Semantic Analysis of Image-Based Learner Sentences

Levi King
Indiana University

August 6, 2021

Motivation

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;
- *"engineering first"*: no second language acquisition, pedagogy;

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;
- *"engineering first"*: no second language acquisition, pedagogy;

SLA research → communicative & task-based learning

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;
- *“engineering first”*: no second language acquisition, pedagogy;

SLA research $\rightarrow$ communicative & task-based learning

How can we bridge this gap?

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;
- *"engineering first"*: no second language acquisition, pedagogy;

SLA research → communicative & task-based learning

How can we bridge this gap?

- My vision:

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;
- *"engineering first"*: no second language acquisition, pedagogy;

SLA research $\rightarrow$ communicative & task-based learning

How can we bridge this gap?

- My vision: open source app; transparent; pipeline of existing tools;

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;
- *"engineering first"*: no second language acquisition, pedagogy;

SLA research $\rightarrow$ communicative & task-based learning

How can we bridge this gap?

- My vision: open source app; transparent; pipeline of existing tools;
- teachers create new games/stories by adding visual prompts and crowdsourcing native speaker (NS) responses;

Motivation

Most intelligent computer-assisted language learning (ICALL) applications (*Rosetta Stone*, *Duolingo*, etc.) rely on outdated, ineffective methods:

- rote memorization & grammatical error detection; menu-based vs. free input;
- *"engineering first"*: no second language acquisition, pedagogy;

SLA research → communicative & task-based learning

How can we bridge this gap?

- My vision: open source app; transparent; pipeline of existing tools;
- teachers create new games/stories by adding visual prompts and crowdsourcing native speaker (NS) responses;
- use NS model to evaluate non-native speaker (NNS) responses

Research Questions

Research Questions

RQ1. Are the picture description task (PDT) responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses?

Research Questions

RQ1. Are the picture description task (PDT) responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses?

RQ2. For PDT responses, what are appropriate representations for the purpose of providing meaning-oriented feedback or evaluation?

Research Questions

RQ1. Are the picture description task (PDT) responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses?

RQ2. For PDT responses, what are appropriate representations for the purpose of providing meaning-oriented feedback or evaluation?

RQ3. What kinds of NLP tools are appropriate here?

Research Questions

RQ1. Are the picture description task (PDT) responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses?

RQ2. For PDT responses, what are appropriate representations for the purpose of providing meaning-oriented feedback or evaluation?

RQ3. What kinds of NLP tools are appropriate here?

RQ4. How do "bag-of-words" and "bag-of-dependencies" approaches compare in terms of performance?

Research Questions

RQ1. Are the picture description task (PDT) responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses?

RQ2. For PDT responses, what are appropriate representations for the purpose of providing meaning-oriented feedback or evaluation?

RQ3. What kinds of NLP tools are appropriate here?

RQ4. How do "bag-of-words" and "bag-of-dependencies" approaches compare in terms of performance?

RQ5. Can the accuracy of the system be improved with information from semantic tools (e.g., BERT)?

Research Questions

RQ1. Are the picture description task (PDT) responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses?

RQ2. For PDT responses, what are appropriate representations for the purpose of providing meaning-oriented feedback or evaluation?

RQ3. What kinds of NLP tools are appropriate here?

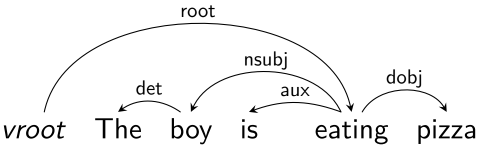
RQ4. How do “bag-of-words” and “bag-of-dependencies” approaches compare in terms of performance?

RQ5. Can the accuracy of the system be improved with information from semantic tools (e.g., BERT)?

RQ6. What is the annotation scheme for this task and can the system perform within the range of human performance?

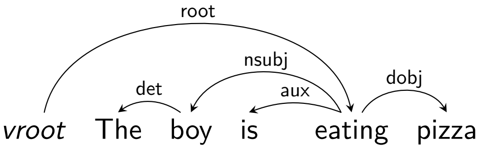
System

Step 1: Dependency parse:



System

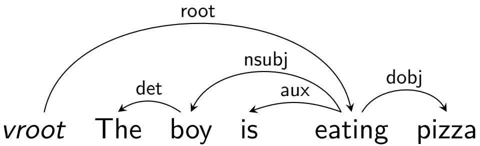
Step 1: Dependency parse:



Get dependencies:
root(eating, *vroot*)
det(the, boy)
nsubj(boy, eating)
aux(is, eating)
dobj(pizza, eating)

System

Step 1: Dependency parse:



Get dependencies:	**Step 2: Lemmatize:**
root(eating, *vroot*)	→ root(**eat**, *vroot*)
det(the, boy)	→ det(the, boy)
nsubj(boy, eating)	→ nsubj(boy, **eat**)
aux(is, eating)	→ aux(**be**, **eat**)
dobj(pizza, eating)	→ dobj(pizza, **eat**)

System

Step 3: tf-idf (term frequency-inverse document frequency)

System

Step 3: tf-idf (term frequency-inverse document frequency)

NS model: [He is eating pizza. The boy is eating pizza.]

System

Step 3: tf-idf (term frequency-inverse document frequency)

NS model: [He is eating pizza. The boy is eating pizza.]

NNS 1: He is eating food.

System

Step 3: tf-idf (term frequency-inverse document frequency)

NS model: [He is eating pizza. The boy is eating pizza.]

NNS 1: He is eating food. NNS 2: He is eating pizza.

System

Step 3: tf-idf (term frequency-inverse document frequency)

NS model: [He is eating pizza. The boy is eating pizza.]

NNS 1: He is eating food. NNS 2: He is eating pizza.

NS ∪ NNSs	NS model		NNS 1		NNS 2	
	tf	tf-idf	tf	tf-idf	tf	tf-idf
aux(be,eat)	2	.04	1	.02	1	.02
det(the,boy)	1	.04	-	0	-	0
dobj(food,eat)	-	0	1	.06	-	0
dobj(pizza,eat)	2	.16	-	0	1	.08
nsubj(boy,eat)	1	.08	-	0	-	0
nsubj(he,eat)	1	.04	1	.02	1	.02
root(eat,vroot)	2	.02	1	.01	1	.01

Response scores: cosine(NS model tf-idf vector, NNS tf-idf vector)

System

Step 3: tf-idf (term frequency-inverse document frequency)

NS model: [He is eating pizza. The boy is eating pizza.]

NNS 1: He is eating food. NNS 2: He is eating pizza.

NS ∪ NNSs	NS model		NNS 1		NNS 2	
	tf	tf-idf	tf	tf-idf	tf	tf-idf
aux(be,eat)	2	.04	1	.02	1	.02
det(the,boy)	1	.04	-	0	-	0
dobj(food,eat)	-	0	1	.06	-	0
dobj(pizza,eat)	2	.16	-	0	1	.08
nsubj(boy,eat)	1	.08	-	0	-	0
nsubj(he,eat)	1	.04	1	.02	1	.02
root(eat,vroot)	2	.02	1	.01	1	.01

Response scores: cosine(NS model tf-idf vector, NNS tf-idf vector)

NNS 1: 0.139;

System

Step 3: tf-idf (term frequency-inverse document frequency)

NS model: [He is eating pizza. The boy is eating pizza.]

NNS 1: He is eating food. NNS 2: He is eating pizza.

NS ∪ NNSs	NS model		NNS 1		NNS 2	
	tf	tf-idf	tf	tf-idf	tf	tf-idf
aux(be,eat)	2	.04	1	.02	1	.02
det(the,boy)	1	.04	-	0	-	0
dobj(food,eat)	-	0	1	.06	-	0
dobj(pizza,eat)	2	.16	-	0	1	.08
nsubj(boy,eat)	1	.08	-	0	-	0
nsubj(he,eat)	1	.04	1	.02	1	.02
root(eat,vroot)	2	.02	1	.01	1	.01

Response scores: cosine(NS model tf-idf vector, NNS tf-idf vector)

NNS 1: 0.139; NNS 2: 0.886

System

Step 3: tf-idf (term frequency-inverse document frequency)

NS model: [He is eating pizza. The boy is eating pizza.]

NNS 1: He is eating food. NNS 2: He is eating pizza.

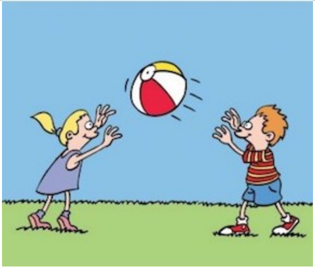
NS ∪ NNSs	NS model		NNS 1		NNS 2	
	tf	tf-idf	tf	tf-idf	tf	tf-idf
aux(be,eat)	2	.04	1	.02	1	.02
det(the,boy)	1	.04	-	0	-	0
dobj(food,eat)	-	0	1	.06	-	0
dobj(pizza,eat)	2	.16	-	0	1	.08
nsubj(boy,eat)	1	.08	-	0	-	0
nsubj(he,eat)	1	.04	1	.02	1	.02
root(eat,vroot)	2	.02	1	.01	1	.01

Response scores: cosine(NS model tf-idf vector, NNS tf-idf vector)

NNS 1: 0.139; NNS 2: 0.886 $\rightarrow$ *NNS 2 is closest to the model.*

Data collection

PDT with very simple images only:

	
Original image	Simplified PDT image

Intended to focus participants' attention on the main action

Data collection

Two PDT prompt versions:

Targeted	Untargeted
	
What is ***the baby*** *doing?*	*What is happening?*

Intended for exploring the specificity needed for my approach

Data collection

3 verb types:

10 **intransitive** items	10 **transitive** items	10 **ditransitive** items
		
What is the girl doing?	What is the boy doing?	What is the girl doing?

Intended for exploring whether my approach can generalize to a range of sentence types

Data collection

The pilot study *rake* problem; 100% of NS used the verb *rake*:

NNS Responses
The gardener is *cleaning* the street.
a man *removing* the tree leafs.
The man is *sweeping* the floor.
A man is *gathering* lots of leafs.

- NNS responses without *rake* are penalized;
- I address this by asking NSs for two non-identical responses.

Main study: Data collection

499 participants, 13,533 responses:

Main study: Data collection

499 participants, 13,533 responses:

- 141 NNSs (ELIP at IU), 4,290 responses;

Main study: Data collection

499 participants, 13,533 responses:

- 141 NNSs (ELIP at IU), 4,290 responses;
  - 125 Mandarin, 4 Korean, 3 Burmese, 2 Hindi; 1 each: Arabic, Indonesian, German, Gujarati, Spanish, Thai, Vietnamese;

Main study: Data collection

499 participants, 13,533 responses:

- 141 NNSs (ELIP at IU), 4,290 responses;
  - 125 Mandarin, 4 Korean, 3 Burmese, 2 Hindi; 1 each: Arabic, Indonesian, German, Gujarati, Spanish, Thai, Vietnamese;
- 358 NSs, 9,243 responses:

Main study: Data collection

499 participants, 13,533 responses:

- 141 NNSs (ELIP at IU), 4,290 responses;
  - 125 Mandarin, 4 Korean, 3 Burmese, 2 Hindi; 1 each: Arabic, Indonesian, German, Gujarati, Spanish, Thai, Vietnamese;
- 358 NSs, 9,243 responses:
  - 329 `crowdsourced`, purchased via SurveyMonkey;

Main study: Data collection

499 participants, 13,533 responses:

- 141 NNSs (ELIP at IU), 4,290 responses;
  - 125 Mandarin, 4 Korean, 3 Burmese, 2 Hindi; 1 each: Arabic, Indonesian, German, Gujarati, Spanish, Thai, Vietnamese;
- 358 NSs, 9,243 responses:
  - 329 `crowdsourced`, purchased via SurveyMonkey;
    - 7,960 responses;

Main study: Data collection

499 participants, 13,533 responses:

- 141 NNSs (ELIP at IU), 4,290 responses;
  - 125 Mandarin, 4 Korean, 3 Burmese, 2 Hindi; 1 each: Arabic, Indonesian, German, Gujarati, Spanish, Thai, Vietnamese;
- 358 NSs, 9,243 responses:
  - 329 `crowdsourced`, purchased via SurveyMonkey;
    - 7,960 responses;
  - 29 `familiar`, unpaid colleagues;
    - 1,283 responses;

Annotation features

5 binary features:

Annotation features

5 binary features:

- Core event:

Annotation features

5 binary features:

- Core event: Does response capture main action?

Annotation features

5 binary features:

- Core event: Does response capture main action?
- Answerhood:

Annotation features

5 binary features:

- CORE EVENT: Does response capture main action?
- ANSWERHOOD: Does response directly answer prompt?

Annotation features

5 binary features:

- Core event: Does response capture main action?
- Answerhood: Does response directly answer prompt?
- Grammaticality:

Annotation features

5 binary features:

- Core event: Does response capture main action?
- Answerhood: Does response directly answer prompt?
- Grammaticality: Is response free from grammar problems?

Annotation features

5 binary features:

- Core event: Does response capture main action?
- Answerhood: Does response directly answer prompt?
- Grammaticality: Is response free from grammar problems?
- Interpretability:

Annotation features

5 binary features:

- Core event: Does response capture main action?
- Answerhood: Does response directly answer prompt?
- Grammaticality: Is response free from grammar problems?
- Interpretability: Does response evoke a clear mental image?

Annotation features

5 binary features:

- CORE EVENT: Does response capture main action?
- ANSWERHOOD: Does response directly answer prompt?
- GRAMMATICALITY: Is response free from grammar problems?
- INTERPRETABILITY: Does response evoke a clear mental image?
- VERIFIABILITY:

Annotation features

5 binary features:

- CORE EVENT: Does response capture main action?
- ANSWERHOOD: Does response directly answer prompt?
- GRAMMATICALITY: Is response free from grammar problems?
- INTERPRETABILITY: Does response evoke a clear mental image?
- VERIFIABILITY: Is all response info supported by image?

Annotation features

Core event, **A**nswerhood, **G**rammaticality, **I**nterpretability, **V**erifiability

What is the boy doing?	C	A	G	I	V
He is eating food.	0	1	1	1	1
he eating pizza.	1	1	0	1	1
The boy is smiling pizza.	0	1	0	0	0
He may get fat eating.	0	0	1	1	0

Inter-rater reliability (Cohen's kappa): 0.744 (**I**) – 0.936 (**A**)

Evaluating performance

To evaluate system performance, I need **benchmark rankings** for the NNS test set.

Evaluating performance

To evaluate system performance, I need **benchmark rankings** for the NNS test set.

- Mean average precision (**MAP**) to see how system rankings predict **individual features**;
  - Also need MAP from benchmark rankings (*upper bound*);

Evaluating performance

To evaluate system performance, I need **benchmark rankings** for the NNS test set.

- Mean average precision (**MAP**) to see how system rankings predict **individual features**;
  - Also need MAP from benchmark rankings (*upper bound*);
- **Spearman** rank correlation: Compare system rankings with benchmark rankings to see how system predicts overall quality;

Evaluating performance

To evaluate system performance, I need **benchmark rankings** for the NNS test set.

- Mean average precision (**MAP**) to see how system rankings predict **individual features**;
  - Also need MAP from benchmark rankings (*upper bound*);
- **Spearman** rank correlation: Compare system rankings with benchmark rankings to see how system predicts overall quality;

How do we get benchmark rankings from 5 binary annotations?

Evaluating performance

To evaluate system performance, I need **benchmark rankings** for the NNS test set.

- Mean average precision (**MAP**) to see how system rankings predict **individual features**;
  - Also need MAP from benchmark rankings (*upper bound*);
- **Spearman** rank correlation: Compare system rankings with benchmark rankings to see how system predicts overall quality;

How do we get benchmark rankings from 5 binary annotations?

- Determine feature weights and apply to annotations to obtain benchmark holistic scores and then rankings.

Evaluating performance

To evaluate system performance, I need **benchmark rankings** for the NNS test set.

- Mean average precision (**MAP**) to see how system rankings predict **individual features**;
    - Also need MAP from benchmark rankings (*upper bound*);
- **Spearman** rank correlation: Compare system rankings with benchmark rankings to see how system predicts overall quality;

How do we get benchmark rankings from 5 binary annotations?

- Determine feature weights and apply to annotations to obtain benchmark holistic scores and then rankings.
- Annotators performed a preference task for pairs of responses.

Evaluating performance

To evaluate system performance, I need **benchmark rankings** for the NNS test set.

- Mean average precision (**MAP**) to see how system rankings predict **individual features**;
  - Also need MAP from benchmark rankings (*upper bound*);
- **Spearman** rank correlation: Compare system rankings with benchmark rankings to see how system predicts overall quality;

How do we get benchmark rankings from 5 binary annotations?

- Determine feature weights and apply to annotations to obtain benchmark holistic scores and then rankings.
- Annotators performed a preference task for pairs of responses.
- Feature weights were derived according to how frequently each feature is “yes” among preferred responses.

Benchmark rankings

Weighted annotation score (WAS); *weighted annotation ranking* (WAR)

What is happening?	C	A	G	I	V	WAS	WAR
The boy is eating pizza	.365	.093	.055	.224	.263	10	1
Child is eating pizza	.365	.093	0	.224	.263	.945	2
Tommy is eating pizza	.365	.093	.055	.224	0	.737	3
The boy's eating his favorite food	0	.093	.055	0	0	.513	4
Pizza is this boy's favorite food	0	0	.055	0	0	.055	5

Benchmark rankings

Weighted annotation score (WAS); *weighted annotation ranking* (WAR)

What is happening?	C	A	G	I	V	WAS	WAR
The boy is eating pizza	.365	.093	.055	.224	.263	10	1
Child is eating pizza	.365	.093	0	.224	.263	.945	2
Tommy is eating pizza	.365	.093	.055	.224	0	.737	3
The boy's eating his favorite food	0	.093	.055	0	0	.513	4
Pizza is this boy's favorite food	0	0	.055	0	0	.055	5

Agreement for two annotators on a sample of 300 pairs:

Chance Agree	Observed Agree	Cohen's Kappa
.621	.883 (265/300)	.692

SBERT for comparison

I also use SBERT for comparing my system's performance.

SBERT for comparison

I also use SBERT for comparing my system's performance.

- State-of-the-art sentence embedding for semantic textual similarity.

SBERT for comparison

I also use SBERT for comparing my system's performance.

- State-of-the-art sentence embedding for semantic textual similarity.
- Replaces dependency parser + lemmatizer + tf-idf cosine pipeline.

SBERT for comparison

I also use SBERT for comparing my system's performance.

- State-of-the-art sentence embedding for semantic textual similarity.
- Replaces dependency parser + lemmatizer + tf-idf cosine pipeline.
- Provides distance between NNS response and NS model; rankable.

SBERT for comparison

I also use SBERT for comparing my system's performance.

- State-of-the-art sentence embedding for semantic textual similarity.
- Replaces dependency parser + lemmatizer + tf-idf cosine pipeline.
- Provides distance between NNS response and NS model; rankable.
- Not explainable; Internal representations are not suitable for informing a feedback module.

SBERT for comparison

I also use SBERT for comparing my system's performance.

- State-of-the-art sentence embedding for semantic textual similarity.
- Replaces dependency parser + lemmatizer + tf-idf cosine pipeline.
- Provides distance between NNS response and NS model; rankable.
- Not explainable; Internal representations are not suitable for informing a feedback module.
    - Dependencies *are* suitable.

System configuration

Optimizing means finding the best system settings:

System configuration

Optimizing means finding the best system settings:

- **Transitivity**: `intransitive`, `transitive`, `ditransitive`;

System configuration

Optimizing means finding the best system settings:

- **Transitivity**: `intransitive`, `transitive`, `ditransitive`;
- **Targeting**: `targeted`, `untargeted`;

System configuration

Optimizing means finding the best system settings:

- **Transitivity**: `intransitive`, `transitive`, `ditransitive`;
- **Targeting**: `targeted`, `untargeted`;
- **Familiarity**: `familiar`, `crowdsourced`;

System configuration

Optimizing means finding the best system settings:

- **Transitivity**: `intransitive`, `transitive`, `ditransitive`;
- **Targeting**: `targeted`, `untargeted`;
- **Familiarity**: `familiar`, `crowdsourced`;
- **Primacy**:
  - `primary`: NS model contains only 1st responses;
  - `mixed`: NS model: 1st & 2nd responses (50-50);

System configuration

Optimizing means finding the best system settings:

- **Transitivity**: `intransitive`, `transitive`, `ditransitive`;
- **Targeting**: `targeted`, `untargeted`;
- **Familiarity**: `familiar`, `crowdsourced`;
- **Primacy**:
  - `primary`: NS model contains only 1st responses;
  - `mixed`: NS model: 1st & 2nd responses (50-50);
- **Term Representation**:
  - `ldh`: label-dependent-head; i.e., labeled dependencies;
  - `xdh`: dependent-head; i.e., unlabeled dependencies;
  - `xdx`: dependent only; cf. *bag of words*;
  - Does not apply to SBERT (operates on plain text);

System configuration

Optimizing means finding the best system settings:

- **Transitivity**: `intransitive`, `transitive`, `ditransitive`;
- **Targeting**: `targeted`, `untargeted`;
- **Familiarity**: `familiar`, `crowdsourced`;
- **Primacy**:
  - `primary`: NS model contains only 1st responses;
  - `mixed`: NS model: 1st & 2nd responses (50-50);
- **Term Representation**:
  - `ldh`: label-dependent-head; i.e., labeled dependencies;
  - `xdh`: dependent-head; i.e., unlabeled dependencies;
  - `xdx`: dependent only; cf. *bag of words*;
  - Does not apply to SBERT (operates on plain text);

A **system configuration** combines one setting from each.

Sampling data

Sampling data

NNS test sets:

- All experiments rank the same randomly sampled NNS test sets;
- 70 `targeted`, 70 `untargeted` per PDT item (max available for NNS data);

Sampling data

NNS test sets:

- All experiments rank the same randomly sampled NNS test sets;
- 70 `targeted`, 70 `untargeted` per PDT item (max available for NNS data);

NS models:

- 14-response models (max available for `familiar` data);
- 50-response models (max available for `crowdsourced` data);

Sampling data: Complexity

Standardized type-to-token ratio (STTR) for response samples. Tokens here are *dependencies*.

	n14		n50	n70
	Fam	Crd	Crd	NNS
Intrans	.558	.525	.535	.391
Trans	.569	.580	.581	.517
Ditrans	.598	.640	.637	.606
Target	.545	.535	.545	.481
Untarg	.610	.633	.621	.528
Primary	N/A	.517	.523	.505
Mixed	.576	.652	.645	N/A
xdx	.364	.424	.421	.364
xdh	.658	.661	.660	.572
ldh	.665	.664	.671	.578
Total	.576	.583	.584	.505

Sampling data: Complexity

Standardized type-to-token ratio (STTR) for response samples. Tokens here are *dependencies*.

	n14		n50	n70
	Fam	Crd	Crd	NNS
Intrans	.558	.525	.535	.391
Trans	.569	.580	.581	.517
Ditrans	.598	.640	.637	.606
Target	.545	.535	.545	.481
Untarg	.610	.633	.621	.528
Primary	N/A	.517	.523	.505
Mixed	.576	.652	.645	N/A
xdx	.364	.424	.421	.364
xdh	.658	.661	.660	.572
ldh	.665	.664	.671	.578
Total	.576	.583	.584	.505

Complexity often correlates with parameter settings in terms of system performance.

Sampling data: Complexity

Standardized type-to-token ratio (STTR) for response samples. Tokens here are *dependencies*.

	n14		n50	n70
	Fam	Crd	Crd	NNS
Intrans	.558	.525	.535	.391
Trans	.569	.580	.581	.517
Ditrans	.598	.640	.637	.606
Target	.545	.535	.545	.481
Untarg	.610	.633	.621	.528
Primary	N/A	.517	.523	.505
Mixed	.576	.652	.645	N/A
xdx	.364	.424	.421	.364
xdh	.658	.661	.660	.572
ldh	.665	.664	.671	.578
Total	.576	.583	.584	.505

Complexity often correlates with parameter settings in terms of system performance.

Within each parameter block, complexity increases as we move down the rows. E.g.:

`Intrans` < `Trans` < `Ditrans`

Sampling data: Complexity

Standardized type-to-token ratio (STTR) for response samples. Tokens here are *dependencies*.

	n14		n50	n70
	Fam	Crd	Crd	NNS
Intrans	.558	.525	.535	.391
Trans	.569	.580	.581	.517
Ditrans	.598	.640	.637	.606
Target	.545	.535	.545	.481
Untarg	.610	.633	.621	.528
Primary	N/A	.517	.523	.505
Mixed	.576	.652	.645	N/A
xdx	.364	.424	.421	.364
xdh	.658	.661	.660	.572
ldh	.665	.664	.671	.578
Total	.576	.583	.584	.505

Complexity often correlates with parameter settings in terms of system performance.

Within each parameter block, complexity increases as we move down the rows. E.g.:

`Intrans` < `Trans` < `Ditrans`

In some settings (e.g., `Intrans`), `Crowd` complexity is closer to NNS than is `Familiar`; other settings vice versa (e.g., `Ditrans`).

Predicting features: CORE EVENT MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	***.859***	.856	.854	.865	.835	**.855**	.854	.852	.865	.831
Tran	***.737***	.735	.728	.742	.703	**.736**	.733	.725	.742	.701
Ditr	***.665***	.661	.664	.660	.634	.657	.656	**.661**	.660	.629
Targ	***.739***	.738	.732	.735	.708	**.737**	.735	.729	.735	.704
Untg	***.768***	.763	.765	.777	.740	.762	.759	**.763**	.777	.736
Prim	***.754***	.752	.747	.756	.723	**.750**	.748	.745	.756	.719
Mix	***.753***	.749	.750	.756	.725	**.749**	.746	.746	.756	.721
Total	***.735***	.751	.748	.756	.724	**.750**	.747	.746	.756	.720

Predicting features: Core event MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	***.859***	.856	.854	.865	.835	**.855**	.854	.852	.865	.831
Tran	***.737***	.735	.728	.742	.703	**.736**	.733	.725	.742	.701
Ditr	***.665***	.661	.664	.660	.634	.657	.656	**.661**	.660	.629
Targ	***.739***	.738	.732	.735	.708	**.737**	.735	.729	.735	.704
Untg	***.768***	.763	.765	.777	.740	.762	.759	**.763**	.777	.736
Prim	***.754***	.752	.747	.756	.723	**.750**	.748	.745	.756	.719
Mix	***.753***	.749	.750	.756	.725	**.749**	.746	.746	.756	.721
Total	***.735***	.751	.748	.756	.724	**.750**	.747	.746	.756	.720

- In all cases, ldh + 14NS is best (slightly);

Predicting features: CORE EVENT MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	***.859***	.856	.854	.865	.835	**.855**	.854	.852	.865	.831
Tran	***.737***	.735	.728	.742	.703	**.736**	.733	.725	.742	.701
Ditr	***.665***	.661	.664	.660	.634	.657	.656	**.661**	.660	.629
Targ	***.739***	.738	.732	.735	.708	**.737**	.735	.729	.735	.704
Untg	***.768***	.763	.765	.777	.740	.762	.759	**.763**	.777	.736
Prim	***.754***	.752	.747	.756	.723	**.750**	.748	.745	.756	.719
Mix	***.753***	.749	.750	.756	.725	**.749**	.746	.746	.756	.721
Total	***.735***	.751	.748	.756	.724	**.750**	.747	.746	.756	.720

- In all cases, ldh + 14NS is best (slightly);
- xdx becomes more competitive for larger model (50NS);

Predicting features: Core event MAP

	Crowd NS model = 14					Crowd NS model = 50				
	`ldh`	`xdh`	`xdx`	WAR	SBERT	`ldh`	`xdh`	`xdx`	WAR	SBERT
`Intr`	***.859***	.856	.854	.865	.835	**.855**	.854	.852	.865	.831
`Tran`	***.737***	.735	.728	.742	.703	**.736**	.733	.725	.742	.701
`Ditr`	***.665***	.661	.664	.660	.634	.657	.656	**.661**	.660	.629
`Targ`	***.739***	.738	.732	.735	.708	**.737**	.735	.729	.735	.704
`Untg`	***.768***	.763	.765	.777	.740	.762	.759	**.763**	.777	.736
`Prim`	***.754***	.752	.747	.756	.723	**.750**	.748	.745	.756	.719
`Mix`	***.753***	.749	.750	.756	.725	**.749**	.746	.746	.756	.721
Total	***.735***	.751	.748	.756	.724	**.750**	.747	.746	.756	.720

- In all cases, `ldh` + 14NS is best (slightly);
- `xdx` becomes more competitive for larger model (50NS);
  - `ditrans`, `untarg`: *most complex*—i.e., highest STTRs;

Predicting features: CORE EVENT MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	***.859***	.856	.854	.865	.835	**.855**	.854	.852	.865	.831
Tran	***.737***	.735	.728	.742	.703	**.736**	.733	.725	.742	.701
Ditr	***.665***	.661	.664	.660	.634	.657	.656	**.661**	.660	.629
Targ	***.739***	.738	.732	.735	.708	**.737**	.735	.729	.735	.704
Untg	***.768***	.763	.765	.777	.740	.762	.759	**.763**	.777	.736
Prim	***.754***	.752	.747	.756	.723	**.750**	.748	.745	.756	.719
Mix	***.753***	.749	.750	.756	.725	**.749**	.746	.746	.756	.721
Total	***.735***	.751	.748	.756	.724	**.750**	.747	.746	.756	.720

- In all cases, `ldh` + 14NS is best (slightly);
- `xdx` becomes more competitive for larger model (50NS);
  - `ditrans`, `untarg`: *most complex*—i.e., highest STTRs;
  - In general: `ldh` STTR > `xdh` STTR > `xdx` STTR

Predicting features: Core event MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.859	.859	***.865***	.865	.838	**.857**	.852	.848	.865	.833
Tran	***.740***	.737	.726	.742	.703	**.738**	.735	.728	.742	.702
Ditr	.651	.648	**.660**	.660	.625	.663	.659	***.673***	.660	.641
Targ	**.733**	.732	.732	.735	.707	***.739***	.736	.733	.735	.709
Untg	.767	.764	***.769***	.777	.737	.767	.761	**.767**	.777	.742
Total	.750	.748	**.751**	.756	.722	***.753***	.749	.750	.756	.725

Predicting features: CORE EVENT MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.859	.859	***.865***	.865	.838	**.857**	.852	.848	.865	.833
Tran	***.740***	.737	.726	.742	.703	**.738**	.735	.728	.742	.702
Ditr	.651	.648	**.660**	.660	.625	.663	.659	***.673***	.660	.641
Targ	**.733**	.732	.732	.735	.707	***.739***	.736	.733	.735	.709
Untg	.767	.764	***.769***	.777	.737	.767	.761	**.767**	.777	.742
Total	.750	.748	**.751**	.756	.722	***.753***	.749	.750	.756	.725

- *mixed only (due to sparse familiar data);

Predicting features: Core event MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.859	.859	***.865***	.865	.838	**.857**	.852	.848	.865	.833
Tran	***.740***	.737	.726	.742	.703	**.738**	.735	.728	.742	.702
Ditr	.651	.648	**.660**	.660	.625	.663	.659	***.673***	.660	.641
Targ	**.733**	.732	.732	.735	.707	***.739***	.736	.733	.735	.709
Untg	.767	.764	***.769***	.777	.737	.767	.761	**.767**	.777	.742
Total	.750	.748	**.751**	.756	.722	***.753***	.749	.750	.756	.725

- *mixed only (due to sparse familiar data);
- Totals: crowdsourced outperforms familiar (slightly);

Predicting features: Core event MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.859	.859	***.865***	.865	.838	**.857**	.852	.848	.865	.833
Tran	***.740***	.737	.726	.742	.703	**.738**	.735	.728	.742	.702
Ditr	.651	.648	**.660**	.660	.625	.663	.659	***.673***	.660	.641
Targ	**.733**	.732	.732	.735	.707	***.739***	.736	.733	.735	.709
Untg	.767	.764	***.769***	.777	.737	.767	.761	**.767**	.777	.742
Total	.750	.748	**.751**	.756	.722	***.753***	.749	.750	.756	.725

- *mixed only (due to sparse familiar data);
- Totals: crowdsourced outperforms familiar (slightly);
- crowdsourced works best with ldh;

Predicting features: Core event MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.859	.859	***.865***	.865	.838	**.857**	.852	.848	.865	.833
Tran	***.740***	.737	.726	.742	.703	**.738**	.735	.728	.742	.702
Ditr	.651	.648	**.660**	.660	.625	.663	.659	***.673***	.660	.641
Targ	**.733**	.732	.732	.735	.707	***.739***	.736	.733	.735	.709
Untg	.767	.764	***.769***	.777	.737	.767	.761	**.767**	.777	.742
Total	.750	.748	**.751**	.756	.722	***.753***	.749	.750	.756	.725

- *mixed only (due to sparse familiar data);
- Totals: crowdsourced outperforms familiar (slightly);
- crowdsourced works best with ldh;
- familiar works best with xdx;

Predicting features: MAP Results

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

- `xdx` > `xdh` > `ldh`;

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

- `xdx` $>$ `xdh` $>$ `ldh`;
- Model size makes no difference;

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

- `xdx` > `xdh` > `ldh`;
- Model size makes no difference;
- `familiar` > `crowdsourced`;

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

- `xdx` > `xdh` > `ldh`;
- Model size makes no difference;
- `familiar` > `crowdsourced`;

GRAMMATICALITY, in *most* cases:

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

- `xdx` > `xdh` > `ldh`;
- Model size makes no difference;
- `familiar` > `crowdsourced`;

GRAMMATICALITY, in *most* cases:

- `xdx` > `xdh` > `ldh`;

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

- xdx > xdh > ldh;
- Model size makes no difference;
- familiar > crowdsourced;

GRAMMATICALITY, in *most* cases:

- xdx > xdh > ldh;
- familiar 14NS > crowd 14NS > crowd 50NS;

Predicting features: MAP Results

For all 5 features, my system outperforms SBERT.

ANSWERHOOD, in *all* cases:

- `xdx` > `xdh` > `ldh`;
- Model size makes no difference;
- `familiar` > `crowdsourced`;

GRAMMATICALITY, in *most* cases:

- `xdx` > `xdh` > `ldh`;
- `familiar` 14NS > `crowd` 14NS > `crowd` 50NS;

Predicting ANSWERHOOD or GRAMMATICALITY is relatively simple; requires only small model and bag-of-words representation.

Predicting features: MAP Results

Predicting features: MAP Results

INTERPRETABILITY:

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` > 14NS `familiar` > 50NS `crowd`;

Predicting features: MAP Results

Interpretability:

- 14NS `crowd` > 14NS `familiar` > 50NS `crowd`;

Verifiability:

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` $>$ 14NS `familiar` $>$ 50NS `crowd`;

VERIFIABILITY:

- 14NS `crowd` $>$ 50NS `crowd` $>$ 14NS `familiar`;

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` > 14NS `familiar` > 50NS `crowd`;

VERIFIABILITY:

- 14NS `crowd` > 50NS `crowd` > 14NS `familiar`;
- Model size effect is most pronounced with `untargeted` & `mixed`;

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` > 14NS `familiar` > 50NS `crowd`;

VERIFIABILITY:

- 14NS `crowd` > 50NS `crowd` > 14NS `familiar`;
- Model size effect is most pronounced with `untargeted` & `mixed`;
    - Unconstrained settings; larger models have more noise;

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` $>$ 14NS `familiar` $>$ 50NS `crowd`;

VERIFIABILITY:

- 14NS `crowd` $>$ 50NS `crowd` $>$ 14NS `familiar`;
- Model size effect is most pronounced with `untargeted` & `mixed`;
  - Unconstrained settings; larger models have more noise;

For both INTERPRETABILITY & VERIFIABILITY:

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` > 14NS `familiar` > 50NS `crowd`;

VERIFIABILITY:

- 14NS `crowd` > 50NS `crowd` > 14NS `familiar`;
- Model size effect is most pronounced with `untargeted` & `mixed`;
    - Unconstrained settings; larger models have more noise;

For both INTERPRETABILITY & VERIFIABILITY:

- `intransitives` & `ditransitives` work best with `xdx`;

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` > 14NS `familiar` > 50NS `crowd`;

VERIFIABILITY:

- 14NS `crowd` > 50NS `crowd` > 14NS `familiar`;
- Model size effect is most pronounced with `untargeted` & `mixed`;
  - Unconstrained settings; larger models have more noise;

For both INTERPRETABILITY & VERIFIABILITY:

- `intransitives` & `ditransitives` work best with `xdx`;
- `transitives` work best with `ldh`;

Predicting features: MAP Results

INTERPRETABILITY:

- 14NS `crowd` > 14NS `familiar` > 50NS `crowd`;

VERIFIABILITY:

- 14NS `crowd` > 50NS `crowd` > 14NS `familiar`;
- Model size effect is most pronounced with `untargeted` & `mixed`;
  - Unconstrained settings; larger models have more noise;

For both INTERPRETABILITY & VERIFIABILITY:

- `intransitives` & `ditransitives` work best with `xdx`;
- `transitives` work best with `ldh`;
  - Why? `Transitive` responses are relatively homogenous; Annotators relatively strict;

Predicting quality

Predicting quality

Holistic quality experiments use one set of 360 Spearman correlations:

targeting (2) $\times$ primacy (2) $\times$ term rep (3) $\times$ items (30) $=$ 360.

Predicting quality

Holistic quality experiments use one set of 360 Spearman correlations:

targeting (2) $\times$ primacy (2) $\times$ term rep (3) $\times$ items (30) $=$ 360.

(`Familiar` vs. `Crowd` handled separately due to sparse data.)

Predicting quality

Holistic quality experiments use one set of 360 Spearman correlations:

targeting (2) $\times$ primacy (2) $\times$ term rep (3) $\times$ items (30) $=$ 360.

(`Familiar` vs. `Crowd` handled separately due to sparse data.)

Each experiment focuses on one variable, e.g., targeting:

Divide 360 into 180 `targeted` scores and 180 `untargeted` scores; compare mean, median, etc.

Predicting quality

Holistic quality experiments use one set of 360 Spearman correlations:

targeting (2) $\times$ primacy (2) $\times$ term rep (3) $\times$ items (30) $=$ 360.

(`Familiar` vs. `Crowd` handled separately due to sparse data.)

Each experiment focuses on one variable, e.g., targeting:

Divide 360 into 180 `targeted` scores and 180 `untargeted` scores; compare mean, median, etc.

SBERT uses plain text (no term rep), thus only 120 total.

Predicting quality

Holistic quality experiments use one set of 360 Spearman correlations:

targeting (2) $\times$ primacy (2) $\times$ term rep (3) $\times$ items (30) $=$ 360.

(`Familiar` vs. `Crowd` handled separately due to sparse data.)

Each experiment focuses on one variable, e.g., targeting:

Divide 360 into 180 `targeted` scores and 180 `untargeted` scores; compare mean, median, etc.

SBERT uses plain text (no term rep), thus only 120 total.

(SBERT always wins over system.)

Predicting quality: `Transitivity`

Spearman rank correlations: System vs. WAR (benchmark)

		`intrans`		`trans`		`ditrans`	
		Sys	SBERT	Sys	SBERT	Sys	SBERT
	count	120	40	120	40	120	40
14NS	mean	***.439***	.497	.314	.563	.267	.400
	median	**.416**	.479	.304	.555	.276	.444
50NS	mean	**.423**	.516	*.345*	.566	*.278*	.446
	median	***.426***	.517	*.331*	.561	*.286*	.471

Predicting quality: `Transitivity`

Spearman rank correlations: System vs. WAR (benchmark)

		intrans		trans		ditrans	
		Sys	SBERT	Sys	SBERT	Sys	SBERT
	count	120	40	120	40	120	40
14NS	mean	***.439***	.497	.314	.563	.267	.400
14NS	median	**.416**	.479	.304	.555	.276	.444
50NS	mean	**.423**	.516	*.345*	.566	*.278*	.446
50NS	median	***.426***	.517	*.331*	.561	*.286*	.471

- SBERT, regardless of model size: `trans` > `intrans` > `ditrans`;

Predicting quality: `Transitivity`

Spearman rank correlations: System vs. WAR (benchmark)

		intrans		trans		ditrans	
		Sys	SBERT	Sys	SBERT	Sys	SBERT
	count	120	40	120	40	120	40
14NS	mean	***.439***	.497	.314	.563	.267	.400
	median	**.416**	.479	.304	.555	.276	.444
50NS	mean	**.423**	.516	*.345*	.566	*.278*	.446
	median	***.426***	.517	*.331*	.561	*.286*	.471

- SBERT, regardless of model size: `trans` > `intrans` > `ditrans`;
- System, regardless of model size: `intrans` > `trans` > `ditrans`;

Predicting quality: `Transitivity`

Spearman rank correlations: System vs. WAR (benchmark)

		intrans		trans		ditrans	
		Sys	SBERT	Sys	SBERT	Sys	SBERT
	count	120	40	120	40	120	40
14NS	mean	***.439***	.497	.314	.563	.267	.400
14NS	median	**.416**	.479	.304	.555	.276	.444
50NS	mean	**.423**	.516	*.345*	.566	*.278*	.446
50NS	median	***.426***	.517	*.331*	.561	*.286*	.471

- SBERT, regardless of model size: `trans` > `intrans` > `ditrans`;
- System, regardless of model size: `intrans` > `trans` > `ditrans`;
- More complex items (TTR) work best with larger models;

Predicting quality: `Transitivity`

Spearman rank correlations: System vs. WAR (benchmark)

		`intrans`		`trans`		`ditrans`	
		Sys	SBERT	Sys	SBERT	Sys	SBERT
	count	120	40	120	40	120	40
14NS	mean	***.439***	.497	.314	.563	.267	.400
	median	**.416**	.479	.304	.555	.276	.444
50NS	mean	**.423**	.516	*.345*	.566	*.278*	.446
	median	***.426***	.517	*.331*	.561	*.286*	.471

- SBERT, regardless of model size: `trans` > `intrans` > `ditrans`;
- System, regardless of model size: `intrans` > `trans` > `ditrans`;
- More complex items (TTR) work best with larger models;
  - `trans` & `ditrans`: 50NS model is best;

Predicting quality: `Transitivity`

Spearman rank correlations: System vs. WAR (benchmark)

		intrans		trans		ditrans	
		Sys	SBERT	Sys	SBERT	Sys	SBERT
	count	120	40	120	40	120	40
14NS	mean	***.439***	.497	.314	.563	.267	.400
14NS	median	**.416**	.479	.304	.555	.276	.444
50NS	mean	**.423**	.516	*.345*	.566	*.278*	.446
50NS	median	***.426***	.517	*.331*	.561	*.286*	.471

- SBERT, regardless of model size: `trans` > `intrans` > `ditrans`;
- System, regardless of model size: `intrans` > `trans` > `ditrans`;
- More complex items (TTR) work best with larger models;
  - `trans` & `ditrans`: 50NS model is best;
  - `intrans`: 14NS gives best mean, 50NS gives best median;

Predicting quality: Results

Predicting quality: Results

Targeting:

Predicting quality: Results

Targeting:

- `targeted` $>$ `untargeted`

Predicting quality: Results

Targeting:

- `targeted` $>$ `untargeted`
- 50NS models $>$ 14NS models

Predicting quality: Results

Targeting:

- `targeted` > `untargeted`
- 50NS models > 14NS models
  - Model size effect is most pronounced for `targeted`

Predicting quality: Results

Targeting:

- `targeted` > `untargeted`
- 50NS models > 14NS models
  - Model size effect is most pronounced for `targeted`

Familiarity (14NS models only):

Predicting quality: Results

Targeting:

- `targeted` > `untargeted`
- 50NS models > 14NS models
    - Model size effect is most pronounced for `targeted`

Familiarity (14NS models only):

- System: No discernible difference for `familiar` vs `crowdsourced`

Predicting quality: Results

Targeting:

- `targeted` > `untargeted`
- 50NS models > 14NS models
  - Model size effect is most pronounced for `targeted`

Familiarity (14NS models only):

- System: No discernible difference for `familiar` vs `crowdsourced`
- SBERT: `familiar` > `crowdsourced`

Predicting quality: Results

Targeting:

- `targeted` > `untargeted`
- 50NS models > 14NS models
  - Model size effect is most pronounced for `targeted`

Familiarity (14NS models only):

- System: No discernible difference for `familiar` vs `crowdsourced`
- SBERT: `familiar` > `crowdsourced`
  - NNS STTR < `familiar` STTR < `crowdsourced` STTR

Predicting quality: Results

Predicting quality: Results

Primacy:

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`
- System & SBERT: 50NS $>$ 14NS

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`
- System & SBERT: 50NS $>$ 14NS
  - System: model size effect is greatest for `primary`

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`
- System & SBERT: 50NS $>$ 14NS
  - System: model size effect is greatest for `primary`

Term representation:

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`
- System & SBERT: 50NS $>$ 14NS
  - System: model size effect is greatest for `primary`

Term representation:

- SBERT: 50NS $>$ 14NS

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`
- System & SBERT: 50NS $>$ 14NS
  - System: model size effect is greatest for `primary`

Term representation:

- SBERT: 50NS $>$ 14NS
- System: for `ldh` & `xdh`: 50NS $>$ 14NS;

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`
- System & SBERT: 50NS $>$ 14NS
  - System: model size effect is greatest for `primary`

Term representation:

- SBERT: 50NS $>$ 14NS
- System: for `ldh` & `xdh`: 50NS $>$ 14NS;
  - Model size effect is greater for `ldh`

Predicting quality: Results

Primacy:

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`
- System & SBERT: 50NS $>$ 14NS
  - System: model size effect is greatest for `primary`

Term representation:

- SBERT: 50NS $>$ 14NS
- System: for `ldh` & `xdh`: 50NS $>$ 14NS;
  - Model size effect is greater for `ldh`
- System: for `xdx`: NS14 $>$ NS50 (very slight)

Summary

- Collected 13,533 PDT responses from 499 participants;

Summary

- Collected 13,533 PDT responses from 499 participants;
- Annotated for 5 features, focused on content;

Summary

- Collected 13,533 PDT responses from 499 participants;
- Annotated for 5 features, focused on content;
- Established feature weights and benchmark rankings;

Summary

- Collected 13,533 PDT responses from 499 participants;
- Annotated for 5 features, focused on content;
- Established feature weights and benchmark rankings;
  - Features and weights are reliable;

Summary

- Collected 13,533 PDT responses from 499 participants;
- Annotated for 5 features, focused on content;
- Established feature weights and benchmark rankings;
  - Features and weights are reliable;
- Developed explainable semantic textual similarity system based on dependencies and tf-idf;

Summary

- Collected 13,533 PDT responses from 499 participants;
- Annotated for 5 features, focused on content;
- Established feature weights and benchmark rankings;
  - Features and weights are reliable;
- Developed explainable semantic textual similarity system based on dependencies and tf-idf;
- Uncovered some exploitable patterns for predicting features and holistic quality;

Future work

With more data, I would:

Future work

With more data, I would:

- Explore results for broader range of L1s;

Future work

With more data, I would:

- Explore results for broader range of L1s;
- Compare results across L2 English proficiency levels;

Future work

With more data, I would:

- Explore results for broader range of L1s;
- Compare results across L2 English proficiency levels;
- Further map relationship between complexity and optimal settings;

Future work

With more data, I would:

- Explore results for broader range of L1s;
- Compare results across L2 English proficiency levels;
- Further map relationship between complexity and optimal settings;
- For a given PDT item, try clustering responses into multiple models;

Future work

With more data, I would:

- Explore results for broader range of L1s;
- Compare results across L2 English proficiency levels;
- Further map relationship between complexity and optimal settings;
- For a given PDT item, try clustering responses into multiple models;
  - Route NNS responses to most appropriate model;

Backup slides

(The following slides are all backup for Q&A.)

Research Questions (full)

Research Questions (full)

RQ1. Are the responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses? In other words, do learners demonstrate significant overlap with native-like usage in a picture description task (PDT) setting?

Research Questions (full)

RQ1. Are the responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses? In other words, do learners demonstrate significant overlap with native-like usage in a picture description task (PDT) setting?

RQ2. In the constrained communicative environment of a PDT, what are appropriate response and model representations for the purpose of providing meaning-oriented feedback or evaluation? In other words, which linguistic components are crucial and which are superfluous?

Research Questions (full)

RQ1. Are the responses of L2 English learners sufficiently similar to those of NSs to allow automatic evaluation based on a collection of NS responses? In other words, do learners demonstrate significant overlap with native-like usage in a picture description task (PDT) setting?

RQ2. In the constrained communicative environment of a PDT, what are appropriate response and model representations for the purpose of providing meaning-oriented feedback or evaluation? In other words, which linguistic components are crucial and which are superfluous?

RQ3. What kinds of existing NLP tools and language resources can be integrated to form a content analysis system for open response language learning tasks?

Research Questions (full)

Research Questions (full)

RQ4. How do “bag-of-words” and “bag-of-dependencies” approaches compare in terms of performance? Is a bag-of-words approach alone adequate for our needs?

Research Questions (full)

RQ4. How do "bag-of-words" and "bag-of-dependencies" approaches compare in terms of performance? Is a bag-of-words approach alone adequate for our needs?

RQ5. Can the accuracy of the system be improved by the inclusion of semantic information from tools like semantic role labelers, WordNet, or word and sentence embeddings?

Research Questions (full)

RQ4. How do “bag-of-words” and “bag-of-dependencies” approaches compare in terms of performance? Is a bag-of-words approach alone adequate for our needs?

RQ5. Can the accuracy of the system be improved by the inclusion of semantic information from tools like semantic role labelers, WordNet, or word and sentence embeddings?

RQ6. What is the annotation scheme for this task and can the system perform within the range of human performance? Relatedly, what does it mean for a response to be *appropriate* and how can this be captured with annotation?

Pilot study: Data

Response (L1)
He is droning his wife pitcher. (Ar)
The artist is drawing a pretty women. (Ch)
The artist is painting a portrait of a lady. (En)
The painter is painting a woman's paint. (Sp)

Figure: Example item from the pilot study showing responses from native speakers of Arabic (Ar), Chinese (Ch), English (En) and Spanish (Sp).

Pilot study: Data

	**Response (L1)**
	He is droning his wife pitcher. (Ar)
	The artist is drawing a pretty women. (Ch)
	The artist is painting a portrait of a lady. (En)
	The painter is painting a woman's paint. (Sp)

Figure: Example item from the pilot study showing responses from native speakers of Arabic (Ar), Chinese (Ch), English (En) and Spanish (Sp).

- 10 (transitive) PDT items $\times$ 53 participants $=$ 530 responses;

Pilot study: Data

Response (L1)
He is droning his wife pitcher. (Ar)
The artist is drawing a pretty women. (Ch)
The artist is painting a portrait of a lady. (En)
The painter is painting a woman's paint. (Sp)

Figure: Example item from the pilot study showing responses from native speakers of Arabic (Ar), Chinese (Ch), English (En) and Spanish (Sp).

- 10 (transitive) PDT items $\times$ 53 participants $=$ 530 responses;
  - 14 NSs (grad students), 39 NNSs (ESL students);

Pilot study: Data

Response (L1)
He is droning his wife pitcher. (Ar)
The artist is drawing a pretty women. (Ch)
The artist is painting a portrait of a lady. (En)
The painter is painting a woman's paint. (Sp)

Figure: Example item from the pilot study showing responses from native speakers of Arabic (Ar), Chinese (Ch), English (En) and Spanish (Sp).

- 10 (transitive) PDT items $\times$ 53 participants $=$ 530 responses;
  - 14 NSs (grad students), 39 NNSs (ESL students);
- Annotation: *Given the prompt, would the response be acceptable to most English speakers? Acceptable/unacceptable*
  - 1 annotator (me)

Pilot study: Data

Response (L1)
He is droning his wife pitcher. (Ar)
The artist is drawing a pretty women. (Ch)
The artist is painting a portrait of a lady. (En)
The painter is painting a woman's paint. (Sp)

Figure: Example item from the pilot study showing responses from native speakers of Arabic (Ar), Chinese (Ch), English (En) and Spanish (Sp).

- 10 (transitive) PDT items $\times$ 53 participants $=$ 530 responses;
  - 14 NSs (grad students), 39 NNSs (ESL students);
- Annotation: *Given the prompt, would the response be acceptable to most English speakers? Acceptable/unacceptable*
  - 1 annotator (me)

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ *V(S,O)* extraction rules;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list $\rightarrow$ covered / not covered;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list $\rightarrow$ covered / not covered;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list $\rightarrow$ covered / not covered;

- Dependency-based

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;
  - Subjects, objects, verbs clearly labeled;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;
  - Subjects, objects, verbs clearly labeled;
- V(S,O) extraction

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;
  - Subjects, objects, verbs clearly labeled;
- V(S,O) extraction
  - Decision tree based on dependency indexing & labels, POS;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
    - Captures aspects of form and meaning;
    - Subjects, objects, verbs clearly labeled;
- V(S,O) extraction
    - Decision tree based on dependency indexing & labels, POS;
    - Custom for my transitive PDT, not generalizable, not robust;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;
  - Subjects, objects, verbs clearly labeled;
- V(S,O) extraction
  - Decision tree based on dependency indexing & labels, POS;
  - Custom for my transitive PDT, not generalizable, not robust;
  - ≈92% accurate, ≈8% extraction errors;

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;
  - Subjects, objects, verbs clearly labeled;
- V(S,O) extraction
  - Decision tree based on dependency indexing & labels, POS;
  - Custom for my transitive PDT, not generalizable, not robust;
  - ≈92% accurate, ≈8% extraction errors;
- Overall accuracy: 58.9%

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list $\rightarrow$ covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;
  - Subjects, objects, verbs clearly labeled;
- V(S,O) extraction
  - Decision tree based on dependency indexing & labels, POS;
  - Custom for my transitive PDT, not generalizable, not robust;
  - $\approx$92% accurate, $\approx$8% extraction errors;
- Overall accuracy: 58.9%

Pilot study: Processing

First approach: **Rule-based** triple extraction and matching

Dependency parser → lemmatizer → *V(S,O)* extraction rules;

Compare NNS *V(S,O)* & NS *V(S,O)* list → covered / not covered;

- Dependency-based
  - Captures aspects of form and meaning;
  - Subjects, objects, verbs clearly labeled;
- V(S,O) extraction
  - Decision tree based on dependency indexing & labels, POS;
  - Custom for my transitive PDT, not generalizable, not robust;
  - ≈92% accurate, ≈8% extraction errors;
- Overall accuracy: 58.9%
  - I.e., *Acceptable* covered, *unacceptable* not covered;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ term frequency-inverse document frequency (tf-idf; "term" $=$ lemmatized dependency);

NNS response score $=$ cosine distance of NS and NNS tf-idf scores;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ term frequency-inverse document frequency (tf-idf; "term" $=$ lemmatized dependency);

NNS response score $=$ cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
  - Get *sorted union set* of NS and NNS dependencies;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
  - Get *sorted union set* of NS and NNS dependencies;
  - NNS vector: Replace deps with their **NNS** tf-idf scores;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser $\rightarrow$ lemmatizer $\rightarrow$ term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
  - Get *sorted union set* of NS and NNS dependencies;
  - NNS vector: Replace deps with their **NNS** tf-idf scores;
  - NS vector: Replace deps with their **NS** tf-idf scores;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
  - Get *sorted union set* of NS and NNS dependencies;
  - NNS vector: Replace deps with their **NNS** tf-idf scores;
  - NS vector: Replace deps with their **NS** tf-idf scores;
  - Response score = *cosine distance* for NNS & NS vectors;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
  - Get *sorted union set* of NS and NNS dependencies;
  - NNS vector: Replace deps with their **NNS** tf-idf scores;
  - NS vector: Replace deps with their **NS** tf-idf scores;
  - Response score = *cosine distance* for NNS & NS vectors;
- Rank by scores & calculate Mean Average Precision (MAP);

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
    - Get *sorted union set* of NS and NNS dependencies;
    - NNS vector: Replace deps with their **NNS** tf-idf scores;
    - NS vector: Replace deps with their **NS** tf-idf scores;
    - Response score = *cosine distance* for NNS & NS vectors;
- Rank by scores & calculate Mean Average Precision (MAP);
    - MAP *acceptable* responses: ≈51%

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
  - Get *sorted union set* of NS and NNS dependencies;
  - NNS vector: Replace deps with their **NNS** tf-idf scores;
  - NS vector: Replace deps with their **NS** tf-idf scores;
  - Response score = *cosine distance* for NNS & NS vectors;
- Rank by scores & calculate Mean Average Precision (MAP);
  - MAP *acceptable* responses: ≈51%
- Process is more robust & generalizable;

Pilot study: Processing

Second approach: **Semantic similarity** scoring

Dependency parser → lemmatizer → term frequency-inverse document frequency (tf-idf; "term" = lemmatized dependency);

NNS response score = cosine distance of NS and NNS tf-idf scores;

- tf-idf: Score dependencies according to importance;
- Vectorize & Score
  - Get *sorted union set* of NS and NNS dependencies;
  - NNS vector: Replace deps with their **NNS** tf-idf scores;
  - NS vector: Replace deps with their **NS** tf-idf scores;
  - Response score = *cosine distance* for NNS & NS vectors;
- Rank by scores & calculate Mean Average Precision (MAP);
  - MAP *acceptable* responses: ≈51%
- Process is more robust & generalizable;
- Dataset (especially NS models) and annotation are weak;

System configuration

System configuration

All parameters or variables and their settings:

Transitivity	**Targeting**	**Familiarity**	**Primacy**	**Term Rep.**
`intransitive`	`targeted`	`familar`	`primary`	`ldh`
`transitive`	`untargeted`	`crowdsourced`	`mixed`	`xdh`
`ditransitive`				`xdx`

A **system configuration** combines one setting from each column.

If particular settings correlate highly with item characteristics (intransitive / transitive / ditransitive; response complexity), I can optimize the system for new items.

Sampling data: Response length

	n=14		n=50	n=70
	Fam	Crowd	Crowd	NNS
Intrans	5.5	4.9	4.9	4.9
Trans	6.9	6.3	6.2	6.7
Ditrans	7.8	7.2	7.2	8.3
Target	6.5	5.4	5.4	6.3
Untarg	6.9	6.8	6.8	6.9
primary	N/A	5.7	5.8	6.6
mixed	6.7	6.5	6.4	N/A
Total	6.7	6.1	6.1	6.6

Table: Comparing average response length (in words) for the samples used throughout this chapter as NS models and NNS test sets, in total and by parameter setting.

Annotation features

Annotation features

First iteration: **accuracy (A)** & **native-likeness (NL)**

Annotation features

First iteration: **accuracy (A)** & **native-likeness (NL)**

- **2**: $+$A, $+$NL $>$ **1**: $+$A, $-$NL $>$ **0**: $-$A, $-$NL

Annotation features

First iteration: **accuracy (A)** & **native-likeness (NL)**

- **2**: $+$A, $+$NL $>$ **1**: $+$A, $-$NL $>$ **0**: $-$A, $-$NL
- Not operationalizable: e.g., response is accurate w.r.t. prompt but adds unverifiable details; is this still *accurate*?
- Not *reliable*, not *valid*;

Annotation features

First iteration: **accuracy (A)** & **native-likeness (NL)**

- **2**: $+$A, $+$NL $>$ **1**: $+$A, $-$NL $>$ **0**: $-$A, $-$NL
- Not operationalizable: e.g., response is accurate w.r.t. prompt but adds unverifiable details; is this still *accurate*?
- Not *reliable*, not *valid*;

This was scrapped and I settled on the 5 binary features.

Annotation features

Annotation features

Inter-rater reliability for two annotators and 10% of the dataset:

Annotation features

Inter-rater reliability for two annotators and 10% of the dataset: *yes* annotations for Annotator 1 (note skewedness), expected chance agreement (*Chance*), actual observed agreement (*Observed*) and Cohen's kappa (*Kappa*)

Set	A1Yes	Chance	Observed	Kappa
Core Event	.733	.601	.923	.808
Answerhood	.834	.721	.982	.936
Grammaticality	.861	.768	.960	.827
Interpretability	.818	.682	.919	.744
Verifiability	.845	.719	.968	.884
`Intransitive`	.863	.758	.978	.910
`Transitive`	.780	.653	.949	.853
`Ditransitive`	.812	.678	.924	.764

Weighting features

Raters perform holistic preference test (blind to annotations)

Weighting features

Raters perform holistic preference test (blind to annotations)

What is the boy doing?	Pref?	Core	Ansr	Gram	Intrp	Verif
He is eating food.	**yes**	**0**	**1**	**1**	**1**	**1**
He may get fat eating.	no	0	0	1	1	0

Weighting features

Raters perform holistic preference test (blind to annotations)

What is the boy doing?	Pref?	Core	Ansr	Gram	Intrp	Verif
He is eating food.	**yes**	**0**	**1**	**1**	**1**	**1**
He may get fat eating.	no	0	0	1	1	0
He is hungry.	no	0	0	1	0	1
the boy is eating pizza	**yes**	**1**	**1**	**1**	**1**	**1**

Weighting features

Raters perform holistic preference test (blind to annotations)

What is the boy doing?	Pref?	Core	Ansr	Gram	Intrp	Verif
He is eating food.	**yes**	**0**	**1**	**1**	**1**	**1**
He may get fat eating.	no	0	0	1	1	0
He is hungry.	no	0	0	1	0	1
the boy is eating pizza	**yes**	**1**	**1**	**1**	**1**	**1**
The child is about to eat pizza.	**yes**	**1**	**0**	**1**	**1**	**1**
he eating.	no	0	1	0	1	1

Weighting features

Raters perform holistic preference test (blind to annotations)

What is the boy doing?	Pref?	Core	Ansr	Gram	Intrp	Verif
He is eating food.	**yes**	**0**	**1**	**1**	**1**	**1**
He may get fat eating.	no	0	0	1	1	0
He is hungry.	no	0	0	1	0	1
the boy is eating pizza	**yes**	**1**	**1**	**1**	**1**	**1**
The child is about to eat pizza.	**yes**	**1**	**0**	**1**	**1**	**1**
he eating.	no	0	1	0	1	1
Totals preferred responses		2	2	3	3	3
Totals dispreferred responses		0	1	2	2	2
Net preferred (pref - dispref)		2	1	1	1	1
Feature weight		.333	.167	.167	.167	.167

Weighting features

Raters perform holistic preference test (blind to annotations)

What is the boy doing?	Pref?	Core	Ansr	Gram	Intrp	Verif
He is eating food.	**yes**	**0**	**1**	**1**	**1**	**1**
He may get fat eating.	no	0	0	1	1	0
He is hungry.	no	0	0	1	0	1
the boy is eating pizza	**yes**	**1**	**1**	**1**	**1**	**1**
The child is about to eat pizza.	**yes**	**1**	**0**	**1**	**1**	**1**
he eating.	no	0	1	0	1	1
Totals preferred responses		2	2	3	3	3
Totals dispreferred responses		0	1	2	2	2
Net preferred (pref - dispref)		2	1	1	1	1
Feature weight		.333	.167	.167	.167	.167
*Real feature weight		.365	.093	.055	.224	.263

Mean Average Precision

Average precision represents the area under the precision-recall curve. *Mean* average precision is an average over multiple average precisions (here it's from multiple PDT items or datasets).

This is a simplification, but for our purposes here, we can think of MAP as a measure of how well a ranking separates "yes" and "no" annotations.

Bad	Okay	Good
yes	**yes**	**yes**
no	**yes**	**yes**
no	no	**yes**
yes	**yes**	no
yes	no	no

	Bad	Okay	Good
AP $\rightarrow$	.51	.64	1.0

$$\text{MAP} = (0.51 + 0.64 + 1.00)/3 \approx 0.72$$

Predicting features: Answerhood MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.868	.871	***.878***	.881	.869	.866	.868	**.874**	.881	.868
Tran	.816	.819	**.846**	.845	.838	.818	.823	***.851***	.845	.838
Ditr	.824	.826	***.841***	.837	.833	.821	.822	**.840**	.837	.833
Targ	.787	.788	**.810**	.817	.799	.787	.789	***.811***	.817	.798
Untg	.885	.890	***.900***	.892	.894	.883	.886	**.899**	.892	.895
Prim	.837	.840	***.854***	.854	.845	.837	.840	***.854***	.854	.846
Mix	.835	.838	***.857***	.854	.848	.833	.835	**.856**	.854	.847
Total	.836	.839	***.855***	.854	.847	.835	.838	**.855**	.854	.846

Predicting features: ANSWERHOOD MAP

	Crowd NS model = 14					Crowd NS model = 50				
	`ldh`	`xdh`	`xdx`	WAR	SBERT	`ldh`	`xdh`	`xdx`	WAR	SBERT
`Intr`	.868	.871	***.878***	.881	.869	.866	.868	**.874**	.881	.868
`Tran`	.816	.819	**.846**	.845	.838	.818	.823	***.851***	.845	.838
`Ditr`	.824	.826	***.841***	.837	.833	.821	.822	**.840**	.837	.833
`Targ`	.787	.788	**.810**	.817	.799	.787	.789	***.811***	.817	.798
`Untg`	.885	.890	***.900***	.892	.894	.883	.886	**.899**	.892	.895
`Prim`	.837	.840	***.854***	.854	.845	.837	.840	***.854***	.854	.846
`Mix`	.835	.838	***.857***	.854	.848	.833	.835	**.856**	.854	.847
Total	.836	.839	***.855***	.854	.847	.835	.838	**.855**	.854	.846

- `xdx` > `xdh` > `ldh`;

Predicting features: ANSWERHOOD MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.868	.871	***.878***	.881	.869	.866	.868	**.874**	.881	.868
Tran	.816	.819	**.846**	.845	.838	.818	.823	***.851***	.845	.838
Ditr	.824	.826	***.841***	.837	.833	.821	.822	**.840**	.837	.833
Targ	.787	.788	**.810**	.817	.799	.787	.789	***.811***	.817	.798
Untg	.885	.890	***.900***	.892	.894	.883	.886	**.899**	.892	.895
Prim	.837	.840	***.854***	.854	.845	.837	.840	***.854***	.854	.846
Mix	.835	.838	***.857***	.854	.848	.833	.835	**.856**	.854	.847
Total	.836	.839	***.855***	.854	.847	.835	.838	**.855**	.854	.846

- xdx > xdh > ldh;
- Model size makes little difference;

Predicting ANSWERHOOD is relatively simple; requires only small model and bag-of-words representation.

Predicting features: ANSWERHOOD MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.868	.871	***.882***	.881	.868	.869	.873	**.878**	.881	.870
Tran	.824	.826	***.852***	.845	.840	.817	.818	**.847**	.845	.840
Ditr	.820	.822	***.846***	.837	.832	.820	.822	**.845**	.837	.835
Targ	.786	.787	***.815***	.817	.798	.785	.787	**.813**	.817	.802
Untg	.889	.892	***.904***	.892	.896	.885	.889	**.900**	.892	.894
Total	.837	.840	***.860***	.854	.847	.835	.838	**.857**	.854	.848

Predicting features: Answerhood MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.868	.871	***.882***	.881	.868	.869	.873	**.878**	.881	.870
Tran	.824	.826	***.852***	.845	.840	.817	.818	**.847**	.845	.840
Ditr	.820	.822	***.846***	.837	.832	.820	.822	**.845**	.837	.835
Targ	.786	.787	***.815***	.817	.798	.785	.787	**.813**	.817	.802
Untg	.889	.892	***.904***	.892	.896	.885	.889	**.900**	.892	.894
Total	.837	.840	***.860***	.854	.847	.835	.838	**.857**	.854	.848

- xdx > xdh > ldh;

Predicting features: Answerhood MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.868	.871	***.882***	.881	.868	.869	.873	**.878**	.881	.870
Tran	.824	.826	***.852***	.845	.840	.817	.818	**.847**	.845	.840
Ditr	.820	.822	***.846***	.837	.832	.820	.822	**.845**	.837	.835
Targ	.786	.787	***.815***	.817	.798	.785	.787	**.813**	.817	.802
Untg	.889	.892	***.904***	.892	.896	.885	.889	**.900**	.892	.894
Total	.837	.840	***.860***	.854	.847	.835	.838	**.857**	.854	.848

- xdx > xdh > ldh;
- familiar > crowdsourced;

Predicting ANSWERHOOD is relatively simple; requires only small model and bag-of-words representation.

Predicting features: Grammaticality MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.868	.870	***.872***	.887	.866	.863	.864	**.866**	.887	.864
Tran	.753	.756	**.757**	.781	.757	.758	.760	***.761***	.781	.757
Ditr	.682	.685	***.700***	.695	.694	.679	.685	**.697**	.695	.693
Targ	.777	.778	***.784***	.800	.782	.776	.776	**.783**	.800	.781
Untg	.758	.763	***.769***	.776	.762	.757	.762	**.766**	.776	.761
Prim	.769	.773	***.776***	.788	.770	.768	.770	**.774**	.788	.770
Mix	.766	.768	***.776***	.788	.774	.765	.768	**.775**	.788	.772
Total	.768	.770	***.776***	.788	.772	.767	.769	**.775**	.788	.771

Predicting features: GRAMMATICALITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	`ldh`	`xdh`	`xdx`	WAR	SBERT	`ldh`	`xdh`	`xdx`	WAR	SBERT
`Intr`	.868	.870	***.872***	.887	.866	.863	.864	**.866**	.887	.864
`Tran`	.753	.756	**.757**	.781	.757	.758	.760	***.761***	.781	.757
`Ditr`	.682	.685	***.700***	.695	.694	.679	.685	**.697**	.695	.693
`Targ`	.777	.778	***.784***	.800	.782	.776	.776	**.783**	.800	.781
`Untg`	.758	.763	***.769***	.776	.762	.757	.762	**.766**	.776	.761
`Prim`	.769	.773	***.776***	.788	.770	.768	.770	**.774**	.788	.770
`Mix`	.766	.768	***.776***	.788	.774	.765	.768	**.775**	.788	.772
Total	.768	.770	***.776***	.788	.772	.767	.769	**.775**	.788	.771

In *most* cases:

Predicting features: GRAMMATICALITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.868	.870	***.872***	.887	.866	.863	.864	**.866**	.887	.864
Tran	.753	.756	**.757**	.781	.757	.758	.760	***.761***	.781	.757
Ditr	.682	.685	***.700***	.695	.694	.679	.685	**.697**	.695	.693
Targ	.777	.778	***.784***	.800	.782	.776	.776	**.783**	.800	.781
Untg	.758	.763	***.769***	.776	.762	.757	.762	**.766**	.776	.761
Prim	.769	.773	***.776***	.788	.770	.768	.770	**.774**	.788	.770
Mix	.766	.768	***.776***	.788	.774	.765	.768	**.775**	.788	.772
Total	.768	.770	***.776***	.788	.772	.767	.769	**.775**	.788	.771

In *most* cases:

- xdx > xdh > ldh;

Predicting features: GRAMMATICALITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	`ldh`	`xdh`	`xdx`	WAR	SBERT	`ldh`	`xdh`	`xdx`	WAR	SBERT
`Intr`	.868	.870	***.872***	.887	.866	.863	.864	**.866**	.887	.864
`Tran`	.753	.756	**.757**	.781	.757	.758	.760	***.761***	.781	.757
`Ditr`	.682	.685	***.700***	.695	.694	.679	.685	**.697**	.695	.693
`Targ`	.777	.778	***.784***	.800	.782	.776	.776	**.783**	.800	.781
`Untg`	.758	.763	***.769***	.776	.762	.757	.762	**.766**	.776	.761
`Prim`	.769	.773	***.776***	.788	.770	.768	.770	**.774**	.788	.770
`Mix`	.766	.768	***.776***	.788	.774	.765	.768	**.775**	.788	.772
Total	.768	.770	***.776***	.788	.772	.767	.769	**.775**	.788	.771

In *most* cases:

- `xdx` > `xdh` > `ldh`;

Predicting GRAMMATICALITY is relatively simple; requires only small model and bag-of-words representation.

Predicting features: GRAMMATICALITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.863	.864	**.873**	.887	.863	.868	.869	***.874***	.887	.869
Tran	.760	.759	***.762***	.781	.760	.752	.754	**.757**	.781	.758
Ditr	.678	.685	**.698**	.695	.698	.678	.680	***.699***	.695	.696
Targ	.776	.776	***.787***	.800	.783	.776	.777	**.786**	.800	.786
Untg	.757	.762	***.768***	.776	.764	.756	.759	**.767**	.776	.763
Total	.767	.769	***.778***	.788	.773	.766	.768	**.776**	.788	.774

Predicting features: Grammaticality MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.863	.864	***.873***	.887	.863	.868	.869	***.874***	.887	.869
Tran	.760	.759	***.762***	.781	.760	.752	.754	**.757**	.781	.758
Ditr	.678	.685	**.698**	.695	.698	.678	.680	***.699***	.695	.696
Targ	.776	.776	***.787***	.800	.783	.776	.777	**.786**	.800	.786
Untg	.757	.762	***.768***	.776	.764	.756	.759	**.767**	.776	.763
Total	.767	.769	***.778***	.788	.773	.766	.768	**.776**	.788	.774

In *most* cases:

Predicting features: GRAMMATICALITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.863	.864	**.873**	.887	.863	.868	.869	***.874***	.887	.869
Tran	.760	.759	***.762***	.781	.760	.752	.754	**.757**	.781	.758
Ditr	.678	.685	**.698**	.695	.698	.678	.680	***.699***	.695	.696
Targ	.776	.776	***.787***	.800	.783	.776	.777	**.786**	.800	.786
Untg	.757	.762	***.768***	.776	.764	.756	.759	**.767**	.776	.763
Total	.767	.769	***.778***	.788	.773	.766	.768	**.776**	.788	.774

In *most* cases:

- xdx > xdh > ldh;

Predicting features: GRAMMATICALITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	`ldh`	`xdh`	`xdx`	WAR	SBERT	`ldh`	`xdh`	`xdx`	WAR	SBERT
`Intr`	.863	.864	**.873**	.887	.863	.868	.869	***.874***	.887	.869
`Tran`	.760	.759	***.762***	.781	.760	.752	.754	**.757**	.781	.758
`Ditr`	.678	.685	**.698**	.695	.698	.678	.680	***.699***	.695	.696
`Targ`	.776	.776	***.787***	.800	.783	.776	.777	**.786**	.800	.786
`Untg`	.757	.762	***.768***	.776	.764	.756	.759	**.767**	.776	.763
Total	.767	.769	***.778***	.788	.773	.766	.768	**.776**	.788	.774

In *most* cases:

- `xdx` > `xdh` > `ldh`;
- `familiar` 14NS > `crowd` 14NS > `crowd` 50NS;

Predicting features: GRAMMATICALITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.863	.864	**.873**	.887	.863	.868	.869	***.874***	.887	.869
Tran	.760	.759	***.762***	.781	.760	.752	.754	**.757**	.781	.758
Ditr	.678	.685	**.698**	.695	.698	.678	.680	***.699***	.695	.696
Targ	.776	.776	***.787***	.800	.783	.776	.777	**.786**	.800	.786
Untg	.757	.762	***.768***	.776	.764	.756	.759	**.767**	.776	.763
Total	.767	.769	***.778***	.788	.773	.766	.768	**.776**	.788	.774

In *most* cases:

- xdx > xdh > ldh;
- familiar 14NS > crowd 14NS > crowd 50NS;

Predicting GRAMMATICALITY is relatively simple; requires only small model and bag-of-words representation.

Predicting features: INTERPRETABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.932	.931	***.933***	.930	.922	.928	.927	**.933**	.930	.923
Tran	***.823***	.821	.811	.803	.806	**.821**	.816	.812	.803	.804
Ditr	.789	.784	***.794***	.721	.777	.786	.782	**.792**	.721	.772
Targ	.835	.832	***.836***	.804	.828	.833	.829	**.834**	.804	.826
Untg	***.862***	.858	.856	.833	.842	**.857**	.855	**.857**	.833	.840
Prim	***.847***	.845	.846	.818	.837	.845	.842	**.846**	.818	.833
Mix	***.849***	.846	.846	.818	.833	.844	.841	**.845**	.818	.833
Total	***.848***	.845	.846	.818	.835	**.845**	.842	**.845**	.818	.833

Predicting features: INTERPRETABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.932	.931	***.933***	.930	.922	.928	.927	**.933**	.930	.923
Tran	***.823***	.821	.811	.803	.806	**.821**	.816	.812	.803	.804
Ditr	.789	.784	***.794***	.721	.777	.786	.782	**.792**	.721	.772
Targ	.835	.832	***.836***	.804	.828	.833	.829	**.834**	.804	.826
Untg	***.862***	.858	.856	.833	.842	**.857**	.855	**.857**	.833	.840
Prim	***.847***	.845	.846	.818	.837	.845	.842	**.846**	.818	.833
Mix	***.849***	.846	.846	.818	.833	.844	.841	**.845**	.818	.833
Total	***.848***	.845	.846	.818	.835	**.845**	.842	**.845**	.818	.833

- 14NS crowdsourced > 50NS crowdsourced;

Predicting features: INTERPRETABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.932	.931	***.933***	.930	.922	.928	.927	**.933**	.930	.923
Tran	***.823***	.821	.811	.803	.806	**.821**	.816	.812	.803	.804
Ditr	.789	.784	***.794***	.721	.777	.786	.782	**.792**	.721	.772
Targ	.835	.832	***.836***	.804	.828	.833	.829	**.834**	.804	.826
Untg	***.862***	.858	.856	.833	.842	**.857**	.855	**.857**	.833	.840
Prim	***.847***	.845	.846	.818	.837	.845	.842	**.846**	.818	.833
Mix	***.849***	.846	.846	.818	.833	.844	.841	**.845**	.818	.833
Total	***.848***	.845	.846	.818	.835	**.845**	.842	**.845**	.818	.833

- 14NS crowdsourced > 50NS crowdsourced;
- intransitives & ditransitives work best with xdx;

Predicting features: INTERPRETABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.932	.931	***.933***	.930	.922	.928	.927	**.933**	.930	.923
Tran	***.823***	.821	.811	.803	.806	**.821**	.816	.812	.803	.804
Ditr	.789	.784	***.794***	.721	.777	.786	.782	**.792**	.721	.772
Targ	.835	.832	***.836***	.804	.828	.833	.829	**.834**	.804	.826
Untg	***.862***	.858	.856	.833	.842	**.857**	.855	**.857**	.833	.840
Prim	***.847***	.845	.846	.818	.837	.845	.842	**.846**	.818	.833
Mix	***.849***	.846	.846	.818	.833	.844	.841	**.845**	.818	.833
Total	***.848***	.845	.846	.818	.835	**.845**	.842	**.845**	.818	.833

- 14NS crowdsourced > 50NS crowdsourced;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;

Predicting features: INTERPRETABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.932	.931	***.933***	.930	.922	.928	.927	**.933**	.930	.923
Tran	***.823***	.821	.811	.803	.806	**.821**	.816	.812	.803	.804
Ditr	.789	.784	***.794***	.721	.777	.786	.782	**.792**	.721	.772
Targ	.835	.832	***.836***	.804	.828	.833	.829	**.834**	.804	.826
Untg	***.862***	.858	.856	.833	.842	**.857**	.855	**.857**	.833	.840
Prim	***.847***	.845	.846	.818	.837	.845	.842	**.846**	.818	.833
Mix	***.849***	.846	.846	.818	.833	.844	.841	**.845**	.818	.833
Total	***.848***	.845	.846	.818	.835	**.845**	.842	**.845**	.818	.833

- 14NS crowdsourced > 50NS crowdsourced;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;
  - Why? Transitive responses are relatively homogenous; Annotators relatively strict;

Predicting features: Interpretability MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.930	.930	***.934***	.930	.923	**.933**	.931	.932	.930	.922
Tran	**.822**	.819	.811	.803	.805	***.826***	.824	.811	.803	.805
Ditr	.787	.786	***.796***	.721	.782	.788	.783	**.795**	.721	.772
Targ	.835	.833	***.836***	.804	.830	**.835**	.832	**.835**	.804	.825
Untg	**.858**	.857	**.858**	.833	.843	***.863***	.859	.857	.833	.841
Total	**.847**	.845	**.847**	.818	.837	***.849***	.846	.846	.818	.833

Predicting features: Interpretability MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.930	.930	***.934***	.930	.923	**.933**	.931	.932	.930	.922
Tran	**.822**	.819	.811	.803	.805	***.826***	.824	.811	.803	.805
Ditr	.787	.786	***.796***	.721	.782	.788	.783	**.795**	.721	.772
Targ	.835	.833	***.836***	.804	.830	**.835**	.832	**.835**	.804	.825
Untg	**.858**	.857	**.858**	.833	.843	***.863***	.859	.857	.833	.841
Total	**.847**	.845	**.847**	.818	.837	***.849***	.846	.846	.818	.833

- 14NS crowdsourced > 14NS familiar;

Predicting features: INTERPRETABILITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.930	.930	***.934***	.930	.923	**.933**	.931	.932	.930	.922
Tran	**.822**	.819	.811	.803	.805	***.826***	.824	.811	.803	.805
Ditr	.787	.786	***.796***	.721	.782	.788	.783	**.795**	.721	.772
Targ	.835	.833	***.836***	.804	.830	**.835**	.832	**.835**	.804	.825
Untg	**.858**	.857	**.858**	.833	.843	***.863***	.859	.857	.833	.841
Total	**.847**	.845	**.847**	.818	.837	***.849***	.846	.846	.818	.833

- 14NS crowdsourced > 14NS familiar;
- intransitives & ditransitives work best with xdx;

Predicting features: INTERPRETABILITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.930	.930	***.934***	.930	.923	**.933**	.931	.932	.930	.922
Tran	**.822**	.819	.811	.803	.805	***.826***	.824	.811	.803	.805
Ditr	.787	.786	***.796***	.721	.782	.788	.783	**.795**	.721	.772
Targ	.835	.833	***.836***	.804	.830	**.835**	.832	**.835**	.804	.825
Untg	**.858**	.857	**.858**	.833	.843	***.863***	.859	.857	.833	.841
Total	**.847**	.845	**.847**	.818	.837	***.849***	.846	.846	.818	.833

- 14NS crowdsourced > 14NS familiar;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;

Predicting features: INTERPRETABILITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.930	.930	***.934***	.930	.923	**.933**	.931	.932	.930	.922
Tran	**.822**	.819	.811	.803	.805	***.826***	.824	.811	.803	.805
Ditr	.787	.786	***.796***	.721	.782	.788	.783	**.795**	.721	.772
Targ	.835	.833	***.836***	.804	.830	**.835**	.832	**.835**	.804	.825
Untg	**.858**	.857	**.858**	.833	.843	***.863***	.859	.857	.833	.841
Total	**.847**	.845	**.847**	.818	.837	***.849***	.846	.846	.818	.833

- 14NS crowdsourced > 14NS familiar;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;
  - Why? Transitive responses are relatively homogenous; Annotators relatively strict;

Predicting features: VERIFIABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.852	.852	***.853***	.866	.840	.849	.849	**.851**	.866	.836
Tran	***.809***	.808	.803	.798	.787	**.807**	.806	.803	.798	.785
Ditr	.814	.812	***.815***	.780	.798	.811	.809	**.812**	.780	.796
Targ	***.825***	.824	.825	.815	.812	**.825**	.824	.823	.815	.810
Untg	***.825***	.824	.822	.815	.805	**.820**	.819	.820	.815	.802
Prim	***.826***	.824	.823	.815	.808	**.824**	.823	.822	.815	.806
Mix	***.825***	.824	.824	.815	.808	**.821**	.821	.821	.815	.805
Total	***.825***	.824	.824	.815	.808	**.823**	.822	.822	.815	.806

Predicting features: VERIFIABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.852	.852	***.853***	.866	.840	.849	.849	**.851**	.866	.836
Tran	***.809***	.808	.803	.798	.787	**.807**	.806	.803	.798	.785
Ditr	.814	.812	***.815***	.780	.798	.811	.809	**.812**	.780	.796
Targ	***.825***	.824	.825	.815	.812	**.825**	.824	.823	.815	.810
Untg	***.825***	.824	.822	.815	.805	**.820**	.819	.820	.815	.802
Prim	***.826***	.824	.823	.815	.808	**.824**	.823	.822	.815	.806
Mix	***.825***	.824	.824	.815	.808	**.821**	.821	.821	.815	.805
Total	***.825***	.824	.824	.815	.808	**.823**	.822	.822	.815	.806

- 14NS crowd > 50NS crowd;

Predicting features: Verifiability MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.852	.852	***.853***	.866	.840	.849	.849	**.851**	.866	.836
Tran	***.809***	.808	.803	.798	.787	**.807**	.806	.803	.798	.785
Ditr	.814	.812	***.815***	.780	.798	.811	.809	**.812**	.780	.796
Targ	***.825***	.824	.825	.815	.812	**.825**	.824	.823	.815	.810
Untg	***.825***	.824	.822	.815	.805	**.820**	.819	.820	.815	.802
Prim	***.826***	.824	.823	.815	.808	**.824**	.823	.822	.815	.806
Mix	***.825***	.824	.824	.815	.808	**.821**	.821	.821	.815	.805
Total	***.825***	.824	.824	.815	.808	**.823**	.822	.822	.815	.806

- 14NS crowd > 50NS crowd;
- intransitives & ditransitives work best with xdx;

Predicting features: Verifiability MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.852	.852	***.853***	.866	.840	.849	.849	**.851**	.866	.836
Tran	***.809***	.808	.803	.798	.787	**.807**	.806	.803	.798	.785
Ditr	.814	.812	***.815***	.780	.798	.811	.809	**.812**	.780	.796
Targ	***.825***	.824	.825	.815	.812	**.825**	.824	.823	.815	.810
Untg	***.825***	.824	.822	.815	.805	**.820**	.819	.820	.815	.802
Prim	***.826***	.824	.823	.815	.808	**.824**	.823	.822	.815	.806
Mix	***.825***	.824	.824	.815	.808	**.821**	.821	.821	.815	.805
Total	***.825***	.824	.824	.815	.808	**.823**	.822	.822	.815	.806

- 14NS crowd > 50NS crowd;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;

Predicting features: VERIFIABILITY MAP

	Crowd NS model = 14					Crowd NS model = 50				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.852	.852	***.853***	.866	.840	.849	.849	**.851**	.866	.836
Tran	***.809***	.808	.803	.798	.787	**.807**	.806	.803	.798	.785
Ditr	.814	.812	***.815***	.780	.798	.811	.809	**.812**	.780	.796
Targ	***.825***	.824	.825	.815	.812	**.825**	.824	.823	.815	.810
Untg	***.825***	.824	.822	.815	.805	**.820**	.819	.820	.815	.802
Prim	***.826***	.824	.823	.815	.808	**.824**	.823	.822	.815	.806
Mix	***.825***	.824	.824	.815	.808	**.821**	.821	.821	.815	.805
Total	***.825***	.824	.824	.815	.808	**.823**	.822	.822	.815	.806

- 14NS crowd > 50NS crowd;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;
  - Why? Transitive responses are relatively homogenous; Annotators relatively strict;

Predicting features: Verifiability MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.847	.847	**.852**	.866	.836	.852	.852	***.854***	.866	.843
Tran	***.808***	.807	.803	.798	.787	**.807**	.807	.802	.798	.786
Ditr	.811	.811	**.812**	.780	.802	.815	.812	***.817***	.780	.796
Targ	.821	.821	**.822**	.815	.814	.824	.824	***.826***	.815	.811
Untg	**.824**	.822	.823	.815	.803	***.825***	.824	.823	.815	.806
Total	.822	.822	**.823**	.815	.808	***.825***	.824	.824	.815	.808

Predicting features: VERIFIABILITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.847	.847	**.852**	.866	.836	.852	.852	***.854***	.866	.843
Tran	***.808***	.807	.803	.798	.787	**.807**	.807	.802	.798	.786
Ditr	.811	.811	**.812**	.780	.802	.815	.812	***.817***	.780	.796
Targ	.821	.821	**.822**	.815	.814	.824	.824	***.826***	.815	.811
Untg	**.824**	.822	.823	.815	.803	***.825***	.824	.823	.815	.806
Total	.822	.822	**.823**	.815	.808	***.825***	.824	.824	.815	.808

- 14NS crowd > 14NS familiar;

Predicting features: VERIFIABILITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.847	.847	**.852**	.866	.836	.852	.852	***.854***	.866	.843
Tran	***.808***	.807	.803	.798	.787	**.807**	.807	.802	.798	.786
Ditr	.811	.811	**.812**	.780	.802	.815	.812	***.817***	.780	.796
Targ	.821	.821	**.822**	.815	.814	.824	.824	***.826***	.815	.811
Untg	**.824**	.822	.823	.815	.803	***.825***	.824	.823	.815	.806
Total	.822	.822	**.823**	.815	.808	***.825***	.824	.824	.815	.808

- 14NS crowd > 14NS familiar;
- intransitives & ditransitives work best with xdx;

Predicting features: VERIFIABILITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.847	.847	**.852**	.866	.836	.852	.852	***.854***	.866	.843
Tran	***.808***	.807	.803	.798	.787	**.807**	.807	.802	.798	.786
Ditr	.811	.811	**.812**	.780	.802	.815	.812	***.817***	.780	.796
Targ	.821	.821	**.822**	.815	.814	.824	.824	***.826***	.815	.811
Untg	**.824**	.822	.823	.815	.803	***.825***	.824	.823	.815	.806
Total	.822	.822	**.823**	.815	.808	***.825***	.824	.824	.815	.808

- 14NS crowd > 14NS familiar;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;

Predicting features: VERIFIABILITY MAP

	Familiar NS model = 14					Crowd NS model = 14				
	ldh	xdh	xdx	WAR	SBERT	ldh	xdh	xdx	WAR	SBERT
Intr	.847	.847	**.852**	.866	.836	.852	.852	***.854***	.866	.843
Tran	***.808***	.807	.803	.798	.787	**.807**	.807	.802	.798	.786
Ditr	.811	.811	**.812**	.780	.802	.815	.812	***.817***	.780	.796
Targ	.821	.821	**.822**	.815	.814	.824	.824	***.826***	.815	.811
Untg	**.824**	.822	.823	.815	.803	***.825***	.824	.823	.815	.806
Total	.822	.822	**.823**	.815	.808	***.825***	.824	.824	.815	.808

- 14NS crowd > 14NS familiar;
- intransitives & ditransitives work best with xdx;
- transitives work best with ldh;
  - Why? Transitive responses are relatively homogenous; Annotators relatively strict;

Predicting quality: `Targeting`

Spearman rank correlations: System vs. WAR (benchmark)

		`targeted`		`untargeted`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	**.380**	.530	.300	.444
	median	**.369**	.545	.314	.472
50NS	mean	***.393***	.550	*.305*	.469
	median	***.389***	.564	*.323*	.496

Predicting quality: `Targeting`

Spearman rank correlations: System vs. WAR (benchmark)

		`targeted`		`untargeted`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	**.380**	.530	.300	.444
	median	**.369**	.545	.314	.472
50NS	mean	***.393***	.550	*.305*	.469
	median	***.389***	.564	*.323*	.496

- `targeted` $>$ `untargeted`

Predicting quality: `Targeting`

Spearman rank correlations: System vs. WAR (benchmark)

		`targeted`		`untargeted`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	**.380**	.530	.300	.444
14NS	median	**.369**	.545	.314	.472
50NS	mean	***.393***	.550	*.305*	.469
50NS	median	***.389***	.564	*.323*	.496

- `targeted` > `untargeted`
- 50NS models > 14NS models

Predicting quality: `Targeting`

Spearman rank correlations: System vs. WAR (benchmark)

		`targeted`		`untargeted`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	**.380**	.530	.300	.444
	median	**.369**	.545	.314	.472
50NS	mean	***.393***	.550	*.305*	.469
	median	***.389***	.564	*.323*	.496

- `targeted` > `untargeted`
- 50NS models > 14NS models
  - Model size effect is most pronounced for `targeted`

Predicting quality: `Familiarity`

Spearman rank correlations: System vs. WAR (benchmark)

		familiar		crowdsourced	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.338	.499	**.339**	.481
	median	**.329**	.513	.326	.500

Predicting quality: `Familiarity`

Spearman rank correlations: System vs. WAR (benchmark)

		`familiar`		`crowdsourced`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.338	.499	**.339**	.481
	median	**.329**	.513	.326	.500

- System: No discernible difference for `familiar` vs `crowdsourced`

Predicting quality: `Familiarity`

Spearman rank correlations: System vs. WAR (benchmark)

		`familiar`		`crowdsourced`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.338	.499	**.339**	.481
	median	**.329**	.513	.326	.500

- System: No discernible difference for `familiar` vs `crowdsourced`
- SBERT: `familiar` $>$ `crowdsourced`

Predicting quality: `Familiarity`

Spearman rank correlations: System vs. WAR (benchmark)

		familiar		crowdsourced	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.338	.499	**.339**	.481
	median	**.329**	.513	.326	.500

- System: No discernible difference for `familiar` vs `crowdsourced`
- SBERT: `familiar` > `crowdsourced`
  - NNS STTR < `familiar` STTR < `crowdsourced` STTR

Predicting quality: `Primacy`

Spearman rank correlations: System vs. WAR (benchmark)

		primary		mixed	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.339	.493	**.340**	.481
14NS	median	.326	.517	**.334**	.500
50NS	mean	***.354***	.514	.344	.505
50NS	median	.345	.532	***.350***	.518

Predicting quality: `Primacy`

Spearman rank correlations: System vs. WAR (benchmark)

		`primary`		`mixed`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.339	.493	**.340**	.481
14NS	median	.326	.517	**.334**	.500
50NS	mean	***.354***	.514	.344	.505
50NS	median	.345	.532	***.350***	.518

Predicting quality: `Primacy`

Spearman rank correlations: System vs. WAR (benchmark)

		`primary`		`mixed`	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.339	.493	**.340**	.481
14NS	median	.326	.517	**.334**	.500
50NS	mean	***.354***	.514	.344	.505
50NS	median	.345	.532	***.350***	.518

- System: 14NS: `primary` $<$ `mixed` (slight difference)

Predicting quality: `Primacy`

Spearman rank correlations: System vs. WAR (benchmark)

		primary		mixed	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.339	.493	**.340**	.481
	median	.326	.517	**.334**	.500
50NS	mean	***.354***	.514	.344	.505
	median	.345	.532	***.350***	.518

- System: 14NS: `primary` < `mixed` (slight difference)
- System: 50NS: `primary` ≈ `mixed`

Predicting quality: `Primacy`

Spearman rank correlations: System vs. WAR (benchmark)

		primary		mixed	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.339	.493	**.340**	.481
	median	.326	.517	**.334**	.500
50NS	mean	***.354***	.514	.344	.505
	median	.345	.532	***.350***	.518

- System: 14NS: `primary` $<$ `mixed` (slight difference)
- System: 50NS: `primary` $\approx$ `mixed`
- SBERT: `primary` $>$ `mixed`

Predicting quality: `Primacy`

Spearman rank correlations: System vs. WAR (benchmark)

		primary		mixed	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.339	.493	**.340**	.481
14NS	median	.326	.517	**.334**	.500
50NS	mean	***.354***	.514	.344	.505
50NS	median	.345	.532	***.350***	.518

- System: 14NS: `primary` < `mixed` (slight difference)
- System: 50NS: `primary` ≈ `mixed`
- SBERT: `primary` > `mixed`
- System & SBERT: 50NS > 14NS

Predicting quality: `Primacy`

Spearman rank correlations: System vs. WAR (benchmark)

		primary		mixed	
		System	SBERT	System	SBERT
	count	180	60	180	60
14NS	mean	.339	.493	**.340**	.481
14NS	median	.326	.517	**.334**	.500
50NS	mean	***.354***	.514	.344	.505
50NS	median	.345	.532	***.350***	.518

- System: 14NS: `primary` < `mixed` (slight difference)
- System: 50NS: `primary` ≈ `mixed`
- SBERT: `primary` > `mixed`
- System & SBERT: 50NS > 14NS
  - System: model size effect is greatest for `primary`

Predicting quality: `Term Representation`

Spearman rank correlations: System vs. WAR (benchmark)

		System			SBERT
		`ldh`	`xdh`	`xdx`	(text)
	count	120	120	120	40
14NS	mean	.333	.336	***.351***	.487
	median	.318	**.344**	.330	.507
50NS	mean	***.350***	*.349*	.348	.509
	median	*.364*	***.374***	*.331*	.523

Predicting quality: `Term Representation`

Spearman rank correlations: System vs. WAR (benchmark)

		System			SBERT
		`ldh`	`xdh`	`xdx`	(text)
	count	120	120	120	40
14NS	mean	.333	.336	***.351***	.487
	median	.318	**.344**	.330	.507
50NS	mean	***.350***	*.349*	.348	.509
	median	*.364*	***.374***	*.331*	.523

- SBERT: 50NS > 14NS

Predicting quality: `Term Representation`

Spearman rank correlations: System vs. WAR (benchmark)

		System			SBERT
		`ldh`	`xdh`	`xdx`	(text)
	count	120	120	120	40
14NS	mean	.333	.336	***.351***	.487
	median	.318	**.344**	.330	.507
50NS	mean	***.350***	*.349*	.348	.509
	median	*.364*	***.374***	*.331*	.523

- SBERT: 50NS > 14NS
- System: for `ldh` & `xdh`: 50NS > 14NS;

Predicting quality: `Term Representation`

Spearman rank correlations: System vs. WAR (benchmark)

		System			SBERT
		`ldh`	`xdh`	`xdx`	(text)
	count	120	120	120	40
14NS	mean	.333	.336	***.351***	.487
	median	.318	**.344**	.330	.507
50NS	mean	***.350***	*.349*	.348	.509
	median	*.364*	***.374***	*.331*	.523

- SBERT: 50NS $>$ 14NS
- System: for `ldh` & `xdh`: 50NS $>$ 14NS;
  - Model size effect is greater for `ldh`

Predicting quality: `Term Representation`

Spearman rank correlations: System vs. WAR (benchmark)

		System			SBERT
		`ldh`	`xdh`	`xdx`	(text)
	count	120	120	120	40
14NS	mean	.333	.336	***.351***	.487
14NS	median	.318	**.344**	.330	.507
50NS	mean	***.350***	*.349*	.348	.509
50NS	median	*.364*	***.374***	*.331*	.523

- SBERT: 50NS > 14NS
- System: for `ldh` & `xdh`: 50NS > 14NS;
  - Model size effect is greater for `ldh`
- System: for `xdx`: NS14 > NS50 (very slight)

Predicting quality: Term Normalization

Response A	Response B	Non-norm -alized weight	Normal -ized weight
The girl is singing	The girl in the cute purple dress is singing a song		
det(the, girl)	det(the, girl)	.143 (2/14)	.175
nsubj(girl, sing)	nsubj(girl, sing)	.143	.175
	det(the, dress)	.071	.050
	amod(cute, dress)	**.071** (1/14)	**.050**
	amod(purple, dress)	**.071**	**.050**
	prep_in(dress, girl)	.071	.050
aux(be, sing)	aux(be, sing)	.143	.175
root(sing, ROOT)	root(sing, ROOT)	.143	.175
	det(a, song)	.071	.050
	dobj(song, sing)	.071	.050
4	10	1.0 (14/14)	1.0

A 2-response toy NS model. Normalizing for response length so each *response* (not *dependency*) in model carries equal weight reduces weight of some extraneous dependencies, but performance suffers overall.