DYNAMICALLY SUPPORTING UNEXPLORED DOMAINS IN CONVERSATIONAL INTERACTIONS BY ENRICHING SEMANTICS WITH NEURAL WORD EMBEDDINGS

Yun-Nung (Vivian) Chen and Alexander I. Rudnicky

Carnegie Mellon

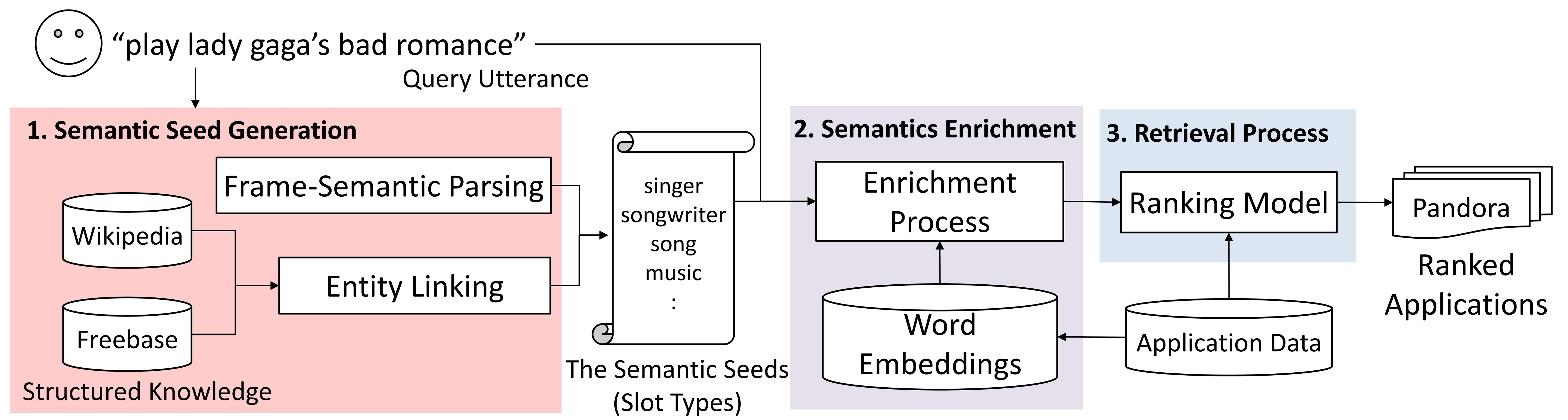
1. The Task

- Motivations
  - A typical SDS needs a predefined task domain that supports specific functionality; it is not able to dynamically support functions provided by newly installed or not yet installed apps.
  - Structured knowledge resources are available (e.g. Freebase, Wikipedia, FrameNet) and may provide semantic information that allows new functionality to be linked into the domain.
  - Neural word embeddings can provide semantic knowledge via unsupervised training.

> In an open domain, with spoken queries, how can we dynamically and effectively provide the corresponding functions to fulfill users' requests?

- Approaches
  1. Generating semantic seeds by using knowledge resources
  2. Enriching the semantics with neural word embeddings
  3. Retrieving relevant applications or dynamically suggesting users install the applications that support new domain functionality.
- Results
  - Compared to original queries, using the **Freebase knowledge resource** (sufficient information about **named entities**) to extract slot types for enriching semantics of queries achieves 25% and 18% relative improvement of MAP and P@5 respectively.

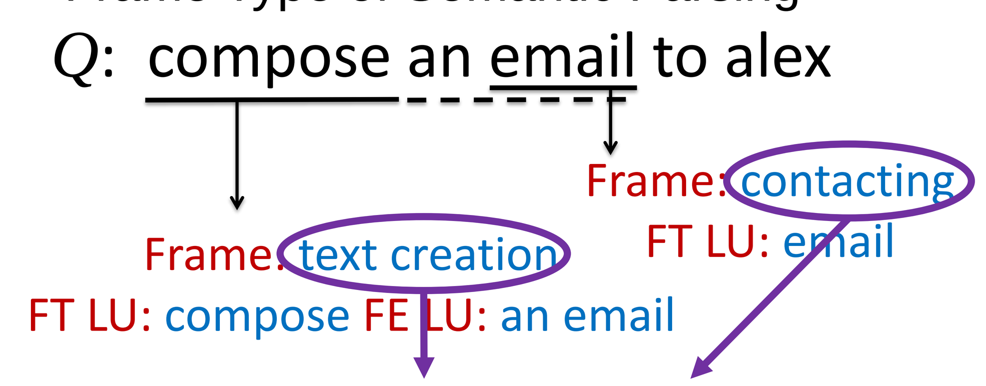
2. Framework



3. Semantic Seed Generation

- **Main idea: Types of slots help determine semantic meaning of the utterance for expanding domain knowledge.**

- Frame Type of Semantic Parsing



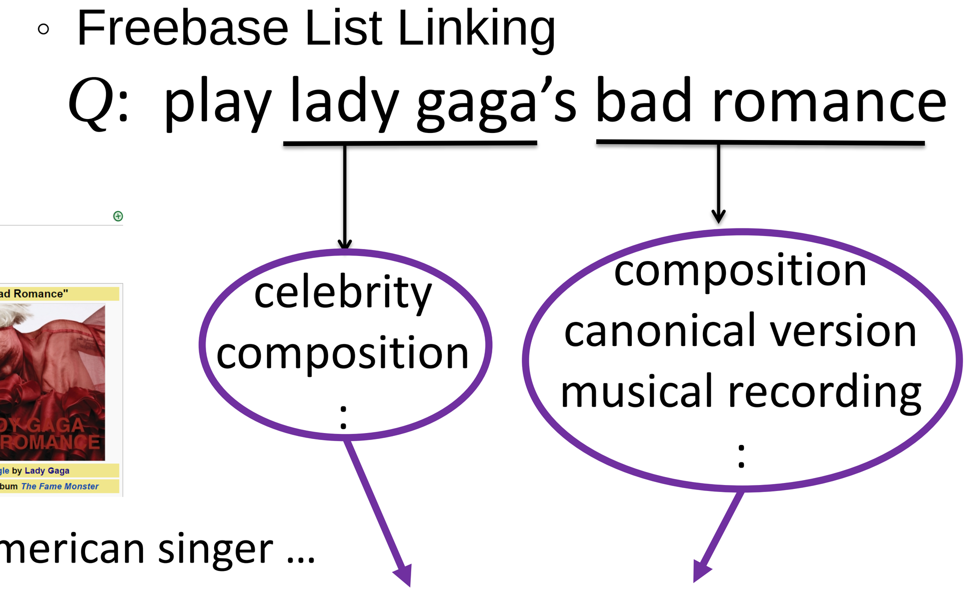
$S_{frm}(Q)$: frame-based semantic seeds

> Semantic parsing performs well on a generic domain, but cannot recognize domain-specific named entities.

- Entity Type from Linked Structured Knowledge
  - Wikipedia Page Linking

$S_{wk}(Q)$: wikipedia-based semantic seeds

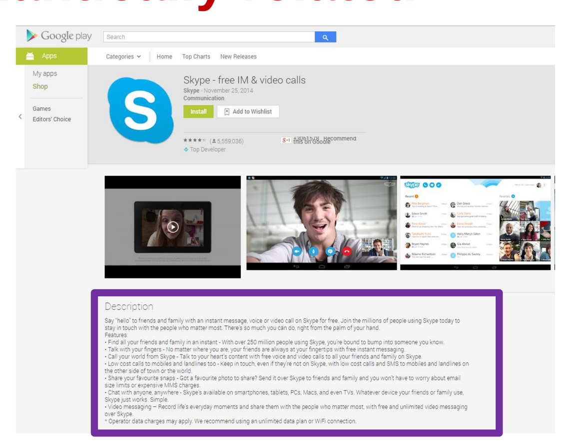
  - Freebase List Linking



$S_{fb}(Q)$: freebase-based semantic seeds

4. Semantics Enrichment

- **Main idea: Use distributed word embeddings to obtain the semantically related knowledge for each word.**
  1) Model word embeddings by using application vender descriptions.
  2) Extract the most related words by trained word embeddings for each semantic seed. "text" → "message", "msg"



> Words with higher similarity suggest that they often occur with common contexts in the embedding training data.

5. Retrieval Process

- **Main idea: retrieve the applications that are more likely to support users' requests via vender descriptions**
- 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

$$\boxed{P(Q \mid A)} = \frac{1}{|Q'|} \sum_{x \in Q'} \log \boxed{P(x \mid A)}$$

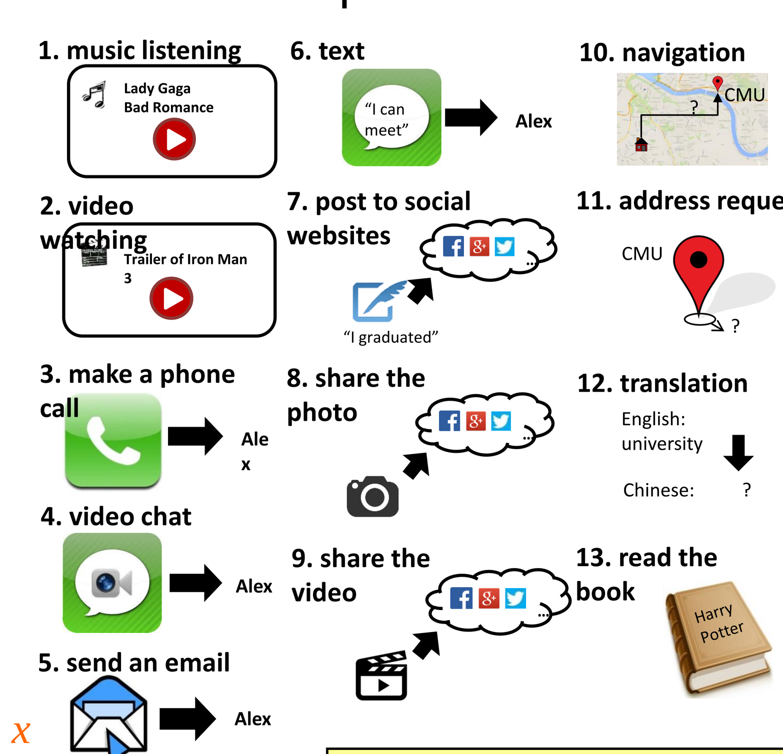
probability that user speaks Q to make the request for launching the application A

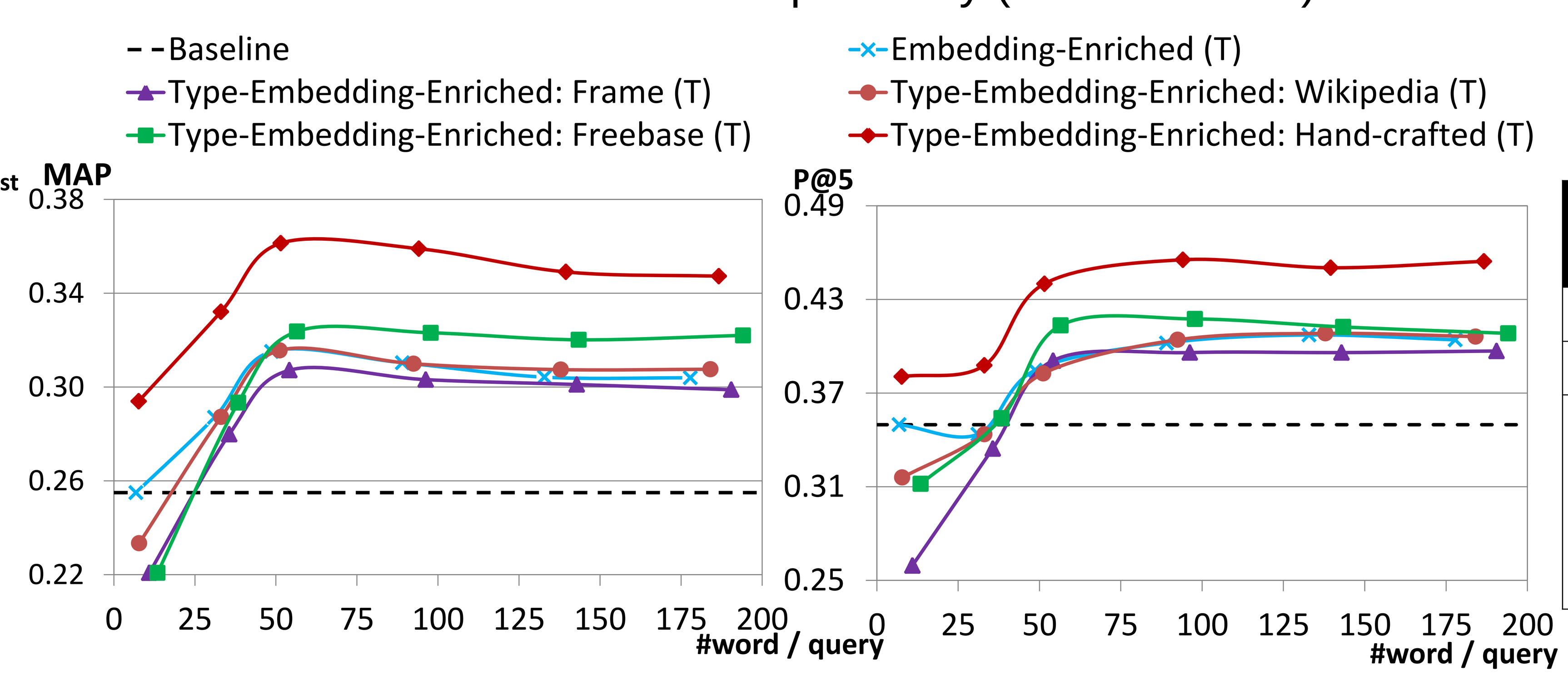
probability that word x from Q' occurs in the application A

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

6. Experiments

- **Domain**: 13 important application types, accessed the most frequently from Google Play
  - Speech data collected from the users with intents described pictorially (WER = 19.8%)





Approach		ASR	
		MAP	P@5
Original Query		25.50	34.97
Embedding-Enriched		30.42	40.72
Type-Embed.-Enriched	Frame	30.11	39.59
	Wikipedia	30.74	40.82
	Freebase	**32.02**	**41.23**
	Hand-Craft	34.91	45.03

> Enriching semantics improves performance by involving domain-specific knowledge.
>
> Freebase results are better than the embedding-enriched method when |Q'| > 50, especially for P@5, 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.

7. Conclusions

- We propose an **unsupervised** approach for acquiring **open domain knowledge** based on a user's verbal request.
- We use **structured knowledge** to extract **slot types** as **semantic seeds** to obtain domain-related information, and retrieve more the most relevant applications without supervision.
- We enable the system to **properly react** to users' queries, such as providing relevant applications or suggesting users install applications that support uncovered domains, based on their open domain requests.

yvchen@cs.cmu.edu vivian.ynchen