Toward automatic content analysis of L2 English learners' sentences

Levi King
Indiana University

November 3, 2020

Background & Motivation

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*
  - not informed by linguistic expertise, pedagogy, psychometrics

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*
  - not informed by linguistic expertise, pedagogy, psychometrics
- Second Language Acquisition (SLA) research $\rightarrow$ communicative & task based learning

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*
  - not informed by linguistic expertise, pedagogy, psychometrics
- Second Language Acquisition (SLA) research $\rightarrow$ communicative & task based learning

How can we bridge this gap between engineers and SLA researchers?

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*
  - not informed by linguistic expertise, pedagogy, psychometrics
- Second Language Acquisition (SLA) research $\rightarrow$ communicative & task based learning

How can we bridge this gap between engineers and SLA researchers?

- My vision:

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*
  - not informed by linguistic expertise, pedagogy, psychometrics
- Second Language Acquisition (SLA) research → communicative & task based learning

How can we bridge this gap between engineers and SLA researchers?

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

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*
  - not informed by linguistic expertise, pedagogy, psychometrics
- Second Language Acquisition (SLA) research $\rightarrow$ communicative & task based learning

How can we bridge this gap between engineers and SLA researchers?

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

Background & Motivation

- Language tutoring applications (*Rosetta Stone*, *Duo Lingo*, etc.) rely on outdated, ineffective methods:
  - rote memorization & grammatical error detection; menu-based vs. free input;
  - *pure engineering approaches*
  - not informed by linguistic expertise, pedagogy, psychometrics
- Second Language Acquisition (SLA) research → communicative & task based learning

How can we bridge this gap between engineers and SLA researchers?

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

Accomplishments

Accomplishments

Developed basic mechanism:

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine $=$ response score;

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine $=$ response score;
- Optimized system settings:

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine $=$ response score;
- Optimized system settings:
  - item type: intransitive, transitive, ditransitive actions

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine $=$ response score;
- Optimized system settings:
  - item type: intransitive, transitive, ditransitive actions
  - item complexity (type-to-token ratios, avg NS response length)

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine = response score;
- Optimized system settings:
  - item type: intransitive, transitive, ditransitive actions
  - item complexity (type-to-token ratios, avg NS response length)

But first – *all of this required appropriate data!*

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine = response score;
- Optimized system settings:
  - item type: intransitive, transitive, ditransitive actions
  - item complexity (type-to-token ratios, avg NS response length)

But first – *all of this required appropriate data!*

- evaluating my system, finding trends, optimizing;

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine = response score;
- Optimized system settings:
  - item type: intransitive, transitive, ditransitive actions
  - item complexity (type-to-token ratios, avg NS response length)

But first – *all of this required appropriate data!*

- evaluating my system, finding trends, optimizing;
- created task, collected data from 499 participants; $\approxeq$14,000 responses, NS + NNS;

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine $=$ response score;
- Optimized system settings:
  - item type: intransitive, transitive, ditransitive actions
  - item complexity (type-to-token ratios, avg NS response length)

But first – *all of this required appropriate data!*

- evaluating my system, finding trends, optimizing;
- created task, collected data from 499 participants; $\approxeq$14,000 responses, NS $+$ NNS;
- developed & implemented meaning-focused annotation scheme;

Accomplishments

Developed basic mechanism:

- dependency parsing & lemmatization $\rightarrow$ dependency-level tf-idf $\rightarrow$ vectorize NS & NNS responses $\rightarrow$ cosine $=$ response score;
- Optimized system settings:
  - item type: intransitive, transitive, ditransitive actions
  - item complexity (type-to-token ratios, avg NS response length)

But first – *all of this required appropriate data!*

- evaluating my system, finding trends, optimizing;
- created task, collected data from 499 participants; $\approxeq$14,000 responses, NS $+$ NNS;
- developed & implemented meaning-focused annotation scheme;
- established feature weights for scoring & ranking NNS (benchmark);

Data collection

Data collection

I chose to use picture description task (PDT) data because:

Data collection

I chose to use picture description task (PDT) data because:

- Most tutoring applications rely on images;

Data collection

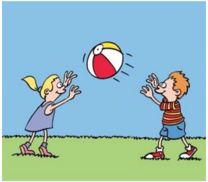
I chose to use picture description task (PDT) data because:

- Most tutoring applications rely on images;
- Simple images constrain the range of likely responses.

Data collection

I chose to use picture description task (PDT) data because:

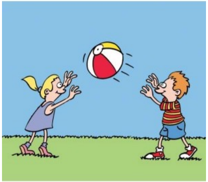
- Most tutoring applications rely on images;
- Simple images constrain the range of likely responses.

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

Data collection

I chose to use picture description task (PDT) data because:

- Most tutoring applications rely on images;
- Simple images constrain the range of likely responses.

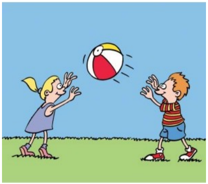
10 intransitive items	10 transitive items	10 ditransitive items
		
What is the girl doing?	What is the boy doing?	What is the boy doing?

- 499 participants: 358 NS (crowdsourced), 141 NNS (ESL students)

Data collection

I chose to use picture description task (PDT) data because:

- Most tutoring applications rely on images;
- Simple images constrain the range of likely responses.

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

- 499 participants: 358 NS (crowdsourced), 141 NNS (ESL students)
- 30 items: Roughly 300 NS responses & 140 NNS responses per item

Task design

Task design

The importance of task design; one example:

What is the man doing?

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.
- The man is **raking** leaves.

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.
- The man is **raking** leaves.

NNS responses:

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.
- The man is **raking** leaves.

NNS responses:

- The man is cleaning the yard.

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.
- The man is **raking** leaves.

NNS responses:

- The man is cleaning the yard.
- He is piling up leaves.

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.
- The man is **raking** leaves.

NNS responses:

- The man is cleaning the yard.
- He is piling up leaves.

This is a coverage problem for my approach.

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.
- The man is **raking** leaves.

NNS responses:

- The man is cleaning the yard.
- He is piling up leaves.

This is a coverage problem for my approach.
Solution?

Task design

The importance of task design; one example:

What is the man doing?

NS responses:

- He's **raking** the leaves.
- The man is **raking** leaves.

NNS responses:

- The man is cleaning the yard.
- He is piling up leaves.

This is a coverage problem for my approach.
Solution? Ask NS for **two** different responses.

Features

Features

Feature requirements:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**: directly answers prompt

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**: directly answers prompt
- **Grammaticality**:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**: directly answers prompt
- **Grammaticality**: no grammar problems

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**: directly answers prompt
- **Grammaticality**: no grammar problems
- **Interpretability**:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**: directly answers prompt
- **Grammaticality**: no grammar problems
- **Interpretability**: evokes clear mental image

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**: directly answers prompt
- **Grammaticality**: no grammar problems
- **Interpretability**: evokes clear mental image
- **Verifiability**:

Features

Feature requirements:

- *reliablity*: consistently annotated by multiple humans;
- *validity*: directly testing for the desired constructs;

First iteration:

- **accuracy** & **native-likeness**: not operationalizable;
- e.g., response is accurate w.r.t. prompt but *adds unverifiable details*; is this *accurate*?

Several iterations later:

- **Core event**: captures main action
- **Answerhood**: directly answers prompt
- **Grammaticality**: no grammar problems
- **Interpretability**: evokes clear mental image
- **Verifiability**: info is supported by image

Feature annotation

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

Table: Annotated for five features: Core event (C), Answerhood (A), Grammaticality (G), Interpretability (I) and Verifiability (V).

Feature reliability

Feature reliability

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

Feature reliability

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	0.733	0.601	0.923	0.808
Answerhood	0.834	0.721	0.982	0.936
Grammaticality	0.861	0.768	0.960	0.827
Interpretability	0.818	0.682	0.919	0.744
Verifiability	0.845	0.719	0.968	0.884

Feature reliability

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	0.733	0.601	0.923	0.808
Answerhood	0.834	0.721	0.982	0.936
Grammaticality	0.861	0.768	0.960	0.827
Interpretability	0.818	0.682	0.919	0.744
Verifiability	0.845	0.719	0.968	0.884
Intransitive	0.863	0.758	0.978	0.910
Transitive	0.780	0.653	0.949	0.853
Ditransitive	0.812	0.678	0.924	0.764

Weighting features

Weighting features

- Features do not contribute equally to response “goodness”

Weighting features

- Features do not contribute equally to response "goodness"
- Raters perform holistic preference test (with annotations hidden)

Weighting features

- Features do not contribute equally to response "goodness"
- Raters perform holistic preference test (with annotations hidden)

What is the boy doing?	Pref?	C	A	G	I	V
He is eating food.	**yes**	**0**	**1**	**1**	**1**	**1**
He may get fat eating.	no	0	0	1	1	0

Weighting features

- Features do not contribute equally to response "goodness"
- Raters perform holistic preference test (with annotations hidden)

What is the boy doing?	Pref?	C	A	G	I	V
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

- Features do not contribute equally to response "goodness"
- Raters perform holistic preference test (with annotations hidden)

What is the boy doing?	Pref?	C	A	G	I	V
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

- Features do not contribute equally to response "goodness"
- Raters perform holistic preference test (with annotations hidden)

What is the boy doing?	Pref?	C	A	G	I	V
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

- Features do not contribute equally to response “goodness”
- Raters perform holistic preference test (with annotations hidden)

What is the boy doing?	Pref?	C	A	G	I	V
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

Preference reliability (feature weights)

Chance Agree	Observed Agree	Kappa
0.621	0.883 (265/300)	0.692

Table: Preference task agreement scores for two annotators on a sample of 300 response pairs; expected chance agreement, observed agreement and Cohen's Kappa.

Weighted annotation scores

Weighted annotation scores

- Calculate weighted annotation score (S_{wa}) for each NNS response;

Weighted annotation scores

- Calculate weighted annotation score (S_{wa}) for each NNS response;
- Rank by S_{wa} ($\rightarrow R_{wa}$); use this as gold standard or benchmark;

Weighted annotation scores

- Calculate weighted annotation score (S_{wa}) for each NNS response;
- Rank by S_{wa} ($\rightarrow R_{wa}$); use this as gold standard or benchmark;
  - Score system output: R_{wa} vs. system ranking $\rightarrow$ Spearman ρ

Weighted annotation scores

- Calculate weighted annotation score (S_{wa}) for each NNS response;
- Rank by S_{wa} ($\rightarrow R_{wa}$); use this as gold standard or benchmark;
  - Score system output: R_{wa} vs. system ranking $\rightarrow$ Spearman ρ

What is the boy doing?	C	A	G	I	V	S_{wa}	R_{wa}
The boy is eating.	0	1	1	1	1	0.635	4
A baby is eating pizza	0	0	1	1	0	0.279	5
The boy enjoys his pizza.	1	0	1	1	1	0.907	2
the boy is eating pizza	1	1	1	1	1	1.0	1
The kid is eats pizza	1	0	0	1	1	0.852	3

Analyzing NNS responses

At this point, my goal is a system that scores and ranks NNS responses via comparison with the crowdsourced NS responses. The system produced ranking should correlate highly with the R_{wa}.

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

Analyzing NNS responses

The system works like this; for each item:

Generate a NS model:

1. dependency parse the collection of NS responses;
2. get tf-idf score for each unique dependency (via a large balanced corpus).

Score each NNS response:

1. As above: dependency parse, tf-idf;
2. Compare NS vs NNS tf-idf vectors: 1 - cosine = response score.

Finally, the NNS responses are ranked by score, and the Spearman rank correlation between R_{wa} and the system is taken as the system configuration score for the item.

By selecting different parameter settings in this approach, I arrive at 12 different system configurations. Each configuration scores and ranks all NNS responses.

System configurations

Consider this simplified set of 2 parameters x 2 settings = 4 configurations.

- **Dependency format**:
  - **labeled**: e.g., nsubj(eat,boy); nobj(eat,pizza)
  - **unlabeled**: e.g., ⟨null⟩(eat,boy); ⟨null⟩(eat,pizza)
- **NS response model**: Note: Each NS participant gave *two* responses per PDT item
  - **first**: Model contains only the first response from NS
  - **mixed**: Model is half first reponses and half second responses

dep\model	first	mixed
labeled	lab_first	lab_mixed
unlabeled	unlab_first	unlab_mixed

System configurations

- Score & rank NNS responses using different configurations;
- Compare with R_{wa} to get a Spearman correlation.

NNS	S_{wa}	R_{wa}	S_{lf}	R_{lf}	S_{uf}	R_{uf}
p1	0.63	4	0.53	4	0.11	5
p2	0.27	5	0.13	5	0.15	4
p3	0.90	2	0.91	1	0.68	1
p4	1.0	1	0.80	2	0.41	2
p5	0.85	3	0.77	3	0.20	3
Spearman ρ			.899		.799	
Spearman p-val			.037		.104	

- *lf* is *labeled_first*; *uf* is *unlabeled_first*:
- *labeled* for labeled dependencies (vs. *unlabeled*)
- *first* for models containing only the first response from NS (vs a *mix* of first and second responses)

Finding trends

From the full set of 360 Spearman correlation scores, I used various subsets of scores to generate hierarchical clusters of the 30 items. I've found some clustering according to verb type (intransitive, transitive or ditransitive), but this trend is not very strong.

I'm currently exploring features other than verb type to see if they correlate with Spearman and can thus help predict optimal system configurations for new items. These features relate to item complexity: type-to-token ratios for the collection of NS responses; mean/median leave-one-out system scores for the NS collection.

Future directions

Other predictive features: image complexity (compression; entropy)

Feedback: Responses scoring below a threshold should be "recast" as the most similar NS response (with $S_{wa} = 1$).
Augment this with SBERT but rely on my system to find appropriate feedback.

For some contexts (e.g., tutoring vs testing), an oracle ranking by the *core event* feature alone may be preferable. Because the feature is binary, Spearman would not be appropriate, so average precision or T-test scores could be used to optimize configurations.

My NNS participants were $>$90% L1 Chinese. I'd love to have an equivalent dataset for another L1 for comparison and to attempt L1-specific optimization.

Generalizing

In a globalized world, we need to be able to analyze speech and text from NNS of English (and other languages). In high stakes contexts, processing should be adaptable, able to abstract from surface form to meaning, and maintain a degree of transparency and explainability.

References

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2014. *Automated Grammatical Error Detection for Language Learners*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, second edition.