On Building Natural Language Interface to Data and Services

Yu Su
Department of Computer Science
University of California, Santa Barbara
Human-machine interface for digitalized world
One interface for all

- Best route to my office. Avoid high way.
- Notify me if the kid watched TV for over one hour when I’m not home.
- How to do list comprehension in Python?
Research on natural language interface

Natural Language Interface

Speech Recognition
- Near human performance

Semantic Parsing
- Sentence level
- Natural language understanding
- Broader domains
- More complex inputs

Dialog System
- Multi-round Dialog
- Dialog management
- Discourse analysis
- (Currently) limited domains, vocabulary, domain complexity, etc.
Semantic parsing

Natural Language → Formal Language

[Zettlemoyer and Collins 2005]
“What is the largest state?” → argmax(\(\lambda x.\text{state}(x)\), \(\lambda x.\lambda y.\text{size}(x,y)\))

[Matuszek et al. 2013]
“Take the left, then next right.” → (do-sequentially (turn-left) (turn-right))

[Berant et al. 2013] [Su et al. 2016]
“How many children of Eddard Stark were born in Winterfell?” → count(\(\lambda x.\text{children}(\text{Eddard_Stark}, x)\) \(\land \text{place_of_birth}(x, \text{Winterfell})\))

[Berant et al. 2013] [Su et al. 2016]
“How many unread emails about PhD application do I have?” → GET-Messages(FILTER(isRead=False), SEARCH(“PhD application”), COUNT())

Table Cell Search for Question Answering (WWW’16)
Improving Semantic Parsing via Answer Type Inference (EMNLP’16)
Cross-domain Semantic Parsing via Paraphrasing (EMNLP’17)
Building Natural Language Interfaces to Web APIs (CIKM’17)
What constitutes a practical NLI?

- Semantic parsing ➡️ most studies
- Scalability ➡️ baby steps
- Interpretability and interactivity
- Robustness (answer triggering, post-inspection, etc.)
Scalability

Vertical scalability
- Scale up to more complex inputs and logical constructs

Who was the head coach when Michael Jordan started playing for the Chicago Bulls?

In which season did Michael Jordan get the most points?

What team did Michael Jordan play for?
Scalability

- **Vertical scalability**
  - Scale up to more complex inputs and logical constructs

- **Horizontal scalability**
  - Scale out to more domains
  - Weather, calendar, hotel, flight, restaurant, ...
  - Knowledge base, relational database, API, robot instruction, ...
  - Graph, table, text, image, audio, ...
Scalability

- Vertical scalability
  - Scale up to more complex inputs and logical constructs

- Horizontal scalability
  - Scale out to more domains
  - Weather, calendar, hotel, flight, restaurant, …
  - Knowledge base, relational database, API, robot instruction, …
  - Graph, table, text, image, audio, …

- More data! Better (more data-efficient) model!

On Generating Characteristic-rich Question Sets for QA Evaluation (EMNLP’16)
Cross-domain Semantic Parsing via Paraphrasing (EMNLP’17)
Building Natural Language Interfaces to Web APIs (CIKM’17)
Interpretability and interactivity

Exploiting Relevance Feedback in Knowledge Graph Search (KDD'15)

Interpretable and Interactive Natural Language Interface via Neural Modular Network (Under review)

Harnessing Predictive Uncertainty for Understanding and Regularizing Neural Predictions (Under review)
Robustness

- Make an NLI more robust without user interaction

- **Answer triggering:** *Is this question answerable by my knowledge sources?*

- **Post-inspection:** *Does the result from semantic parser make sense?*
Outline

☐ Introduction

☐ The cold start problem
   - “I want to build an NLI for my domain, but I don’t have any user and training data”
   - Can I collect training data via crowdsourcing? (EMNLP’16, CIKM’17)
   - Can I leverage existing training data from other domains? (EMNLP’17)

☐ Future work
The cold start problem
“I want to build an NLI for my domain, but I don’t have any user and training data”
Outline

- Introduction

- The cold start problem
  - “I want to build an NLI for my domain, but I don’t have any user and training data”
  - Can I collect training data via crowdsourcing? (EMNLP’16, CIKM’17)
  - Can I leverage existing training data from other domains? (EMNLP’17)

- Future work
Web API: interface to web services
Natural language interface to web API

**Utterance**

Show me unread emails about intern NEO, early ones first

Unread intern NEO emails reverse ordered by time

Find those emails containing intern NEO that I have not read, starting with the oldest one

... ...

**API Frame**

HTTP Verb: GET
Resource: Messages
Return Type: {Message}
Required Parameters: {}
Optional Parameters: {
  FILTER(isRead = False),
  SEARCH("intern NEO"),
  ORDERBY(receivedDateTime, asc)
}

**API Call**

GET https://graph.microsoft.com/v1.0/<user-id>/messages?
$filter=isRead%20eq%20false&
$search="intern%20neo"&
$orderby=receivedDateTime%20asc
Why NLI to API?

- Distributed development of virtual assistants
- Developer assistant: program by dictation
- Enterprise intelligence
- Search, discovery, recommendation of services / APIs
What we did in this work

- The first framework for building an NLI for a given web API starting from zero user and data

- Probabilistic modeling of the crowdsourcing process for NLI

- Optimization of the crowdsourcing process for NLI
  - Select which API calls to annotate?
The data challenge of NLI

“How many unread emails about PhD application do I have?”

GET-Messages{(FILTER(isRead=False), SEARCH("PhD application"), COUNT())}
Low-cost data collection via crowdsourcing

HUGE SPACE!

API Call
GET-Events{SELECT(location), TOP(1)
FILTER(start > now), ORDERBY(start, asc)}

Automatic

Canonical Command
get the location of the top 1 event that start time is after now, and ordered by start time in ascending order

Crowdsourcing

Paraphrases from Crowdsourcing
Where is my next meeting?
What’s the location of my next event?
Find next meeting’s location.
... ...

Wang et al. Building a Semantic Parser Overnight (ACL’15)
Canonical utterance generation: grammar

[Lexicon]

(L1) get → V[GET]
(L2) email → NP[Messages]
(L3) sender → NP[from]
(L4) in the category of → PP/NP[categories]
(L5) read → JJ[isRead]
(L6) have attachment → VP[hasAttachments]
(L7) receive time → NP[receivedDateTime]

... ...

[Boolean Expression]

(B1) VP[x] → VP[x = True]
(B2) do not VP[x] → VP[x = False]
(B3) is JJ[x] → VP[x = True]
(B4) is not JJ[x] → VP[x = False]
(B5) is PP/NP[x] NP[y] → VP[x = y]
(B6) NP[x] is before | after DATETIME[y] → NP[x < | > y]
(B7) NP[x] is | is smaller than | is larger than *[y] → NP[x = | < | > y]

[Query Option]

(Q1) CP[x] → CP[FILTER(x)]
(Q2) that contain NP[x] → CP[SEARCH(x)]
(Q3) ordered by NP[x] in ascending | descending order → CP[ORDERBY(x, ‘asc’ | ‘desc’) ]
(Q4) the NP[x] of → NP/NP[SELECT(x)]
(Q5) the top NUMBER[x] of → NP/NP[TOP(x)]
(Q6) the number of → NP/NP[COUNT()]

[Glue]

(G1) that VP[x] → CP[x]
(G2) CP[x], and CP[y] → CP[x, y]
(G3) NP/NP[x] NP/NP[y] → NP/NP[x, y]
(G4) V[x] NP/NP[w] NP[y] CP[u] → S[x-y{w, u}]
Canonical utterance generation: example
The data challenge (cont’d)

- **Problem**: thousands of API calls can be generated for each API
  - Neither economic nor necessary to annotate all of them
  - *How to optimize the crowdsourcing process?*

- **Solution**: Actively and iteratively select a *subset* of the *most confusing* API calls to annotate

- **Challenge**: How to quantify “confusion”?
Interplay of utterance and API call

- $u$: an utterance, $z$: an API call
- Natural language interface: estimate $p(z|u)$
- Conditional language model: estimate $p(u|z)$

**Definition.** Let $\theta^z: p(u|z)$, $z_1$ and $z_2$ cause confusion if $\theta^{z_1}$ and $\theta^{z_2}$ are similar.
Estimate $\theta^Z$

- If an API call has been annotated via crowdsourcing, use maximum likelihood estimation

**API Call**

```sql
GET-Events{SELECT(location), TOP(1)
FILTER(start > now), ORDERBY(start, asc)}
```

**Paraphrases from Crowdsourcing**

- Where is my next meeting?
- What’s the location of my next event?
- Find next meeting’s location.
- ... ...
Estimate $\theta^z$

- If an API call has been annotated via crowdsourcing, use maximum likelihood estimation

- Otherwise, leverage the compositionality of API calls (or formal languages in general)
How many emails do I have?

GET-Messages{COUNT()}

GET-Messages{FILTER(isRead=False)}

unread emails

GET-Messages{COUNT(), FILTER(isRead=False)}

Yu Su
University of California, Santa Barbara
Compositionality

Assumption:
Utterances follow the composition structure of API calls

Predict the language model of an API call without annotating it!

Crowdsourcing optimization
Semantic mesh: hierarchical probabilistic model for crowdsourcing

Table. Node Operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ANNOTATE</strong></td>
<td>Collect $u$ from crowd, estimate $\theta_{em}$ using MLE</td>
</tr>
<tr>
<td><strong>COMPOSE</strong></td>
<td>$\theta_{ex} = f(\theta^1, \ldots, \theta^n)$</td>
</tr>
<tr>
<td><strong>INTERPOLATE</strong></td>
<td>$\theta = \alpha \cdot \theta_{em} + (1 - \alpha) \cdot \theta_{ex}$</td>
</tr>
</tbody>
</table>
Optimization: differential propagation (DP)

- Objective: maximize pairwise distance $d(z_i, z_j)$

$$w = \min(1.0, \frac{1}{d(z_i, z_j)})$$
Evaluation: data collection via crowdsourcing

- **Input**: the specification of two Microsoft Graph APIs
- **Output**: a set of training data for each API
- **Cost and efficiency**: 82 examples from a worker per hour at a cost of $8.2 on average
- **Quality**: error rate = 17.4% by manual evaluation, on par with related work

<table>
<thead>
<tr>
<th># of API calls</th>
<th># of workers</th>
<th>Avg. time per example</th>
<th>$ per example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1141</td>
<td>201</td>
<td>44s</td>
<td>$0.1</td>
</tr>
</tbody>
</table>

Yu Su  
University of California, Santa Barbara
Utterances **DO** follow composition of API calls

- LM: language model based retrieval model
- # of annotation: ROOT (~20), TOP2 (~100), TOP3 (~500)

<table>
<thead>
<tr>
<th></th>
<th>ROOT</th>
<th>TOP2</th>
<th>TOP3</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM</td>
<td>3.18</td>
<td>12.1</td>
<td>25.5</td>
</tr>
<tr>
<td><strong>GET-Messages</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LM + <strong>SemMesh</strong></td>
<td>46.5</td>
<td>47.8</td>
<td>50.3</td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>5.73</td>
<td>22.9</td>
<td>57.3</td>
</tr>
<tr>
<td><strong>GET-Events</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LM</td>
<td>7.89</td>
<td>13.7</td>
<td>17.4</td>
</tr>
<tr>
<td>LM + <strong>SemMesh</strong></td>
<td>39.0</td>
<td>42.6</td>
<td>44.2</td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>8.95</td>
<td>20.5</td>
<td>45.3</td>
</tr>
</tbody>
</table>

Table 1. Accuracy of different parsing models with different training sets.
Crowdsourcing optimization

- Baseline: Breadth-first annotation (BF)
- Our method: Differential propagation (DP)

(a) GET-Events

(b) GET-Messages
Outline

- Introduction
- The cold start problem
  - “I want to build an NLI for my domain, but I don’t have any user and training data”
  - Can I collect training data via crowdsourcing? (EMNLP’16, CIKM’17)
  - Can I leverage existing training data from other domains? (EMNLP’17)
- Future work
What is *transferrable* in semantic parsing?

In which season did Kobe Bryant play for the Lakers?

\[
R\text{[season].}(\text{player.KobeBryant} \cap \text{team.Lakers})
\]
What is **transferrable** in semantic parsing?

In which season did Kobe Bryant play for the Lakers?

\[ R[\text{season}].(\text{player.KobeBryant} \cap \text{team.Lakers}) \]

\[ p(\text{team} | "\text{play for"}) \]
What is transferrable in semantic parsing?

In which season did Kobe Bryant play for the Lakers?

\[ R[\text{season}].(\text{player.KobeBryant} \land \text{team.Lakers}) \]

When did Alice start working for Mckinsey?

\[ R[\text{start}].(\text{employee.Alice} \land \text{employer.Mckinsey}) \]
What is **transferrable** in semantic parsing?

*In which season did Kobe Bryant play for the Lakers?*

\[ \mathcal{R}[\text{season}].(\text{player.KobeBryant} \land \text{team.Lakers}) \]

*When did Alice start working for Mckinsey?*

\[ \mathcal{R}[\text{start}].(\text{employee.Alice} \land \text{employer.Mckinsey}) \]
Cross-domain semantic parsing via paraphrasing

**Input Utterance**

- *In which seasons did Kobe Bryant play for the Lakers?*

**Canonical Utterance**

- *Season of player Kobe Bryant whose team is Los Angeles Lakers*

**Logical Form**

- \( R[\text{season}].(\text{player.KobeBryant} \land \text{team.Lakers}) \)

---

**Target Domain**

**Paraphrase Model**

- *When did Alice start working for Mckinsey?*

**External Language Resources**

- pre-trained word embeddings, monolingual parallel corpora, ...

**Logical Form**

- \( R[\text{start}].(\text{employee.Alice} \land \text{employer.Mckinsey}) \)
Transfer across domains becomes possible

- Source domain: “play for” ⇒ “whose team is”
- Word embedding: “play” ⇒ “work”, “team” ⇒ “employer”
- Target domain: “work for” ⇒ “whose employer is”
Neural transfer learning for semantic parsing

Source Domain

Target Domain

Pre-trained Word Embedding
Vocabulary shifting

- Only 45%~70% target domain vocabulary are covered by source domains
- Pre-trained word embedding can **potentially** alleviate the vocabulary shifting problem
  - Word2vec: 300-d vectors pre-trained on the 100B-token Google News Corpus; vocabulary size = 3M

<table>
<thead>
<tr>
<th>Metric</th>
<th>CALENDAR</th>
<th>BLOCKS</th>
<th>HOUSING</th>
<th>RESTAURANTS</th>
<th>PUBLICATIONS</th>
<th>RECIPES</th>
<th>SOCIAL</th>
<th>BASKETBALL</th>
</tr>
</thead>
<tbody>
<tr>
<td># of example (N)</td>
<td>837</td>
<td>1995</td>
<td>941</td>
<td>1657</td>
<td>801</td>
<td>1080</td>
<td>4419</td>
<td>1952</td>
</tr>
<tr>
<td># of logical form (</td>
<td>Z</td>
<td>,</td>
<td>C</td>
<td>)</td>
<td>196</td>
<td>469</td>
<td>231</td>
<td>339</td>
</tr>
<tr>
<td>vocab. size (</td>
<td>V</td>
<td>)</td>
<td>228</td>
<td>227</td>
<td>318</td>
<td>342</td>
<td>203</td>
<td>256</td>
</tr>
<tr>
<td>% ∈ other domains</td>
<td>71.1</td>
<td>61.7</td>
<td>60.7</td>
<td>55.8</td>
<td>65.6</td>
<td>71.9</td>
<td>46.0</td>
<td>45.6</td>
</tr>
<tr>
<td>% ∈ WORD2VEC</td>
<td>91.2</td>
<td>91.6</td>
<td>88.4</td>
<td>88.6</td>
<td>91.1</td>
<td>93.8</td>
<td>86.9</td>
<td>86.9</td>
</tr>
<tr>
<td>% ∈ other domains + WORD2VEC</td>
<td>93.9</td>
<td>93.8</td>
<td>90.9</td>
<td>90.4</td>
<td>95.6</td>
<td>97.3</td>
<td>89.3</td>
<td>89.4</td>
</tr>
</tbody>
</table>

Dataset: Overnight, 8 closed-domain KBs

Problems of pre-trained word embedding

- Small *micro variance*: hurt optimization
- Large *macro variance*: hurt generalization

<table>
<thead>
<tr>
<th>Initialization</th>
<th>L2 norm</th>
<th>Variance</th>
<th>Cosine Sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>17.3 ± 0.45</td>
<td>1.00 ± 0.05</td>
<td>0.00 ± 0.06</td>
</tr>
<tr>
<td>WORD2VEC</td>
<td>2.04 ± 1.08</td>
<td>0.02 ± 0.02</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>WORD2VEC + ES</td>
<td>17.3 ± 0.05</td>
<td>1.00 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>WORD2VEC + FS</td>
<td>16.0 ± 8.47</td>
<td>1.09 ± 1.31</td>
<td>0.12 ± 0.10</td>
</tr>
<tr>
<td>WORD2VEC + EN</td>
<td>1.00 ± 0.00</td>
<td>0.01 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
</tbody>
</table>

**Random**: randomly draw from $U(-\sqrt{3}, \sqrt{3})$

**ES**: per-example standardization (per row)

**FS**: per-feature standardization (per column)

**EN**: per-example normalization
Problems of pre-trained word embedding

- Small *micro variance*: hurt optimization
- Large *macro variance*: hurt generalization

<table>
<thead>
<tr>
<th>Initialization</th>
<th>L2 norm</th>
<th>Variance</th>
<th>Cosine Sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>17.3 ± 0.45</td>
<td>1.00 ± 0.05</td>
<td>0.00 ± 0.06</td>
</tr>
<tr>
<td>WORD2VEC</td>
<td>2.04 ± 1.08</td>
<td>0.02 ± 0.02</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>WORD2VEC + ES</td>
<td>17.3 ± 0.05</td>
<td>1.00 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>WORD2VEC + FS</td>
<td>16.0 ± 8.47</td>
<td>1.09 ± 1.31</td>
<td>0.12 ± 0.10</td>
</tr>
<tr>
<td>WORD2VEC + EN</td>
<td>1.00 ± 0.00</td>
<td>0.01 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
</tbody>
</table>

**Random**: randomly draw from $U(-\sqrt{3}, \sqrt{3})$

**ES**: per-example standardization (per row)

**FS**: per-feature standardization (per column)

**EN**: per-example normalization
Problems of pre-trained word embedding

- Small *micro variance*: hurt optimization
- Large *macro variance*: hurt generalization

<table>
<thead>
<tr>
<th>Initialization</th>
<th>L2 norm</th>
<th>Variance</th>
<th>Cosine Sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>17.3 ± 0.45</td>
<td>1.00 ± 0.05</td>
<td>0.00 ± 0.06</td>
</tr>
<tr>
<td><strong>WORD2VEC</strong></td>
<td>2.04 ± 1.08</td>
<td>0.02 ± 0.02</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td><strong>WORD2VEC + ES</strong></td>
<td>17.3 ± 0.05</td>
<td>1.00 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td><strong>WORD2VEC + FS</strong></td>
<td>16.0 ± 8.47</td>
<td>1.09 ± 1.31</td>
<td>0.12 ± 0.10</td>
</tr>
<tr>
<td><strong>WORD2VEC + EN</strong></td>
<td>1.00 ± 0.00</td>
<td>0.01 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
</tbody>
</table>

**Random**: randomly draw from $U(-\sqrt{3}, \sqrt{3})$

**ES**: per-example standardization (per row)

**FS**: per-feature standardization (per column)

**EN**: per-example normalization
Problems of pre-trained word embedding

- Small *micro variance*: hurt optimization
- Large *macro variance*: hurt generalization

<table>
<thead>
<tr>
<th>Initialization</th>
<th>L2 norm</th>
<th>Variance</th>
<th>Cosine Sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>17.3 ± 0.45</td>
<td>1.00 ± 0.05</td>
<td>0.00 ± 0.06</td>
</tr>
<tr>
<td>WORD2VEC</td>
<td>2.04 ± 1.08</td>
<td>0.02 ± 0.02</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>WORD2VEC + ES</td>
<td>17.3 ± 0.05</td>
<td>1.00 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>WORD2VEC + FS</td>
<td>16.0 ± 8.47</td>
<td>1.09 ± 1.31</td>
<td>0.12 ± 0.10</td>
</tr>
<tr>
<td>WORD2VEC + EN</td>
<td>1.00 ± 0.00</td>
<td>0.01 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
</tbody>
</table>

**Random**: randomly draw from $U(-\sqrt{3}, \sqrt{3})$

**ES**: per-example standardization (per row)

**FS**: per-feature standardization (per column)

**EN**: per-example normalization
Problems of pre-trained word embedding

- Small *micro variance*: hurt optimization
- Large *macro variance*: hurt generalization

<table>
<thead>
<tr>
<th>Initialization</th>
<th>L2 norm</th>
<th>Variance</th>
<th>Cosine Sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>17.3 ± 0.45</td>
<td>1.00 ± 0.05</td>
<td>0.00 ± 0.06</td>
</tr>
<tr>
<td>\textsc{word2vec}</td>
<td>2.04 ± 1.08</td>
<td>0.02 ± 0.02</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>\textsc{word2vec + es}</td>
<td>17.3 ± 0.05</td>
<td>1.00 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
<tr>
<td>\textsc{word2vec + fs}</td>
<td>16.0 ± 8.47</td>
<td>1.09 ± 1.31</td>
<td>0.12 ± 0.10</td>
</tr>
<tr>
<td>\textsc{word2vec + en}</td>
<td>1.00 ± 0.00</td>
<td>0.01 ± 0.00</td>
<td>0.13 ± 0.11</td>
</tr>
</tbody>
</table>

**Random**: randomly draw from $U(−\sqrt{3}, \sqrt{3})$  
**ES**: per-example standardization (per row)  
**FS**: per-feature standardization (per column)  
**EN**: per-example normalization
In-domain evaluation

- Wang et al. (2015): 58.8
- Xiao et al. (2016): 72.7
- Jia and Liang (2016): 75.8
- Random: 75.7
- Word2vec: 69.5
- Word2vec+EN: 68.4
- Word2vec+FS: 77.1
- Word2vec+ES: 78.2
Cross-domain evaluation

- Random
- Word2vec
- Word2vec+EN
- Word2vec+FS
- Word2vec+ES

In-domain vs Cross-domain comparison.
Recap

- "I want to build an NLI for my domain, but I don’t have any training data"
- Can I collect training data via crowdsourcing?
  - Yes, and it’s not so expansive
  - Cost can be further reduced by crowdsourcing optimization
- Can I leverage existing training data from other domains?
  - Yes, if you turn it into a paraphrasing problem
  - Pre-trained word embedding can greatly help neural transfer learning, but only when properly standardized
Answer triggering

- **Motivation**: *Most QA systems always output an answer for any question, no matter whether it’s answerable*
- **Definition**: Given a question and a set of answer candidates, determine whether the candidate set contains any correct answer, and if so, select a correct answer as system output

<table>
<thead>
<tr>
<th>Method</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Yang et al., 2015)</td>
<td>32.2</td>
</tr>
<tr>
<td>(Jurczyk et al., 2016)</td>
<td>36.7</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>43.3</strong></td>
</tr>
</tbody>
</table>

Dataset: WikiQA

Yu Su

University of California, Santa Barbara
Post-inspection

- **Motivation:** Before execution, can we check whether the parse result makes sense?
- **Solution:** Cross-check the rational behind the parse result by revising the original question

---

<table>
<thead>
<tr>
<th>Method</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAGG (Yih et al., 2015)</td>
<td>52.5</td>
</tr>
<tr>
<td>(Xu et al., 2016) (SOTA)</td>
<td>53.8</td>
</tr>
<tr>
<td>STAGG + Ours</td>
<td>53.9</td>
</tr>
</tbody>
</table>

Dataset: WebQuestions
Outline

- Introduction
- The cold start problem
  - “I want to build an NLI for my domain, but I don’t have any user and training data”
  - Can I collect training data via crowdsourcing? (EMNLP’16, CIKM’17)
  - Can I leverage existing training data from other domains? (EMNLP’17)
- Future work
One interface for all

- One NLI simultaneously supports all domains
- Incrementally learn new domains without forgetting (or instead boosting) existing ones
Application to important vertical domains

"Which cement stocks go up the most when a Category 3 hurricane hits Florida?"
Programming: the new inequality

We are living in a divided world

programmers

non-programmers

Ultimate Goal
Natural language as a programming language
— Let everyone be a programmer!
Come join us if interested!

- Interdisciplinary research
  - Data mining, natural language processing, machine/deep learning, bioinformatics, programming, literature study, etc.

- Deal with real problems and get your hands dirty

- Always keep an open mind

Xifeng Yan, UCSB

Huan Sun, OSU
Acknowledgements

- Advisor: Xifeng Yan (UCSB)
- Ahmed H. Awadallah, Patrick Pantel, Michael Gamon, and Scott Yih (Microsoft Research)
- Brian Sadler, Michelle Vanni, and Sue Kase (Army Research Lab)
- Mudhakar Srivatsa (IBM Research)
- Ph.D. Committee: Ambuj Singh, Amr El Abbadi (UCSB)
- Many other great collaborators!
On Building Natural Language Interface to Data and Services

Contact Info:
Yu Su
Ph.D. Candidate
Department of Computer Science
University of California, Santa Barbara
ysu@cs.ucsb.edu

Thank you! Questions?