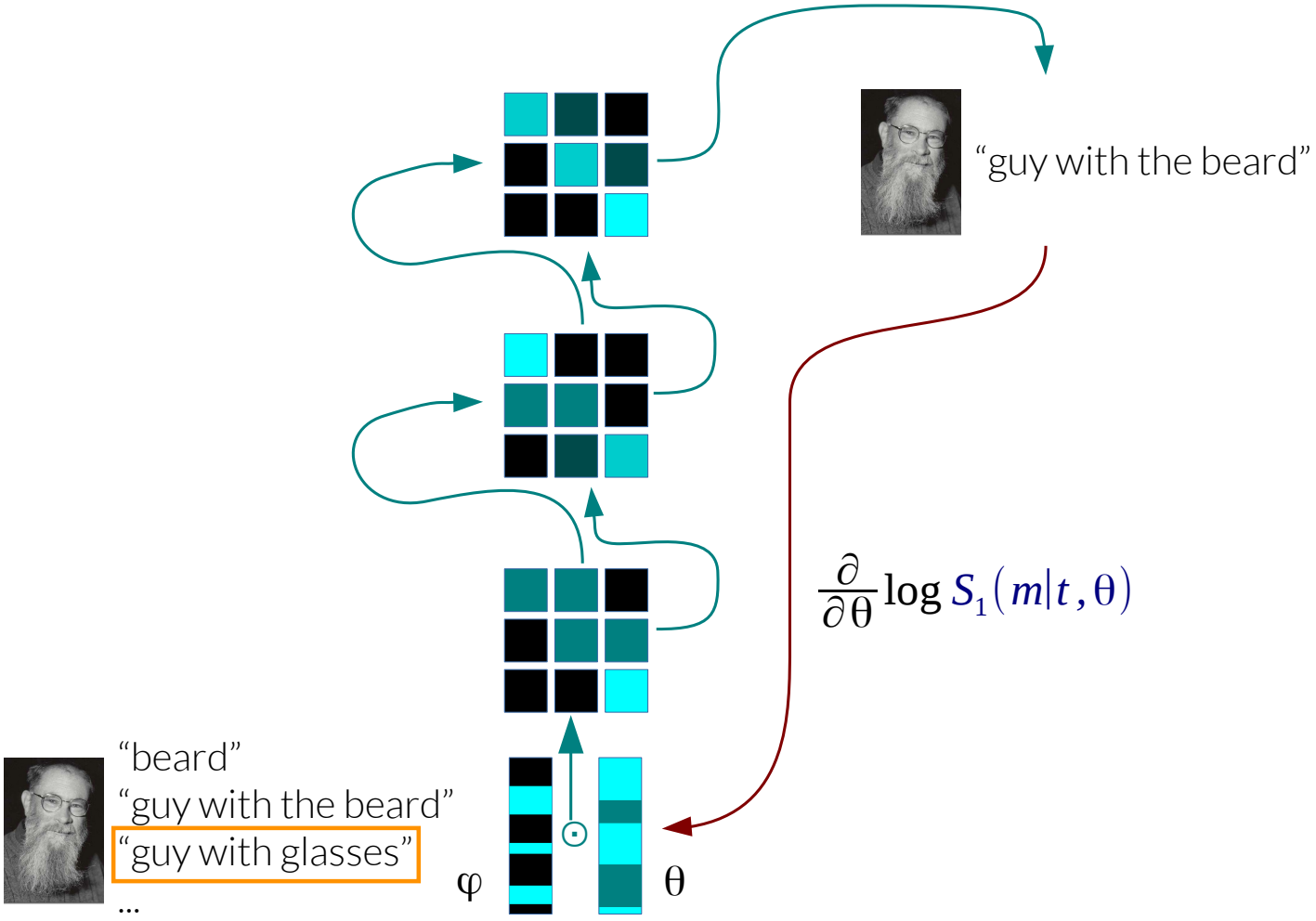
type $\gg$ **color** $\gg$ **size**

type $\gg$ **orientation** $\gg$ **color** $\gg$ **size**

Model definition



Optimization



Addressing the drawbacks of RSA

Goal	Features

Addressing the drawbacks of RSA

Goal	Features
Avoid hand-built lexicon	

Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon	Cross-product features

Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon Learn quirks of production	Cross-product features

Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon	Cross-product features
Learn quirks of production	Features like **color**

Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon	Cross-product features
Learn quirks of production	Features like **color**
Learn attribute hierarchies	

Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon	Cross-product features
Learn quirks of production	Features like **color**
Learn attribute hierarchies	Attribute-pair features like **color** $+$ $\neg$**size**

Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon	Cross-product features
Learn quirks of production	Features like **color**
Learn attribute hierarchies	Attribute-pair features like **color** $+$ $\neg$**size**
Learn message costs	

Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon	Cross-product features
Learn quirks of production	Features like **color**
Learn attribute hierarchies	Attribute-pair features like **color** $+$ $\neg$**size**
Learn message costs	Length features and others

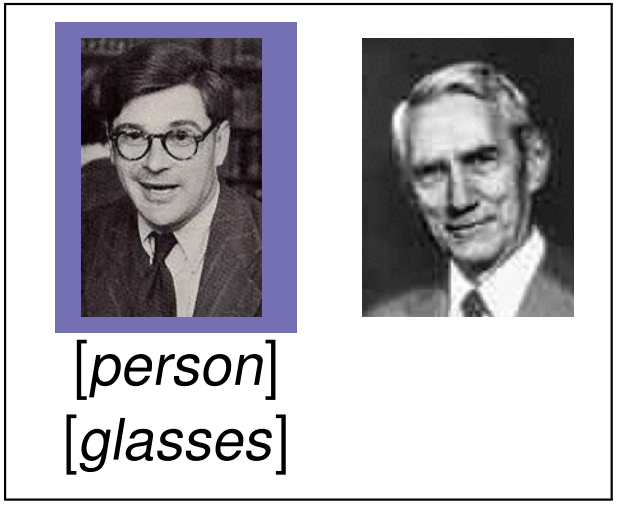
Addressing the drawbacks of RSA

Goal	**Features**
Avoid hand-built lexicon	Cross-product features
Learn quirks of production	Features like **color**
Learn attribute hierarchies	Attribute-pair features like **color** $+$ $\neg$**size**
Learn message costs	Length features and others

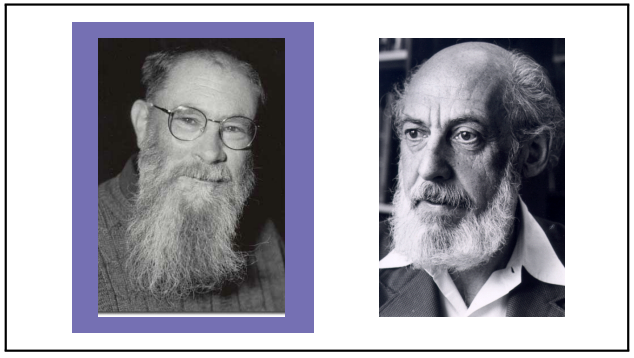
Cognitive and linguistic insights combined with learning

Example

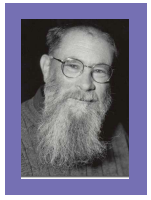
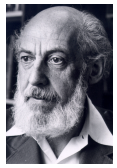
Train



Test



Example

$\emptyset$

[*person*]

[*glasses*]

[*beard*]

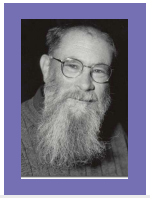
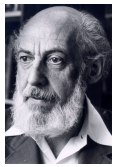
[*person*];[*glasses*]

[*person*];[*beard*]

[*glasses*];[*beard*]

[*all*]

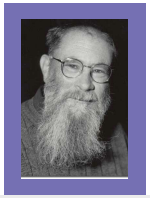
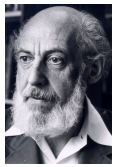
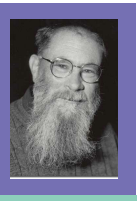
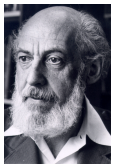
Example

∅	.08	.25
[*person*]	.08	.25
[*glasses*]	.17	.00
[*beard*]	.08	.25
[*person*];[*glasses*]	.17	.00
[*person*];[*beard*]	.08	.25
[*glasses*];[*beard*]	.17	.00
[*all*]	.17	.00

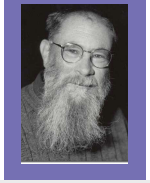
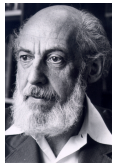
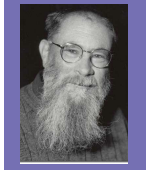
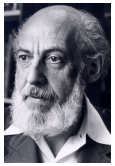
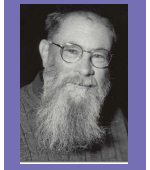
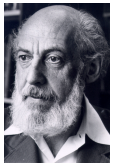
RSA

Example

∅	.08	.25	.03	.00
[*person*]	.08	.25	.22	.10
[*glasses*]	.17	.00	.03	.00
[*beard*]	.08	.25	.03	.04
[*person*];[*glasses*]	.17	.00	.22	.01
[*person*];[*beard*]	.08	.25	.22	.74
[*glasses*];[*beard*]	.17	.00	.03	.00
[*all*]	.17	.00	.22	.10
	RSA		**Learned** S_0	

Example

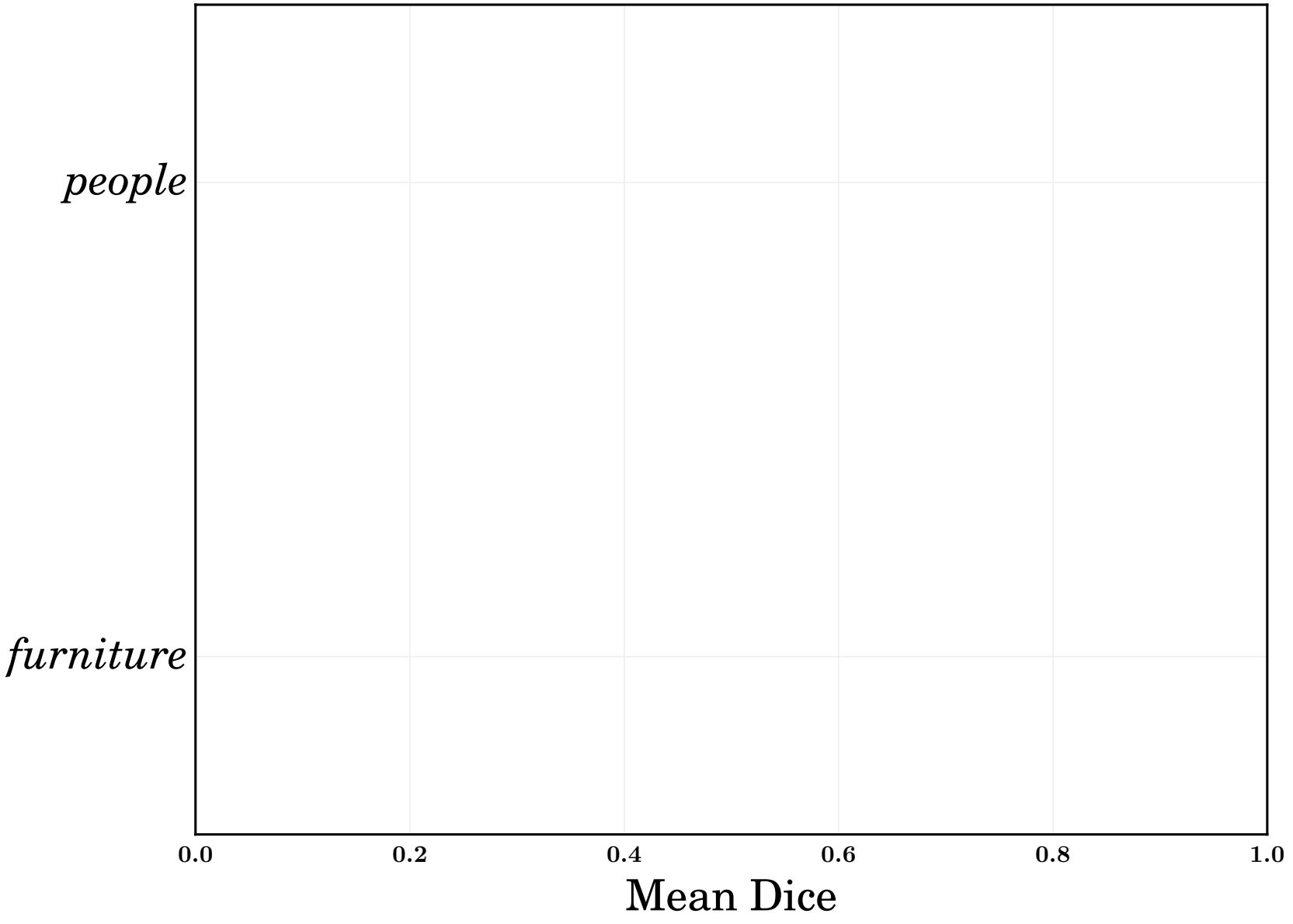
						
∅	.08	.25	.03	.00	.10	.11
[*person*]	.08	.25	.22	.10	.16	.13
[*glasses*]	.17	.00	.03	.00	.11	.07
[*beard*]	.08	.25	.03	.04	.08	.17
[*person*];[*glasses*]	.17	.00	.22	.01	.18	.08
[*person*];[*beard*]	.08	.25	.22	.74	.12	.19
[*glasses*];[*beard*]	.17	.00	.03	.00	.10	.11
[*all*]	.17	.00	.22	.10	.16	.11
	RSA		**Learned S_0**		**Learned S_1**	

Experiments

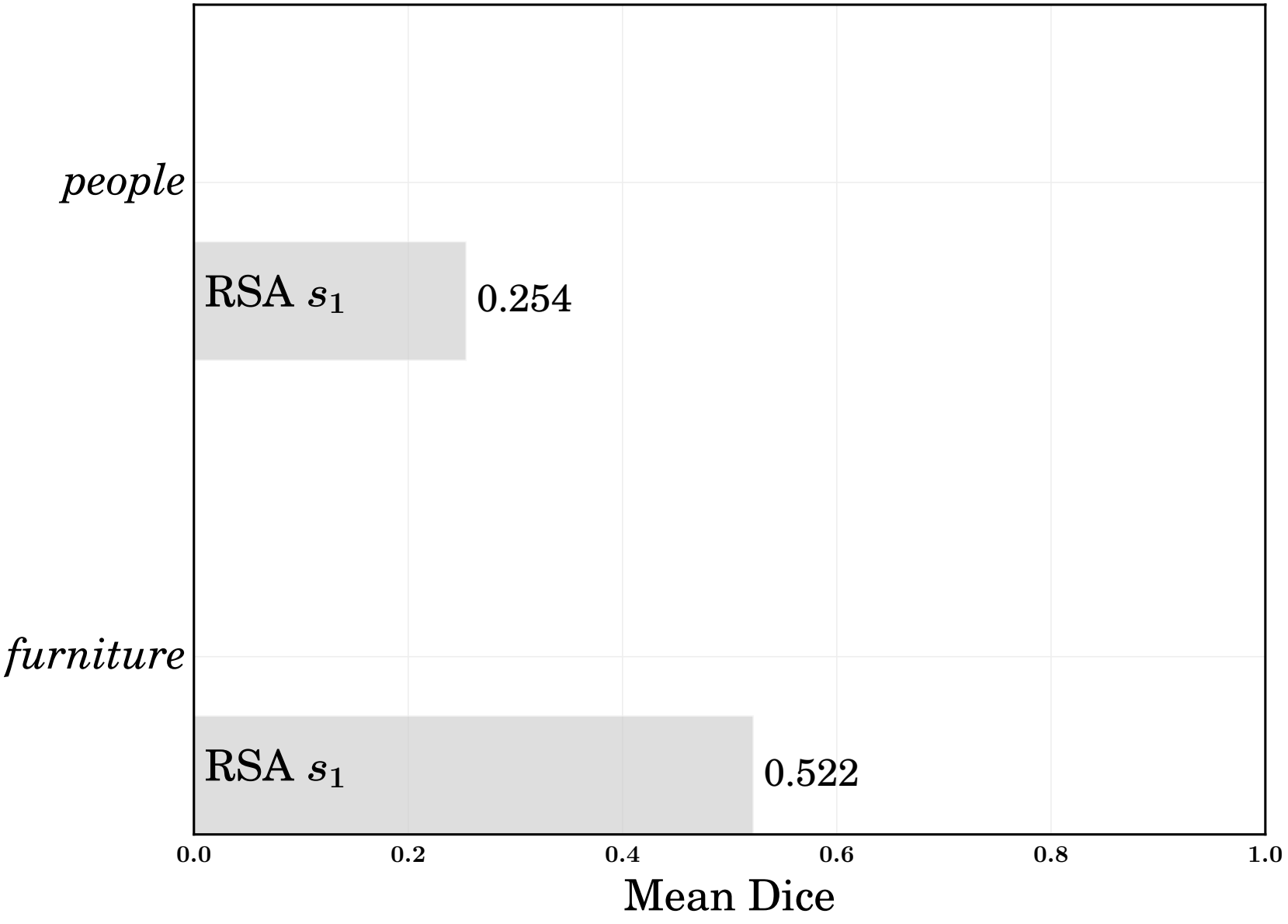
Experimental set-up

- TUNA *furniture*: 420 trials, 176 distinct referents
- TUNA *people*: 360 trials, 228 distinct referents
- Five-fold cross-validation for all models
- RSA cross-validation sets λ and the cost function
- Learned RSA optimized via AdaGrad with ℓ_2 regularization

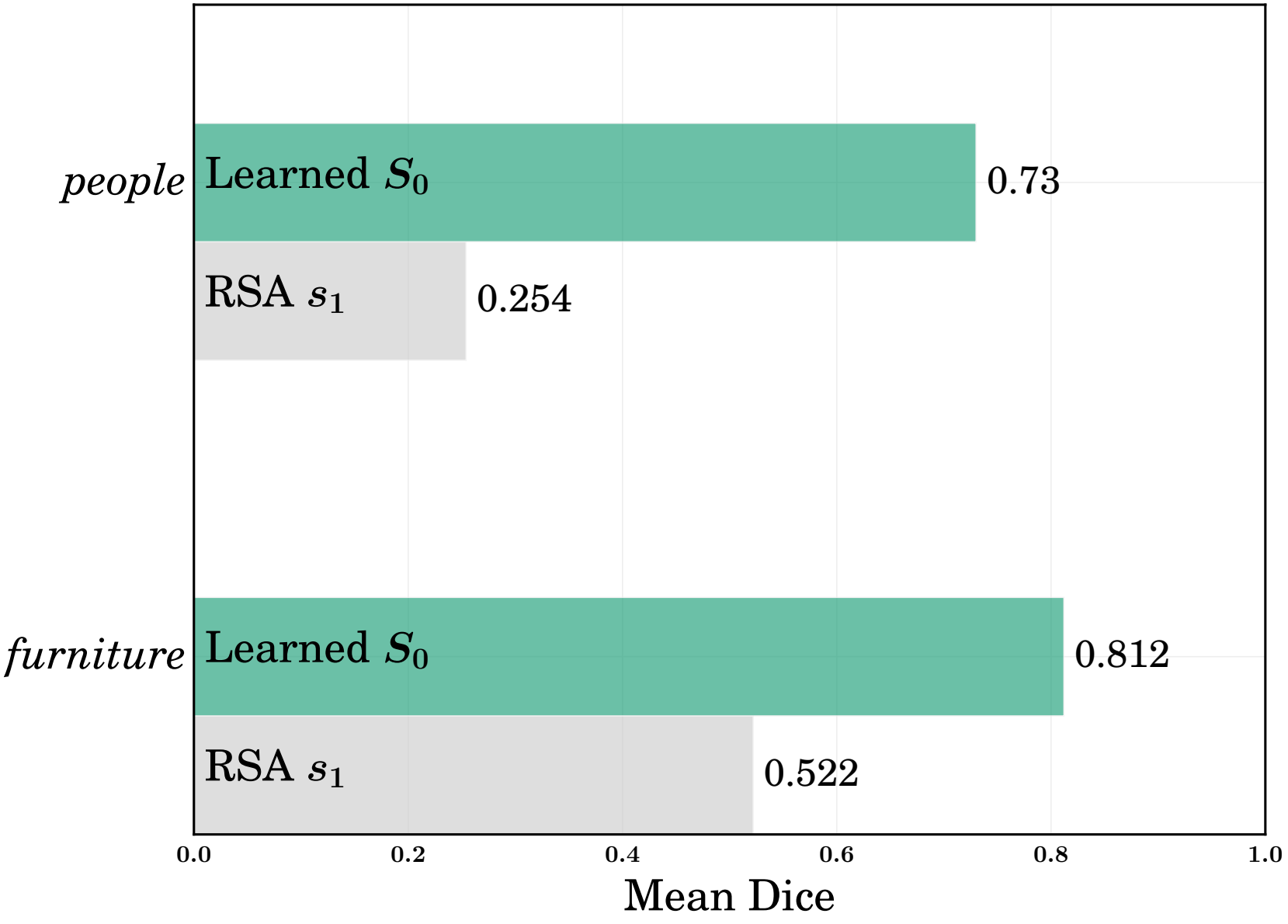
Results



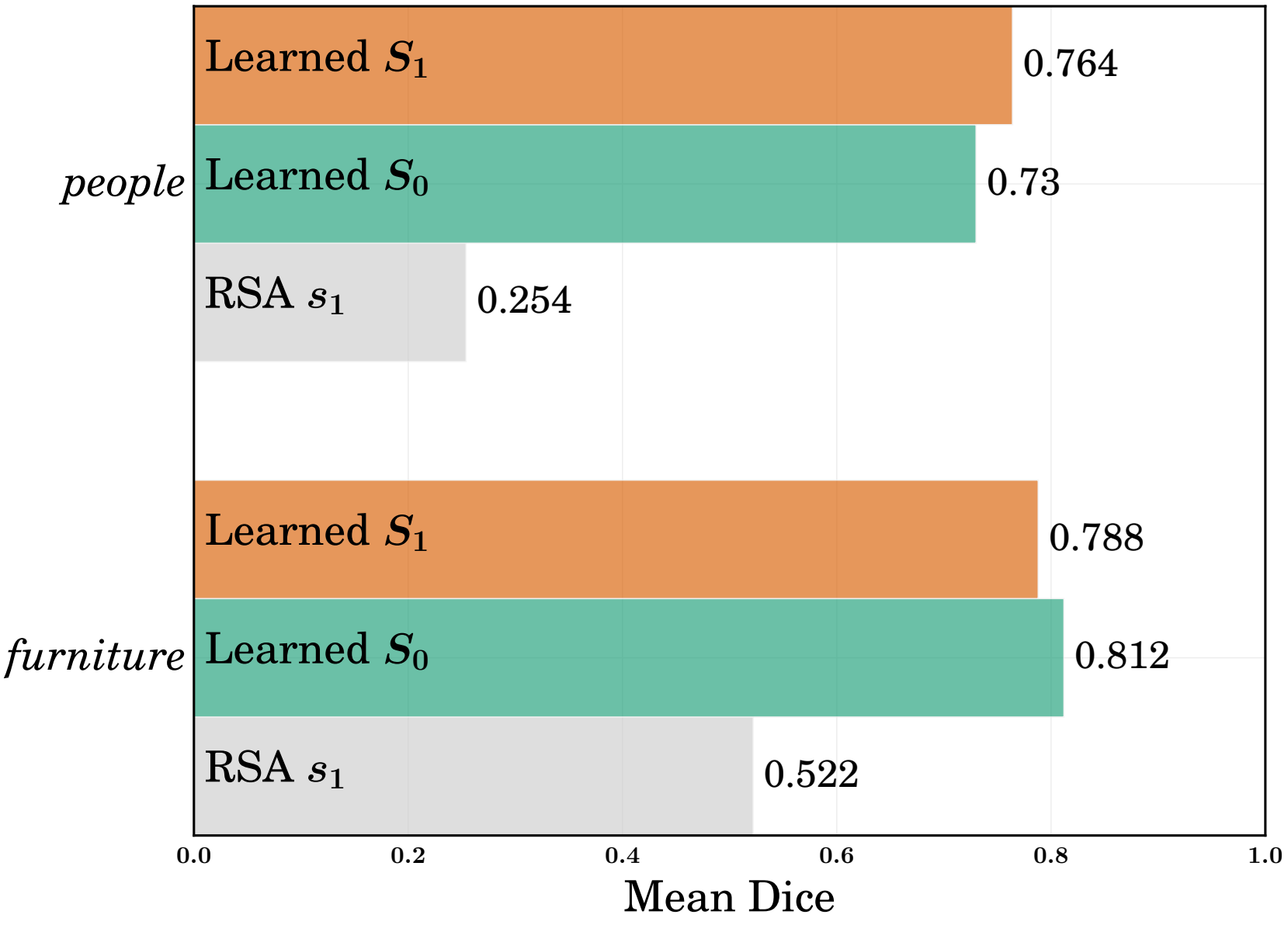
Results


people
RSA s_1
0.254
furniture
RSA s_1
0.522
0.0
0.2
0.4
0.6
0.8
1.0
Mean Dice

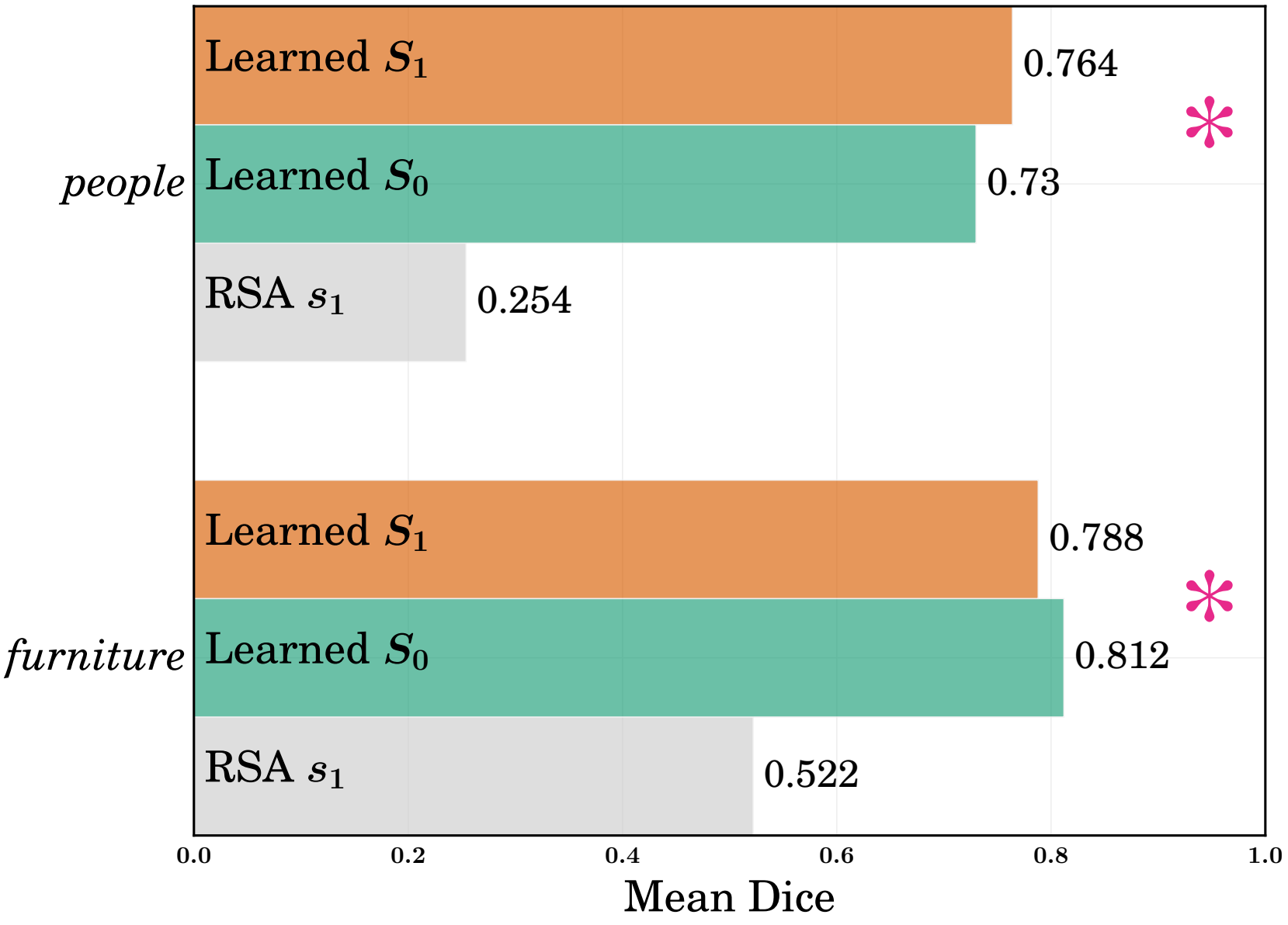
Results


people
Learned S_0
0.73
RSA s_1
0.254
furniture
Learned S_0
0.812
RSA s_1
0.522
0.0
0.2
0.4
0.6
0.8
1.0
Mean Dice

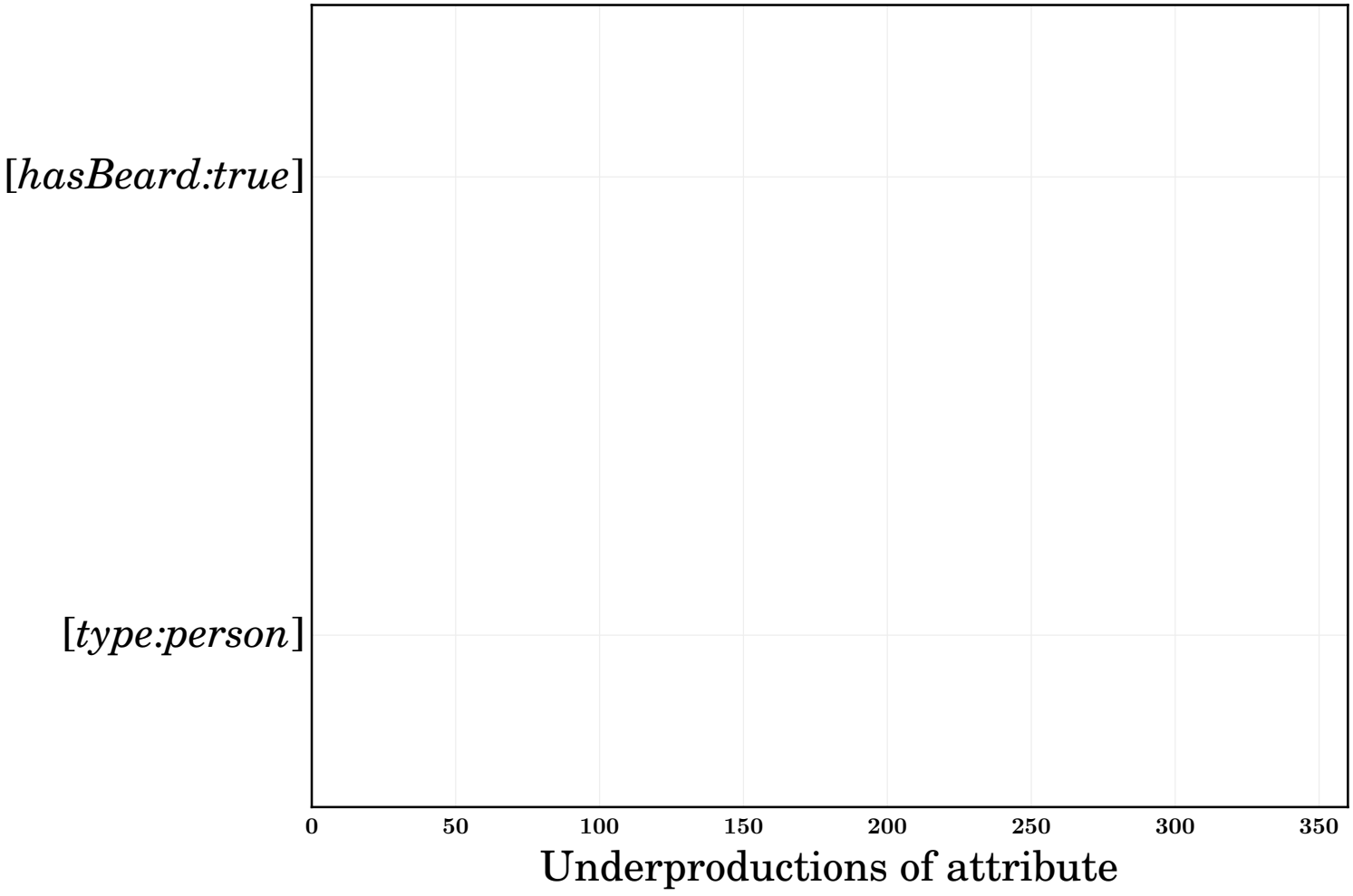
Results


Learned S_1
0.764
Learned S_0
0.73
RSA s_1
0.254
people
Learned S_1
0.788
Learned S_0
0.812
RSA s_1
0.522
furniture
0.0
0.2
0.4
0.6
0.8
1.0
Mean Dice

Results

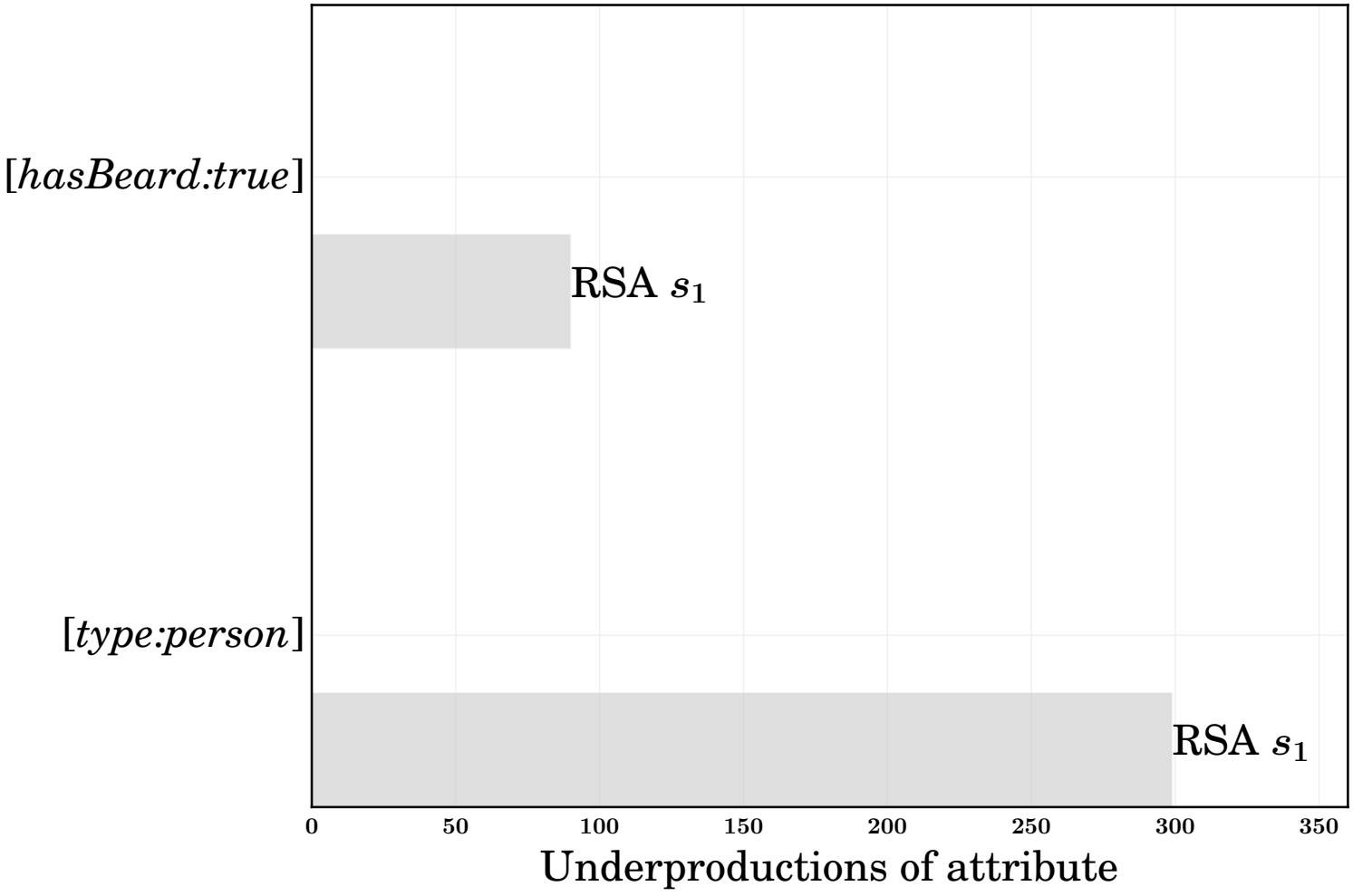

Learned S_1
0.764
Learned S_0
0.73
RSA s_1
0.254
people
Learned S_1
0.788
Learned S_0
0.812
RSA s_1
0.522
furniture
0.0
0.2
0.4
0.6
0.8
1.0
Mean Dice

Error analysis



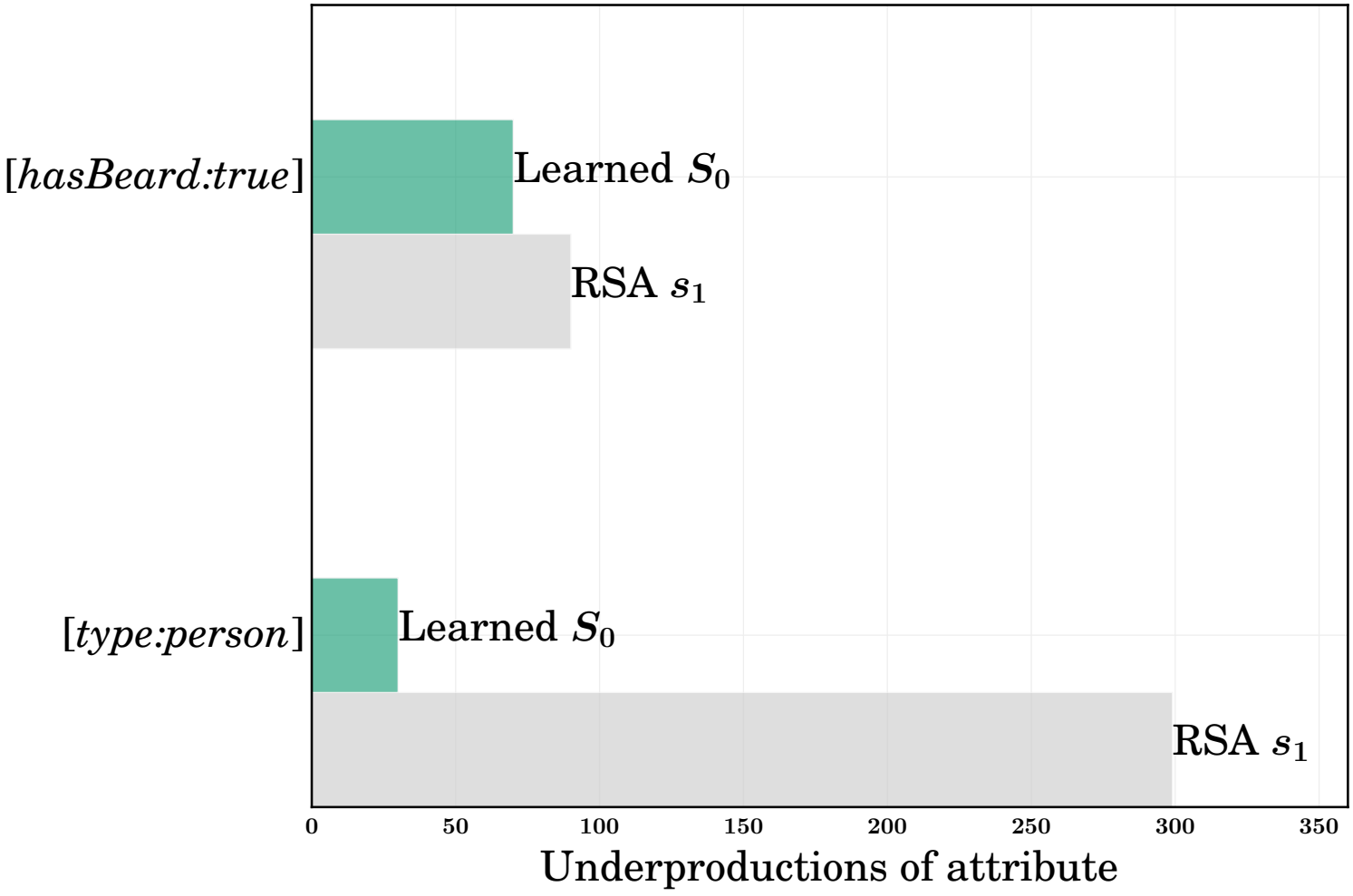
(Lower is better!)

Error analysis



(Lower is better!)

Error analysis



(Lower is better!)

Error analysis

(Lower is better!)

Summary and general lessons

Summary and general lessons

- Best model of production: established insights about natural language generation synthesized with RSA

Summary and general lessons

- Best model of production: established insights about natural language generation synthesized with RSA
- Cognitive and linguistic insights combined with learning

Summary and general lessons

- Best model of production: established insights about natural language generation synthesized with RSA
- Cognitive and linguistic insights combined with learning
- Learning a lexicon, speaker preferences, and context-dependent disambiguation in one process

Summary and general lessons

- Best model of production: established insights about natural language generation synthesized with RSA
- Cognitive and linguistic insights combined with learning
- Learning a lexicon, speaker preferences, and context-dependent disambiguation in one process
- Next: learning in extended versions of RSA

Summary and general lessons

- Best model of production: established insights about natural language generation synthesized with RSA
- Cognitive and linguistic insights combined with learning
- Learning a lexicon, speaker preferences, and context-dependent disambiguation in one process
- Next: learning in extended versions of RSA
- Next: increased scalability via variational inference

Summary and general lessons

- Best model of production: established insights about natural language generation synthesized with RSA
- Cognitive and linguistic insights combined with learning
- Learning a lexicon, speaker preferences, and context-dependent disambiguation in one process
- Next: learning in extended versions of RSA
- Next: increased scalability via variational inference

Thanks!