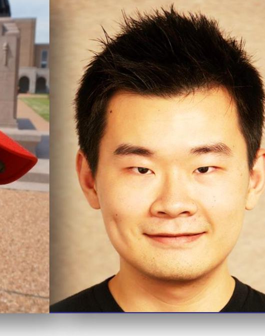
LEVERAGING FRMAE SEMANTICS AND DISTRIBUTIONAL SEMANTICS FOR UNSUPERVISED SEMANTIC SLOT INDUCTION IN SPOKEN DIALOGUE SYSTEMS

Carnegie Mellon

Yun-Nung (Vivian) Chen, William Yang Wang, and Alexander I. Rudnicky


1. The Task

- Motivations
  - SDSs require predefined semantic slots to support parsing users' input into semantic representations
  - Frame semantics theory provides generic semantics
  - Distributional semantics capture contextual latent semantics

> Given a collection of unlabeled raw speech audio, can we automatically induce and fill the semantic slots in an unsupervised manner?

- Approaches
  - Generate slot candidates using a frame-semantic parser
  - Propose a re-ranking model to differentiate domain-specific concepts from generic semantic concepts using word representations
- Results
  - Automatically induced semantic slots have average precision (AP) of 76% for ASR-transcribed data

2. Probabilistic Frame-Semantic Parsing

- Frame Semantics: the meaning of most words can be expressed as semantic frames.
- SEMAFOR is a state-of-the-art parser for frame semantic parsing, trained on a linguistically-principled semantic resource, FrameNet.

> How to adapt the FrameNet-style frame-semantic parses to the semantic slots in the target semantic space for practical use in spoken dialogue systems?

- We parse ASR-decoded utterances using SEMAFOR and extract all frames from semantic parsing results as slot candidates; LUs that correspond to the frames are used for slot-filling.
  - Slot candidate
  - Slot filler

can i have a cheap restaurant

can i	cheap	restaurant
Frame: capability	Frame: expensiveness	Frame: locale by use
FT LU: can FE Filler: i	FT LU: cheap	FT/FE LU: restaurant
generic semantic concept (useless for SDS)	domain-specific concept	domain-specific concept

3. Word Representations

- Distributional Semantics hypothesizes that words occurring in the same contexts may have similar meanings.
- Recurrent neural network language models use the context history to include long-distance information, capturing both syntactic and semantic regularities.

> Allows the use of large external data sets to differentiate semantic concepts and help the adaptation process

- Pre-trained distributed word embedding vectors:
  - Word/Phrase Vectors from Google News
  - Entity Vectors with Freebase Naming

4. Slot Ranking Model

- **Main idea: rank domain-specific concepts higher than generic semantic concepts**
- Rank the slot candidates by integrating two scores

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

  - $f(s_i)$: the frequency of each candidate slot in the parsed corpus

> slots with higher frequency may be more important

  - $h(s_i)$: the coherence of values the slot corresponds to

> domain-specific concepts should focus on fewer topics and be similar to each other, so that coherence can help measure the prominence of the slots.

a, one, three, all — lower coherence in topic space — slot: quantity

cheap, inexpensive, expensive — higher coherence in topic space — slot: expensiveness

- Measure coherence by pair-wised similarity of slot-fillers
  - For each slot s_i, $V(\boxed{s_i}) = \{x_a, \boxed{x_b}, \ldots\}$

slot candidate: expensiveness

corresponding slot filler: "cheap", "not expensive" (from the utterances with s_i in the parsing results)

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

> The slot with higher $h(s_i)$ usually focuses on fewer topics, which are more specific, which is preferable for slots of SDS.

① Representation-derived similarity
- For each word x, it can be represented as a word embedding vector $\mathbf{x}$
- Compute the similarity as cosine similarity between their embedding vectors

$$\mathrm{RepSim}(x_a, x_b) = cosine(\mathbf{x_a}, \mathbf{x_b})$$

> Words occurring in similar domains have similar word representations
> RepSim will be larger when x_a, x_b are semantically related

- Complexity: size of vocabulary

② Neighbor-derived similarity
- For each word x, we build a neighbor-derived sparse vector

$$\mathbf{r}_x = [r_x(1), \ldots, \boxed{r_x(t)}, \ldots]$$

cosine similarity between x and t-th word in the dict. if it is the top N nearest neighbors of x in embedding vector space

- Compute the similarity as cosine similarity between their vectors

$$\mathrm{NeiSim}(x_a, x_b) = cosine(\mathbf{r}_{x_a}, \mathbf{r}_{x_b})$$

> Words with similar concepts share similar neighbors
> NeiSim will be larger when x_a, x_b have more overlapped neighbors in continuous space

- Complexity: number of neighbors

5. Experiments

- **Domain**: restaurant recommendation in an in-car setting in Cambridge (Word Error Rate = 37%)
  - Dialogue slots: addr, area, food, phone, postcode, price range, task, and type
- **Slot Induction Evaluation**: AP of the slot ranking model used to measure the quality of induced slots via the mapping table
- **Slot Filling Evaluation**: AP-F-H/S*: weight the AP score with F-measure of two slot filler lists

* F-H (Hard): the values of two slot fillers are exactly the same
F-S (Soft): the values of two slot fillers both contain at least one overlapping words

Induced slots	Reference slot
locale by use, building	type
food, origin	food
speak on topic	addr
seeking, desiring, locating	task
commerce scenario, expensiveness, range	price range
part orientational, direction, locale, part inner outer	area
contacting	phone
sending	postcode

The mapping table between induced and reference slots

> Adding distributional information outperforms our baselines

> Neighbor-derived similarity performs better because it considers more semantically-related words (instead of two tokens)
> Neighbor-derived similarity requires less space for computational procedure

Approach				ASR AP	ASR AP-F-H	ASR AP-F-S
Frame Sem	(a)	Frequency		67.61	26.96	27.29
	(b)	K-Means		67.38	27.38	27.99
	(c)	Spectral Clustering		68.06	30.52	28.40
Frame Sem + Dist Sem	(d)	Google News	RepSim	72.71	31.14	31.44
	(e)		NeiSim	73.35	**31.44**	**31.81**
	(f)	Freebase	RepSim	71.48	29.81	30.37
	(g)		NeiSim	73.02	30.89	30.72
	(h)	(d) + (f)		74.60	29.82	30.31
	(i)	(e) + (g)		74.34	31.01	31.28
	(j)	(d) + (e) + (f) + (g)		**76.22**	30.17	30.53

> Combining two datasets to integrate the coverage of Google and precision of Freebase can rank correct slots higher and provides the best MAP scores

6. Conclusions

- We propose an **unsupervised** approach unifying **frame and distributional semantics** for automatic induction and filling of semantic slots.
- Our work makes use of a state-of-the-art semantic parser, and adapts the linguistically principled generic FrameNet-style outputs to the target semantic space that corresponds to a domain-specific SDS setting.
- With the incorporation of word embeddings, our experiments show that automatically induced semantic slots align well with the reference slots created by domain experts.
- Automatically induced semantic slots achieve **AP of 76%** for ASR-transcribed data

yvchen@cs.cmu.edu vivian.ynchen