Unsupervised Spoken Language Understanding in Dialogue Systems

YUN-NUNG (VIVIAN) CHEN 陳縕儂
CARNEGIE MELLON UNIVERSITY
2015/01/16 @NCTU

HTTP://VIVIANCHEN.IDV.TW
Outline

Introduction

Unsupervised Slot Induction [Chen et al., ASRU’13 & Chen et al., SLT’14]

Unsupervised Domain Exploration [Chen and Rudnicky, SLT’14]

Unsupervised Relation Detection [Chen et al., SLT’14]

Conclusions & Future Work
Outline

Introduction

Unsupervised Slot Induction [Chen et al., ASRU’13 & Chen et al., SLT’14]

Unsupervised Domain Exploration [Chen and Rudnicky, SLT’14]

Unsupervised Relation Detection [Chen et al., SLT’14]

Conclusions & Future Work
Spoken Language Understanding (SLU)

SLU in dialogue systems

- SLU maps *natural language inputs to semantic forms*
  “I would like to go to NCTU on Friday.”
  
  location: NCTU  
  date: Friday

- Semantic frames, slots, and values
  - often manually defined by domain experts or developers.

What are the problems?
Problems with Predefined Information

**Generalization:** may not generalize to real-world users.

**Bias propagation:** can bias subsequent data collection and annotation.

**Maintenance:** when new data comes in, developers need to start a new round of annotation to analyze the data and update the grammar.

**Efficiency:** time consuming, and high costs.

Can we automatically induce semantic information w/o annotations?
Outline

Introduction

**Unsupervised Slot Induction** [Chen et al., ASRU’13 & Chen et al., SLT’14]

Unsupervised Domain Exploration [Chen and Rudnicky, SLT’14]

Unsupervised Relation Detection [Chen et al., SLT’14]

Conclusions & Future Work
Unsupervised Slot Induction

Motivation

- Spoken dialogue systems (SDS) require predefined semantic slots to parse users’ input into semantic representations
- *Frame semantics theory* provides generic semantics
- *Distributional semantics* capture contextual latent semantics
Probabilistic Frame-Semantic Parsing

**FrameNet** [Baker et al., 1998]
- a linguistically-principled semantic resource, based on the frame-semantics theory.
- “low fat milk” → “milk” evokes the “food” frame; “low fat” fills the descriptor frame element
- **Frame (food):** contains words referring to items of food.
- **Frame Element:** a descriptor indicates the characteristic of food.

**SEMAFOR** [Das et al., 2010; 2013]
- a state-of-the-art frame-semantics parser, trained on manually annotated FrameNet sentences
Step 1: Frame-Semantic Parsing for ASR outputs

**Task:** adapting *generic* frames to *task-specific* settings for SDSs

**can i have a cheap restaurant**

- **Frame:** capability
  - **FT LU:** can
  - **FE LU:** i

- **Frame:** expensiveness
  - **FT LU:** cheap

- **Frame:** locale by use
  - **FT/FE LU:** restaurant

**Good!**
Step 2: Slot Ranking Model

Main Idea

- Ranking domain-specific concepts higher than generic semantic concepts

**can i have a cheap restaurant**

- **Frame: capability**
  - FT LU: can
  - FE LU: i

- **Frame: expensiveness**
  - FT LU: cheap

- **Frame: locale by use**
  - FT/FE LU: restaurant

**slot candidate**

**slot filler**
Step 2: Slot Ranking Model

Rank the slot candidates by integrating two scores

\[ w(s_i) = (1 - \alpha) \log f(s_i) + \alpha \cdot \log h(s_i) \]

- the frequency of the slot candidate in the SEMAFOR-parsed corpus
- the coherence of slot fillers

Slots with higher frequency may be more important.

Domain-specific concepts should focus on fewer topics and be similar to each other.

```
slot: quantity
  a
  three
  all

lower coherence in topic space
```

```
slot: expensiveness
  cheap
  inexpensive
  expensive

higher coherence in topic space
```
Step 2: Slot Ranking Model

Measure coherence by pair-wised similarity of slot fillers

- For each slot candidate $S_i$

$$V(S_i) = \{x_a, x_b, \ldots\}$$

slot candidate: expensiveness corresponding slot filler: “cheap”, “not expensive”

$$h(s_i) = \sum_{x_a, x_b \in V(s_i), x_a \neq x_b} \frac{\text{Sim}(x_a, x_b)}{|V(s_i)|^2}$$

The slot with higher $h(s_i)$ usually focuses on fewer topics, which are more specific, which is preferable for slots of SDS.
Step 2: Slot Ranking Model

How to define the vector for each slot filler?

- Run clustering and then build vectors based on clustering results
  - K-means, spectral clustering, etc.
- Use distributional semantics to transfer words into vectors
  - LSA, PLSA, neural word embeddings (word2vec)
Experiments for Slot Induction

Dataset

- Cambridge University SLU corpus [Henderson, 2012]
  - Restaurant recommendation in an in-car setting in Cambridge
    - WER = 37%
    - vocabulary size = 1868
    - 2,166 dialogues
    - 15,453 utterances
  - dialogue slot: addr, area, food, name, phone, postcode, price range, task, type

The mapping table between induced and reference slots
Experiments for Slot Induction

- **Slot Induction Evaluation**: MAP of the slot ranking model to measure the quality of induced slots via the mapping table.
- **Slot Filling Evaluation**: MAP-F-H/S: weight the MAP score with F-measure of two slot filler lists.

<table>
<thead>
<tr>
<th>Approach</th>
<th>ASR MAP</th>
<th>ASR MAP-F-H</th>
<th>ASR MAP-F-S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Sem</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(a) Frequency</td>
<td>67.61</td>
<td>26.96</td>
<td>27.29</td>
</tr>
<tr>
<td>(b) K-Means</td>
<td>67.38</td>
<td>27.38</td>
<td>27.99</td>
</tr>
<tr>
<td>(c) Spectral Clustering</td>
<td>68.06</td>
<td>30.52</td>
<td>28.40</td>
</tr>
<tr>
<td>Frame Sem + Dist Sem</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(d) Google News RepSim</td>
<td>72.71</td>
<td>31.14</td>
<td>31.44</td>
</tr>
<tr>
<td>(e) NeiSim</td>
<td>73.35</td>
<td><strong>31.44</strong></td>
<td><strong>31.81</strong></td>
</tr>
<tr>
<td>(f) Freebase RepSim</td>
<td>71.48</td>
<td>29.81</td>
<td>30.37</td>
</tr>
<tr>
<td>(g) NeiSim</td>
<td>73.02</td>
<td>30.89</td>
<td>30.72</td>
</tr>
<tr>
<td>(h) (d) + (e) + (f) + (g)</td>
<td><strong>76.22</strong></td>
<td>30.17</td>
<td>30.53</td>
</tr>
</tbody>
</table>
Experiments for Slot Induction

<table>
<thead>
<tr>
<th>Approach</th>
<th>ASR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
</tr>
<tr>
<td><strong>Frame Sem</strong></td>
<td></td>
</tr>
<tr>
<td>(a) Frequency</td>
<td>67.61</td>
</tr>
<tr>
<td>(b) K-Means</td>
<td>67.38</td>
</tr>
<tr>
<td>(c) Spectral Clustering</td>
<td>68.06</td>
</tr>
<tr>
<td><strong>Frame Sem + Dist Sem</strong></td>
<td></td>
</tr>
<tr>
<td>(d) Google News RepSim</td>
<td>72.71</td>
</tr>
<tr>
<td>(e) NeiSim</td>
<td>73.35</td>
</tr>
<tr>
<td>(f) Freebase RepSim</td>
<td>71.48</td>
</tr>
<tr>
<td>(g) NeiSim</td>
<td>73.02</td>
</tr>
<tr>
<td>(h) (d) + (e) + (f) + (g)</td>
<td><strong>76.22</strong></td>
</tr>
</tbody>
</table>

Adding distributional information outperforms our baselines
# Experiments for Slot Induction

<table>
<thead>
<tr>
<th>Approach</th>
<th>ASR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
</tr>
<tr>
<td>Frame Sem</td>
<td></td>
</tr>
<tr>
<td>(a) Frequency</td>
<td>67.61</td>
</tr>
<tr>
<td>(b) K-Means</td>
<td>67.38</td>
</tr>
<tr>
<td>(c) Spectral Clustering</td>
<td>68.06</td>
</tr>
<tr>
<td>Frame Sem + Dist Sem</td>
<td></td>
</tr>
<tr>
<td>(d) Google News</td>
<td>72.71</td>
</tr>
<tr>
<td>(e) NeiSim</td>
<td>73.35</td>
</tr>
<tr>
<td>(f) Freebase</td>
<td>71.48</td>
</tr>
<tr>
<td>(g) NeiSim</td>
<td>73.02</td>
</tr>
<tr>
<td>(h) (d) + (e) + (f) + (g)</td>
<td><strong>76.22</strong></td>
</tr>
</tbody>
</table>

Combining two datasets to integrate the **coverage of Google** and **precision of Freebase** can rank correct slots higher and performs the best MAP scores.
Outline

Introduction

Unsupervised Slot Induction [Chen et al., ASRU’13 & Chen et al., SLT’14]  Question?

Unsupervised Domain Exploration [Chen and Rudnicky, SLT’14]

Unsupervised Relation Detection [Chen et al., SLT’14]

Conclusions & Future Work
Unsupervised Domain Exploration

Target: given conversation interaction with SDS, predicting which application the user wants to launch

Approach:
- Step 1: enriching the semantics using word embeddings
- Step 2: using the descriptions of applications as a retrieval cue to find relevant applications
Proposed Framework

Query Utterance
“play lady gaga’s bad romance”

1. Semantic Seed Generation
   - Wikipedia
   - Freebase
   - Structured Knowledge
   - Frame-Semantic Parsing
   - Entity Linking

2. Semantics Enrichment
   - The Semantic Seeds (Slot Types)
   - Word Embeddings
   - Enrichment Process

3. Retrieval Process
   - Ranking Model
   - Application Data
   - Pandora
   - Ranked Applications

Frame-Semantic Parsing
Entity Linking
Wikipedia
Freebase
Structured Knowledge
Proposed Framework

1. Semantic Seed Generation
   - Wikipedia
   - Freebase
   - Structured Knowledge
   - Frame-Semantic Parsing
   - Entity Linking

2. Semantics Enrichment
   - The Semantic Seeds (Slot Types)
   - Word Embeddings
   - Enrichment Process

3. Retrieval Process
   - Ranking Model
   - Application Data
   - Pandora
   - Ranked Applications

Query Utterance
“play lady gaga’s bad romance”
Semantic Seed Generation

- Main idea: Slot types help imply semantic meaning of the utterance for expanding domain knowledge.
- Frame Type of Semantic Parsing

\[ Q: \text{compose an email to alex} \]

Frame: text creation  Frame: contacting
FT LU: compose  FE LU: an email  FT LU: email

\[ S_{frm}(Q): \text{frame-based semantic seeds} \]

Semantic parsing performs well on a generic domain, and cannot recognize domain-specific named entities.
Semantic Seed Generation

- **Main idea:** Slot types help imply semantic meaning of the utterance for expanding domain knowledge.

- **Entity Type from Linked Structured Knowledge**
  - Wikipedia Page Linking
  - Freebase List Linking

\[
Q: \text{play lady gaga’s bad romance} \quad S_{wk}(Q): \text{wikipedia-based semantic seeds}
\]

\[
Q: \text{play lady gaga’s bad romance} \quad S_{fb}(Q): \text{freebase-based semantic seeds}
\]

... is a song by American singer ...
... is an American singer, songwriter, and actress

\[
\text{composition} \quad \text{canonical version} \quad \text{musical recording}
\]
Proposed Framework

Query Utterance
“play lady gaga’s bad romance”

1. Semantic Seed Generation
   - Wikipedia
   - Freebase
   - Structured Knowledge
   - Frame-Semantic Parsing
   - Entity Linking

2. Semantics Enrichment
   - The Semantic Seeds (Slot Types)
   - singer
   - songwriter
   - song
   - music

3. Retrieval Process
   - Enrichment Process
   - Word Embeddings
   - Ranking Model
   - Application Data
   - Pandora

Ranked Applications
Semantic Enrichment

- **Main idea:** Utilizing distributed word embeddings to obtain the semantically related knowledge of each word.
  1) Modeling word embeddings by the application vendor descriptions.
  2) Extracting the most related words by trained word embeddings for each word. (ex. “text” → “message”, “msg”)

Words with higher similarity suggest that they are often occurs with common contexts in the embedding training data.
Proposed Framework

1. Semantic Seed Generation
   - Frame-Semantic Parsing
   - Entity Linking
   - Wikipedia
   - Freebase
   - Structured Knowledge

2. Semantics Enrichment
   - The Semantic Seeds (Slot Types)
   - Word Embeddings
   - Enrichment Process

3. Retrieval Process
   - Ranking Model
   - Application Data
   - Pandora
   - Ranked Applications

Query Utterance
“play lady gaga’s bad romance”
Retrieval Process

- **Main idea:** Retrieving the applications that are more likely to support users’ requests.

- **Query Reformulation** ($Q'$)
  - Embedding-Enriched Query: integrates similar words to all words in $Q$
  - Type-Embedding-Enriched Query: additionally adds similar words to semantic seeds $S(Q)$

- **Ranking Model**

  \[
  P(Q | A) = \frac{1}{|Q'|} \sum_{x \in Q'} \log P(x | A)
  \]

  probability that user speaks $Q$ to make the request for launching the application $A$
  probability that word $x$ occurs in the application

  The application with higher $P(Q | A)$ is more likely to be able to support the user desired functions.
Results

Baseline
Type-Embedding-Enriched: Frame (T)
Type-Embedding-Enriched: Freebase (T)
Embedding-Enriched (T)
Type-Embedding-Enriched: Wikipedia (T)
Type-Embedding-Enriched: Freebase (T)
Type-Embedding-Enriched: Hand-crafted (T)
Overall Results

Tune the thresholds by develop set

<table>
<thead>
<tr>
<th>Approach</th>
<th>ASR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
</tr>
<tr>
<td>Original Query</td>
<td>25.50</td>
</tr>
<tr>
<td>Embedding-Enriched</td>
<td>30.42</td>
</tr>
<tr>
<td>Frame</td>
<td>30.11</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>30.74</td>
</tr>
<tr>
<td>Freebase</td>
<td><strong>32.02</strong></td>
</tr>
<tr>
<td>Hand-Craft</td>
<td>34.91</td>
</tr>
</tbody>
</table>

- Enriching semantics improves performance by involving domain-specific knowledge.
- **Freebase** results are better than the embedding-enriched method, showing that we can effectively and efficiently expand domain-specific knowledge by types of slots from Freebase.
- Hand-crafted mapping shows that the **correct types of slots offer better understanding** and tells the room of improvement.
Outline

Introduction

Unsupervised Slot Induction [Chen et al., ASRU’13 & Chen et al., SLT’14]

Unsupervised Domain Exploration [Chen and Rudnicky, SLT’14]

Unsupervised Relation Detection [Chen et al., SLT’14]

Conclusions & Future Work
Unsupervised Relation Detection

**Spoken Language Understanding (SLU):** convert ASR outputs into pre-defined semantic output format

“when was james cameron’s avatar released”

Intent: FIND_RELEASE_DATE
Slot-Val: MOVIE_NAME=“avatar”, DIRECTOR_NAME=“james cameron”

**Relation:** semantic interpretation of input utterances
- movie.release_date, movie.name, movie.directed_by, director.name

**Unsupervised SLU:** utilize external knowledge to help relation detection without labelled data
Semantic Knowledge Graph

Priors for SLU

What are knowledge graphs?

- Graphs with
  - strongly typed and uniquely identified entities (nodes)
  - facts/literals connected by relations (edge)

Examples:

- Satori, Google KG, Facebook Open Graph, Freebase

How large?

- > 500M entities, >1.5B relations, > 5B facts

How broad?


Semantic Interpretation via Relations

Two Examples

- differentiate two examples by including the originating node types in the relation

User Utterance:
find movies produced by james cameron

SPARQL Query (simplified):

Logical Form:
\[ \lambda x. \exists y. \text{movie.produced_by}(x, y) \land \text{person.name}(y, z) \land z=\text{"James Cameron"} \]

Relation:
\text{movie.produced_by} \leftrightarrow \text{producer.name}

User Utterance:
who produced avatar

SPARQL Query (simplified):
SELECT ?producer {?movie.name"Avatar". ?movie.produced_by?producer.}

Logical Form:
\[ \lambda y. \exists x. \text{movie.produced_by}(x, y) \land \text{movie.name}(x, z) \land z=\text{"Avatar"} \]

Relation:
\text{movie.name} \leftrightarrow \text{movie.produced_by}
Proposed Framework

“find me some films directed by james cameron”

Input Utterance

Background Knowledge

Entity Dict. → Relation Inference from Gazetteers → Knowledge Graph Entity

Probabilistic Enrichment $R_u(r)$

Boostrapping

Final Results

Local Relational Surface Form

Bing Query Snippets → Entity Embeddings → Relational Surface Form Derivation → Entity Surface Forms

Relabel

Entity Syntactic Contexts

Proposed Framework

"find me some films directed by james cameron"

Input Utterance

Background Knowledge

Entity Dict. → Relation Inference from Gazetteers → Knowledge Graph Entity

Probabilistic Enrichment $R_u(r)$

Boostrapping

Final Results

Local Relational Surface Form

Bing Query Snippets → Entity Embeddings → Relational Surface Form Derivation → Entity Surface Forms

Relabel
Proposed Framework

“find me some films directed by james cameron”

Input Utterance

Background Knowledge

Entity Dict.

Relation Inference from Gazetteers

$P_E(r \mid w)$

Knowledge Graph Entity

Probabilistic Enrichment $R_u(r)$

Boostraping

Local Relational Surface Form

Bing Query Snippets

Entity Embeddings

Relational Surface Form Derivation

$P_F(r \mid w)$

Entity Surface Forms

$P_C(r \mid w)$

Entity Syntactic Contexts

Relabel

Final Results
Relation Inference from Gazetteers

Gazetteers (entity lists)

\[ P_E(t_i | w) = \frac{C(w, t_i)}{\sum_{t_k \in T(w)} C(w, t_k)} \]

\[ P_E(r_i | w) = P_E(t_i | w) \]

Proposed Framework

“find me some films directed by james cameron”

Input Utterance

Background Knowledge

Entity Dict. → Relation Inference from Gazetteers → \( P_E(r \mid w) \)

Knowledge Graph Entity → Probabilistic Enrichment \( R_u(r) \)

Boostraping → Relabel → Final Results

Local Relational Surface Form

Bing Query Snippets → Entity Embeddings → Relational Surface Form Derivation

Entity Surface Forms → \( P_F(r \mid w) \)

Entity Syntactic Contexts → \( P_C(r \mid w) \)
Bing query snippets including entity pairs connected with specific relations in KG

**Avatar** is a 2009 American epic science fiction film directed by **James Cameron**.

**Dependency Parsing**

- **nsubj**: Avatar
- **cop**: is
- **det**: a
- **num**: 2009
- **amod**: American
- **nmod**: epic
- **nmod**: science
- **nmod**: fiction
- **xcomp**: film
- **prep**: directed
- **pobj**: by
- **nsubj**: James Cameron
- **nmod**: directed
- **pobj**: by

**Directed by**
Relational Surface Form Derivation
Dependency-Based Entity Embeddings

1) Word & Context Extraction

<table>
<thead>
<tr>
<th>Word</th>
<th>Contexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>$movie</td>
<td>film/nsub⁻¹</td>
</tr>
<tr>
<td>is</td>
<td>film/cop⁻¹</td>
</tr>
<tr>
<td>a</td>
<td>film/det⁻¹</td>
</tr>
<tr>
<td>2009</td>
<td>film/num⁻¹</td>
</tr>
<tr>
<td>american, epic, science, fiction</td>
<td>film/nn⁻¹</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Word</th>
<th>Contexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>film</td>
<td>film/nsub, is/cop, a/det, 2009/num, american/nn, epic/nn, science/nn, fiction/nn, directed/vmod</td>
</tr>
<tr>
<td>directed</td>
<td>$director/prep_by</td>
</tr>
<tr>
<td>$director</td>
<td>directed/prep_by⁻¹</td>
</tr>
</tbody>
</table>
Relational Surface Form Derivation
Dependency-Based Entity Embeddings

2) Training Process
   ◦ Each word $w$ is associated with a vector $v_w$ and each context $c$ is
     represented as a vector $v_c$
   ◦ Learn vector representations for both words and contexts such that the
     dot product $v_w \cdot v_c$ associated with good word-context pairs belonging to
     the training data $D$ is maximized
   ◦ Objective function:

\[
\arg \max_{v_w, v_c} \sum_{(w, c) \in D} \log \frac{1}{1 + \exp(-v_c \cdot v_w)}
\]

<table>
<thead>
<tr>
<th>Word</th>
<th>Contexts</th>
<th>Word</th>
<th>Contexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>$movie$</td>
<td>film/nsub$^{-1}$</td>
<td>film</td>
<td>film/nsub, is/cop, a/det, 2009/num, american/nн, epic/nн, science/nн, fiction/nн, directed/vmod</td>
</tr>
<tr>
<td>is</td>
<td>film/cop$^{-1}$</td>
<td>2009</td>
<td>directed</td>
</tr>
<tr>
<td>a</td>
<td>film/det$^{-1}$</td>
<td>$director$</td>
<td>$director/prep_by$</td>
</tr>
<tr>
<td>2009</td>
<td>film/num$^{-1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>american, epic,</td>
<td>film/nn$^{-1}$</td>
<td>directed</td>
<td>directed/prep_by$^{-1}$</td>
</tr>
<tr>
<td>science, fiction</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Relational Surface Form Derivation

Surface Form Derivation

Entity Surface Forms

- Learn the surface forms corresponding to entities

\[
S^F_i(w_j) = \frac{\text{sim}(w_j, e_i)}{\sum_{e_k \in E} \text{sim}(w_j, e_k)}
\]

\[
P^F_i(r_i | w_j)
\]

Based on word vector $v_w$

Entity Syntactic Contexts

- Learn the important contexts of entities

\[
S^C_i(w_j) = \frac{\text{sim}(\hat{w}_j, e_i)}{\sum_{e_k \in E} \text{sim}(\hat{w}_j, e_k)}
\]

\[
P^C_i(r_i | w_j)
\]

Based on context vector $v_c$

- $\text{$char$, $director$, etc.}$
  - $\text{$char$: “character”, “role”, “who”}$
  - $\text{$director$: “director”, “filmmaker”}$
  - $\text{$genre$: “action”, “fiction”}$

- With similar contexts

- Frequently occurring together
Proposed Framework

"find me some films directed by james cameron"

Input Utterance

Background Knowledge
- Entity Dict.
- Relation Inference from Gazetteers
- Knowledge Graph Entity
- Probabilistic Enrichment $R_u(r)$
- Boostraping
- Relabel
- Final Results

Local Relational Surface Form
- Bing Query Snippets
- Entity Embeddings
- Relational Surface Form Derivation
- Entity Surface Forms
- Entity Syntactic Contexts

Entity Dict.

Knowledge Graph
Probabilistic Enrichment

Integrate relations from
- Prior knowledge $P_E(r \mid w)$
- Entity surface forms $P_F(r \mid w)$
- Entity syntactic contexts $P_C(r \mid w)$

Integrated Relations for Words by
- **Unweighted**: combine all relations with binary values
- **Weighted**: combine all relations and keep the highest weights of relations
- **Highest Weighted**: combine the most possible relation of each word

<table>
<thead>
<tr>
<th>$r$</th>
<th>actor</th>
<th>produced_by</th>
<th>location</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_E(r \mid w)$</td>
<td>0.7</td>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>$P_F(r \mid w)$</td>
<td>0.4</td>
<td>0</td>
<td>0.6</td>
</tr>
<tr>
<td>$P_C(r \mid w)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Unweighted $R_w(r)$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Weighted $R_w(r)$</td>
<td>0.7</td>
<td>0.3</td>
<td>0.6</td>
</tr>
<tr>
<td>Highest Weighted $R_w(r)$</td>
<td>0.7</td>
<td>0</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Integrated Relations for Utterances by

$$R_u(r_i) = \max_{w \in u} R_w(r_i)$$

Boostrapping
Unsupervised Self-Training

Training a multi-label multi-class classifier estimating relations given an utterance

Utterances with relation weights

R_{u1}(r) \quad R_{u2}(r) \quad R_{u3}(r)

creating labels by a threshold

\[ u_1: \text{L}_{u1}(r) \]
\[ u_2: \text{L}_{u2}(r) \]
\[ u_3: \text{L}_{u3}(r) \]

Adaboost: ensemble M weak classifiers

Classifier

output prob dist. of relations

Pseudo labels for training
Experiments of Relation Detection

Dataset

Knowledge Base: Freebase
- 670K entities
- 78 entity types (movie names, actors, etc)

Relation Detection Data
- Crowd-sourced utterances
- Manually annotated with SPARQL queries → relations

<table>
<thead>
<tr>
<th>Query Statistics</th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>% entity only</td>
<td>8.9%</td>
<td>10.7%</td>
</tr>
<tr>
<td>% rel only w/ specified movie names</td>
<td>27.1%</td>
<td>27.5%</td>
</tr>
<tr>
<td>% rel only w/ specified other names</td>
<td>39.8%</td>
<td>39.6%</td>
</tr>
<tr>
<td>% more complicated relations</td>
<td>15.4%</td>
<td>14.7%</td>
</tr>
<tr>
<td>% not covered</td>
<td>8.8%</td>
<td>7.6%</td>
</tr>
<tr>
<td>#utterances</td>
<td>3338</td>
<td>1084</td>
</tr>
</tbody>
</table>
### Experiments of Relation Detection

**All performance**

**Evaluation Metric:** micro F-measure (%)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Unweighted</th>
<th>Weighted</th>
<th>Highest Weighted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Ori Bootstrap</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gazetteer</td>
<td>35.21</td>
<td>36.91</td>
<td>37.93</td>
</tr>
<tr>
<td></td>
<td>37.93</td>
<td>40.10</td>
<td>40.10</td>
</tr>
<tr>
<td></td>
<td>36.08</td>
<td>38.89</td>
<td></td>
</tr>
<tr>
<td><strong>Gazetteer + Weakly Supervised</strong></td>
<td>25.07</td>
<td>37.39</td>
<td>39.04</td>
</tr>
<tr>
<td></td>
<td>39.07</td>
<td>39.40</td>
<td>39.98</td>
</tr>
<tr>
<td><strong>Gazetteer + Entity Surface Form (Reg)</strong></td>
<td>34.23</td>
<td>34.91</td>
<td>36.57</td>
</tr>
<tr>
<td></td>
<td>38.13</td>
<td>34.69</td>
<td>37.16</td>
</tr>
</tbody>
</table>

**Baseline**
Experiments of Relation Detection

All performance

Evaluation Metric: micro F-measure (%)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Unweighted</th>
<th>Weighted</th>
<th>Highest Weighted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ori</td>
<td>Bootstrap</td>
<td>Ori</td>
</tr>
<tr>
<td>Gazetteer</td>
<td>35.21</td>
<td>36.91</td>
<td>37.93</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Reg)</td>
<td>34.23</td>
<td>34.91</td>
<td>36.57</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Dep)</td>
<td>37.44</td>
<td>38.37</td>
<td>41.01</td>
</tr>
</tbody>
</table>

Words derived by dependency embeddings can successfully capture the surface forms of entity tags, while words derived by regular embeddings cannot.
Experiments of Relation Detection
All performance

Evaluation Metric: micro F-measure (%)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Unweighted</th>
<th></th>
<th>Weighted</th>
<th></th>
<th>Highest Weighted</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Ori</td>
<td>Bootstrap</td>
<td>Ori</td>
<td>Bootstrap</td>
<td>Ori</td>
</tr>
<tr>
<td>Gazetteeer</td>
<td>35.21</td>
<td>36.91</td>
<td>37.93</td>
<td>40.10</td>
<td>36.08</td>
<td>38.89</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Reg)</td>
<td>34.23</td>
<td>34.91</td>
<td>36.57</td>
<td>38.13</td>
<td>34.69</td>
<td>37.16</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Dep)</td>
<td>37.44</td>
<td>38.37</td>
<td>41.01</td>
<td>41.10</td>
<td>39.19</td>
<td>42.74</td>
</tr>
<tr>
<td>Gazetteer + Entity Context</td>
<td>35.31</td>
<td>37.23</td>
<td>38.04</td>
<td>38.88</td>
<td>37.25</td>
<td>38.04</td>
</tr>
</tbody>
</table>

Words derived from entity contexts slightly improve performance.
## Experiments of Relation Detection

### All performance

**Evaluation Metric: micro F-measure (%)**

<table>
<thead>
<tr>
<th>Approach</th>
<th>Unweighted</th>
<th>Weighted</th>
<th>Highest Weighted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ori</td>
<td>Bootstrap</td>
<td>Ori</td>
</tr>
<tr>
<td>Gazetteer</td>
<td>35.21</td>
<td>36.91</td>
<td>37.93</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Reg)</td>
<td>34.23</td>
<td>34.91</td>
<td>36.57</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Dep)</td>
<td>37.44</td>
<td>38.37</td>
<td>41.01</td>
</tr>
<tr>
<td>Gazetteer + Entity Context</td>
<td>35.31</td>
<td>37.23</td>
<td>38.04</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form + Context</td>
<td>37.66</td>
<td>38.64</td>
<td>40.29</td>
</tr>
</tbody>
</table>

Combining all approaches performs best, while the major improvement is from derived entity surface forms.
# Experiments of Relation Detection

All performance

**Evaluation Metric: micro F-measure (%)**

| Approach | Unweighted | | Weighted | | | | Highest Weighted | |
|----------|-----------|----------|-----------|----------|----------|-----------------|----------|
|          |          |          | Ori       | Bootstrap | Ori       | Bootstrap | Ori       | Bootstrap |
| Gazetteer| 35.21     | 36.91    | 37.93     | 40.10     | 36.08     | 38.89     |           |           |
| Gazetteer + Entity Surface Form (Reg) | 34.23 | 34.91 | 36.57 | 38.13 | 34.69 | 37.16 | |
| Gazetteer + Entity Surface Form (Dep) | 37.44 | 38.37 | 41.01 | 41.10 | 39.19 | 42.74 | |
| Gazetteer + Entity Context | 35.31 | 37.23 | 38.04 | 38.88 | 37.25 | 38.04 | |
| Gazetteer + Entity Surface Form + Context | 37.66 | 38.64 | 40.29 | 41.98 | 40.07 | 43.34 | |

With the same information, learning surface forms from dependency-based embedding performs better, because there’s mismatch between written and spoken language.
Experiments of Relation Detection
All performance

Evaluation Metric: micro F-measure (%)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Unweighted</th>
<th>Weighted</th>
<th>Highest Weighted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ori</td>
<td>Bootstrap</td>
<td>Ori</td>
</tr>
<tr>
<td>Gazetteer</td>
<td>35.21</td>
<td>36.91</td>
<td>37.93</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Reg)</td>
<td>34.23</td>
<td>34.91</td>
<td>36.57</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form (Dep)</td>
<td>37.44</td>
<td>38.37</td>
<td>41.01</td>
</tr>
<tr>
<td>Gazetteer + Entity Context</td>
<td>35.31</td>
<td>37.23</td>
<td>38.04</td>
</tr>
<tr>
<td>Gazetteer + Entity Surface Form + Context</td>
<td>37.66</td>
<td>38.64</td>
<td>40.29</td>
</tr>
</tbody>
</table>

Weighted methods perform better when less features, and highest weighted methods perform better when more features.
Experiments of Relation Detection
Entity Surface Forms Derived from Dependency Embeddings

The functional similarity carried by dependency-based entity embeddings effectively benefits relation detection task.

<table>
<thead>
<tr>
<th>Entity Tag</th>
<th>Derived Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>$character</td>
<td>character, role, who, girl, she, he, officier</td>
</tr>
<tr>
<td>$director</td>
<td>director, dir, filmmaker</td>
</tr>
<tr>
<td>$genre</td>
<td>comedy, drama, fantasy, cartoon, horror, sci</td>
</tr>
<tr>
<td>$language</td>
<td>language, spanish, english, german</td>
</tr>
<tr>
<td>$producer</td>
<td>producer, filmmaker, screenwriter</td>
</tr>
</tbody>
</table>
Experiments of Relation Detection

Effectiveness of Boosting

- The best result is the combination of all approaches, because probabilities came from different resources can complement each other.
- Only adding entity surface forms performs similarly, showing that the major improvement comes from relational entity surface forms.
- Boosting significantly improves most performance.
Outline

Introduction

Unsupervised Slot Induction [Chen et al., ASRU’13 & Chen et al., SLT’14]

Unsupervised Domain Exploration [Chen and Rudnicky, SLT’14]

Unsupervised Relation Detection [Chen et al., SLT’14]  Question?

Conclusions & Future Work
Conclusions & Future Work

Conclusions
◦ Unsupervised SLU are more and more popular.
◦ Using external knowledge helps SLU in different ways.
◦ Word embeddings is very useful

Future Work
◦ Fusion of various knowledge resources
  ◦ Different resources help SLU in different ways
◦ Relation between slots
  ◦ Understanding Inter-slot relations can help develop better SDS
◦ Active learning
  ◦ In terms of practical and efficiency, manually labeling a small set of samples can boost performance.
Q & A 😊

THANKS FOR YOUR ATTENTIONS!!