

知识问答

大纲

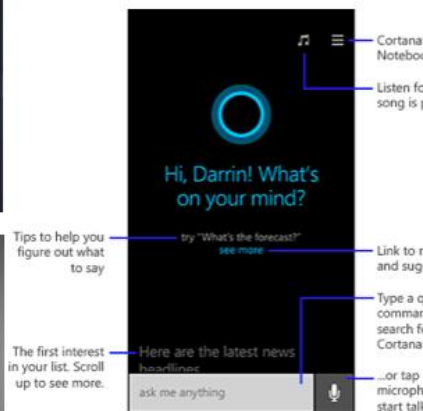
- 知识问答概述和相关数据集
- KBQA基本概念及挑战
- 知识问答主流方法介绍

大纲

- 知识问答概述和相关数据集
- KBQA基本概念及挑战
- 知识问答主流方法介绍

交互方法的转变

- 需要信息服务模式的转变
- 移动互联网以及可穿戴设备的飞速发展需要有效、准确的自然语言形式的信息服务方式



问答系统

□ 是下一代搜索引擎的基本形态



Prof. Oren Etzioni

Turing Center
University of Washington

以直接而准确的方式
回答用户自然语言提
问的自动问答系统将
构成下一代搜索引擎
的基本形态

— 《Nature》2011.8

IBM Watson

- 沃森 (Watson): 2011年, IBM研发的超级计算机“沃森”在美国知识竞赛节目《危险边缘Jeopardy!》中上演“人机问答大战”, 战胜人类选手Ken和Brad



辅助医疗



金融辅助决策



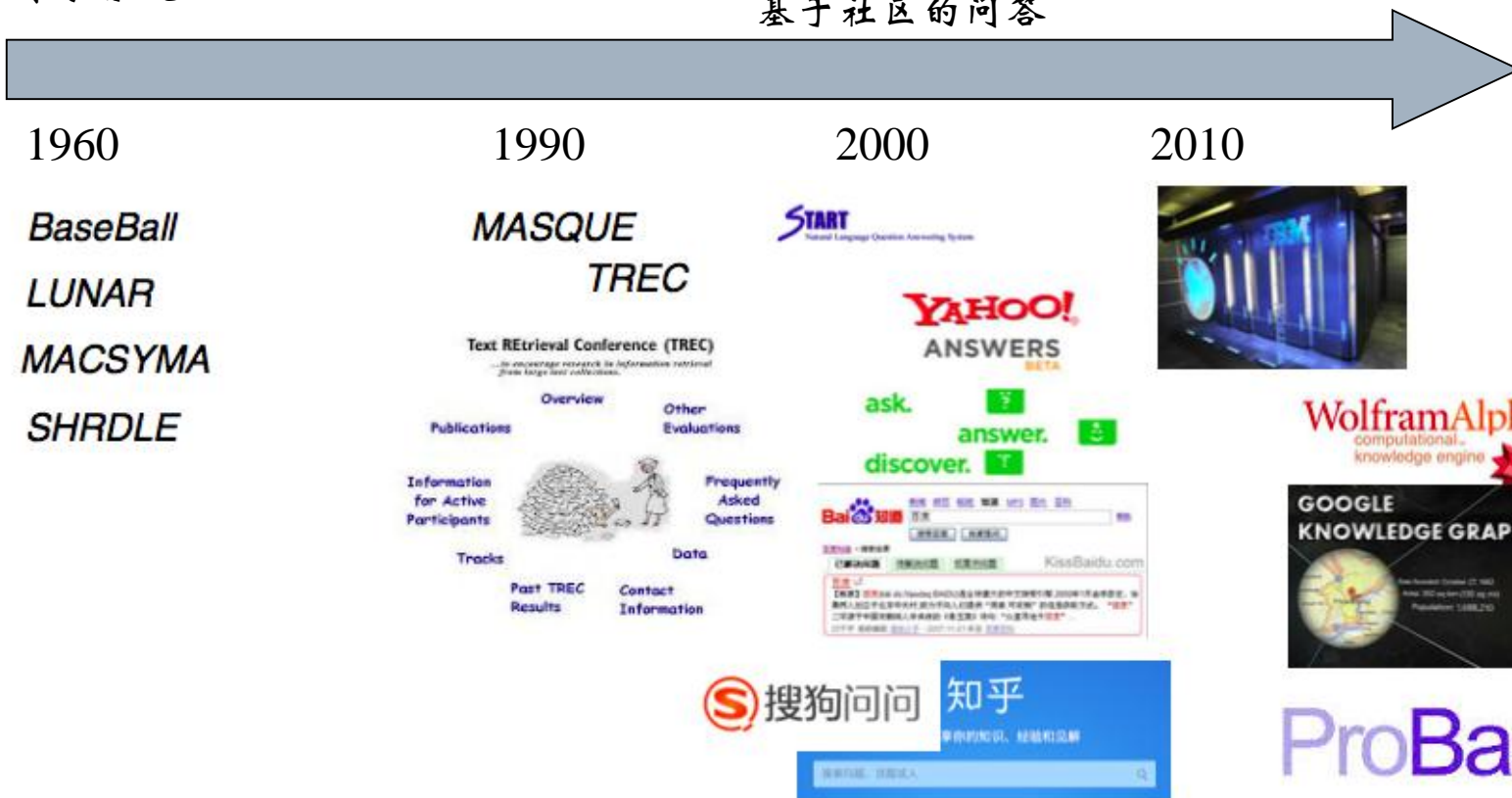
企业服务

问答系统历史

基于模板的问答
专家系统

基于信息检索的问答

基于知识库的问答
基于社区的问答



问答系统历史

基于信息检索的问答

基于关键词匹配+信息抽取，基于浅层语义分析



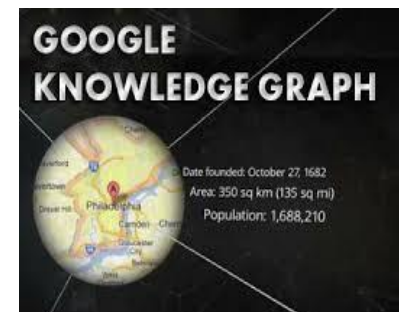
基于社区的问答

依赖于网民贡献，问答过程依赖于关键词检索技术



基于知识库的问答

知识库
语义解析



根据问答形式分类

□ 一问一答



□ 交互式问答




□ 阅读理解

Mary journeyed to the den.
Mary went back to the kitchen.
John journeyed to the bedroom.
Mary discarded the milk.
Q: Where was the milk before the den?
A. Hallway



Brian is a lion.
Julius is a lion.
Julius is white.
Bernhard is green.
Q: What color is Brian?
A. White




Sam walks into the kitchen.
Sam picks up an apple.
Sam walks into the bedroom.
Sam drops the apple.
Q: Where is the apple?
A. Bedroom

KBQA应用



ned poulter



+Ned   Share 

Web Images Maps Videos More ▾ Search tools



About 155,000 results (0.48 seconds)

Cookies help us deliver our services. By using our services, you agree to our use of cookies.

OK [Learn more](#)

Ned Poulter



www.nedpoulter.com/ ▾

by Ned Poulter - in 1,122 Google+ circles
Geek, Blogger & Digital Marketer currently working as the SEO Manager for Miinto. Check out my blogs and contact me here.

Ned Poulter (NedPoulter) on Twitter

<https://twitter.com/NedPoulter> ▾

The latest from **Ned Poulter** (@NedPoulter). Geek, Film Buff, Blogger, Social Butterfly, Aspiring Indiana Jones. Lover of SEO & Online Marketing. Copenhagen ...

Ned Poulter - Danmark | LinkedIn

dk.linkedin.com/in/nedpoulter/ ▾

Copenhagen Area, Denmark - Head of Digital at Miinto Group
Vis Ned Poulter's (Danmark) faglige profil på LinkedIn. LinkedIn er verdens største erhvervsnetværk, der hjælper fagfolk som **Ned Poulter** med at finde interne ...

Ned Poulter - author on State of Digital

www.stateofdigital.com/author/ned-poulter/ ▾

by Ned Poulter - in 1,122 Google+ circles
Ned Poulter is an author on State of Digital and SEO Manager at Miinto. Find his articles and bio here.



Ned Poulter

1,126 followers on Google+

Copenhagen based SEO, Blogging & Digital Marketing enthusiast who thrives on everything geeky. I love traveling and never stop learning....

Contact info

From Ned Poulter's profile

Email 

Personal email

From Google Contacts

- Visible only to you

Alternative emails

Email 

Phone 

Mobile phone number

Address 

Home address

Recent posts



Fantastic Visualisation of The Marketing Technology Landscape in 2014 [Infographic]. Well Worth Saving + Sharing 9 Jan 2014

KBQA 应用



全部 新闻 图片 视频 地图 更多 ▾ 搜索工具

找到约 21,400,000 条结果 (用时 0.62 秒)

NBA总决赛 (2016 年)

克利夫兰骑士



球员名单和概述

反馈

[2016 NBA Playoffs | Schedules, Brackets, Highlights and Video ...](#)

www.nba.com/playoffs/ ▾ 翻译此页

Find playoff schedules, brackets of your favorite teams and video highlights of all the **2016 NBA Playoffs** games on NBA.com.



Enter what you want to calculate or know about:

how big is China



Examples Random

Assuming "how big" is international data | Use as [referring to socioeconomic data](#) or [referring to species](#) or [referring to administrative divisions](#) instead

Assuming total area | Use [population](#) instead

Input interpretation:

China total area

Result:

Show non-metric

$9.597 \times 10^6 \text{ km}^2$ (square kilometers) (world rank: 4th)

Unit conversions:

$9.597 \times 10^{12} \text{ m}^2$ (square meters)

3.705 million mi² (square miles)

$1.033 \times 10^{14} \text{ ft}^2$ (square feet)

Comparisons as area:

$\approx 0.96 \times \text{total area of Canada}$ ($9.98467 \times 10^6 \text{ km}^2$)

$\approx 0.996 \times \text{total area of the United States}$ ($9.63142 \times 10^6 \text{ km}^2$)

$\approx \text{largest extent of the Roman Empire}$ ($\approx 9 \text{ Mm}^2$)

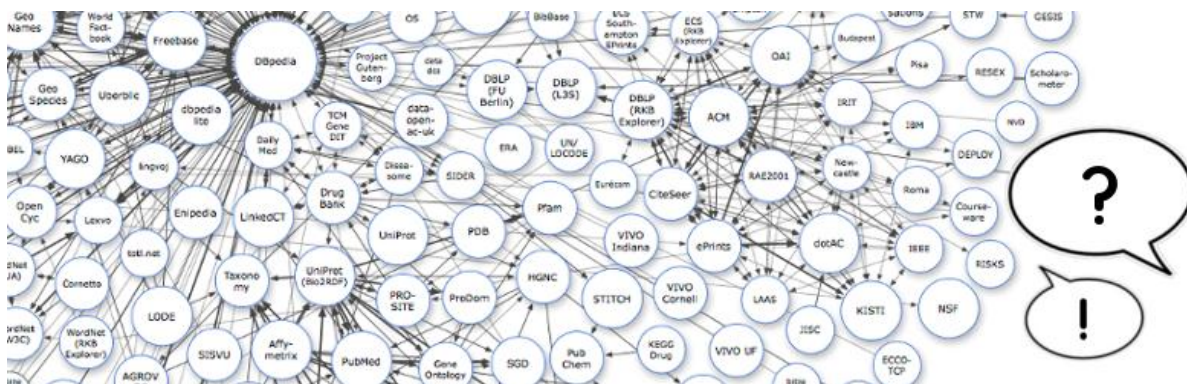
答题机器人

- ❑ “多合一”机器人:日本富士通联合日本国立信息学研究所的高考机器人项目 (2011)
- ❑ 微软创始人Paul Allen与华盛顿大学计划制造能够通过高中生物考试的系统
- ❑ 863: 基于大数据的类人智能关键技术与系统



测评数据集 - QALD

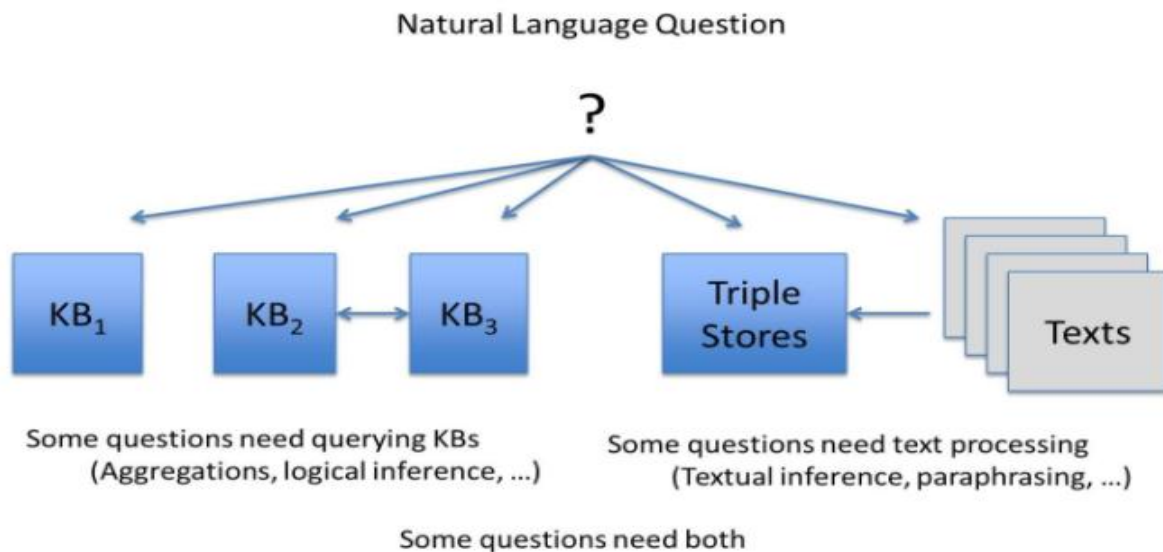
- ❑ 知识库问答评测，Question Answering over Linked Data (QALD); 是多语言的链接数据问答 (multilingual question answering over linked data) 系统的评测竞赛活动。
- ❑ 数据来源包括Dbpedia、YAGO和MusicBrainz;
- ❑ ESWC workshop上开展，旨在建立一个统一的测评基准。每年100个问题左右，从2011年开始，到目前共举办了7届;



测评数据集 - QALD

□ 主要任务有三类

- 多语种问答, 基于Dbpedia
- 问答基于链接数据(interlinked data)
- Hybrid QA, 基于RDF and free text data



QALD任务1

□ 多语种问答

Given a natural language question or keywords, either retrieve the correct answer(s) from a given RDF repository, or provide a SPARQL query that retrieves these answer(s).

Dataset: DBpedia 3.9
(with multilingual labels)

Questions:
200 training + 50 test

Seven languages:
—English, Spanish, German,
Italian, French, Dutch, Romanian

```
<question id = "36" answertype = "resource"
          aggregation = "false"
          onlydbo = "true" >
```

- Through which countries does the Yenisei river flow?
- Durch welche Länder fließt der Yenisei?
- Por qué países fluye el río Yenisei?
- ...

```
PREFIX res: <http://dbpedia.org/resource/>
```

```
PREFIX dbo: <http://dbpedia.org/ontology/>
```

```
SELECT DISTINCT ?uri WHERE {
    res:Yenisei_River dbo:country ?uri .
}
```


QALD任务2

- 问答基于链接数据 (interlinked data)
- 数据集:SIDER, Diseasome, Drugbank
- 问题:25 training + 25 test
- 从不同数据集融合信息
- Example: What is the side effects of drugs used for Tuberculosis?

```
SELECT DISTINCT ?x WHERE {  
  disease:1154 diseasome:possibleDrug ?v2 .  
  ?v2 a drugbank:drugs .  
  ?v3 owl:sameAs ?v2 .  
  ?v3 sider:sideEffect ?x .  
}
```

QALD任务3

- **Hybrid QA**
- 数据集: DBpedia 3.9 (with English abstracts)
- 问题: 25 training + 10 test
- require both structured data and free text from the abstract to be answered
- Example: Give me the currencies of all G8 countries.

```
SELECT DISTINCT ?uri WHERE {  
  ?x text:"member of" text:"G8" .  
  ?x dbo:currency ?uri .  
}
```

QALD的评测指标

□ 采用准确率P，召回率R和F值作为测评指标

question q , precision, recall and F-measure were computed as follows:

$$Recall(q) = \frac{\text{number of correct system answers for } q}{\text{number of gold standard answers for } q}$$

$$Precision(q) = \frac{\text{number of correct system answers for } q}{\text{number of system answers for } q}$$

$$F\text{-Measure}(q) = \frac{2 * Precision(q) \times Recall(q)}{Precision(q) + Recall(q)}$$

评测数据集

□ WebQuestions

- 使用Freebase，通过Google Suggest API爬取得到候选问题，最终得到5,810个问题（3,778个训练数据，2,032个测试数据）
- 利用Amazon Mechanical Turk服务得到答案（在Freebase中可以找到），一个问题可能存在多个答案，利用Average F1评价
- WebQuestionsSP是WebQuestions包含SPARQL标注的升级版本，其中4737个问题包含semantic parses和对应的SPARQL查询语句，是“answerable”的，剩余问题被标注为：“not answerable”

□ Free917

- 同样使用Freebase，共917个问题，包含641个训练样例，276个测试样例

Examples

1. What are the neighborhoods in New York City?
 $\lambda x. \text{neighborhoods}(\text{new_york}, x)$
2. How many countries use the rupee?
 $\text{count}(x). \text{countries_used}(\text{rupee}, x)$
3. How many Peabody Award winners are there?
 $\text{count}(x). \exists y. \text{award_honor}(y) \wedge$
 $\text{award_winner}(y, x) \wedge$
 $\text{award}(y, \text{peabody_award})$

大纲

- 知识问答概述和相关数据集
- KBQA基本概念及挑战
- 知识问答主流方法介绍

知识问答简单流程和分类

问句



语义匹配、推理

答案

- 传统问答方法(符号表示)

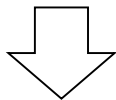
- 基于关键词检索
- 基于文本蕴含推理
- 基于逻辑表达式

- 基于深度学习的问答方法
(分布式表示)

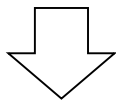
- LSTM
- Attention Model
- Memory Network

基于符号表示(传统)的知识库问答

姚明的老婆的国籍是?

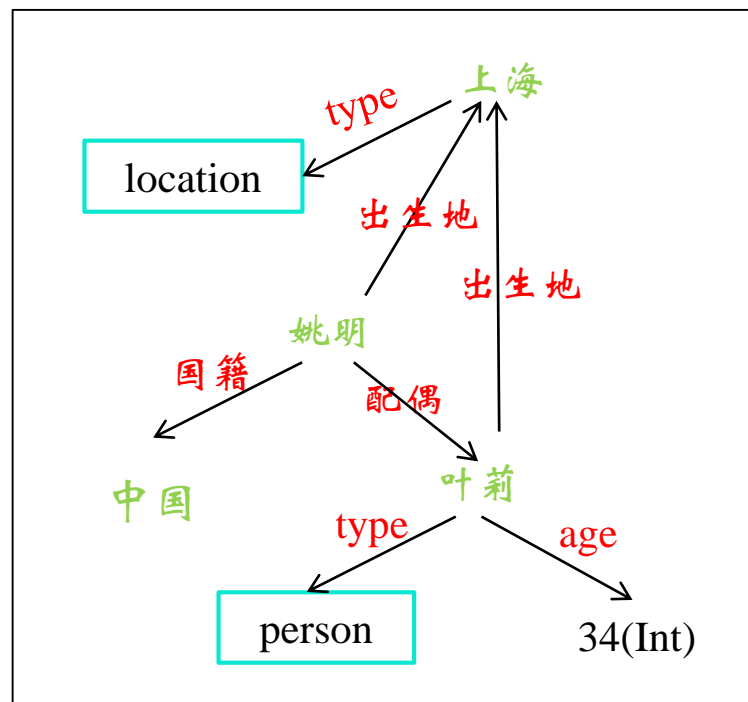
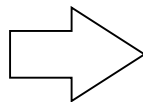


语义解析

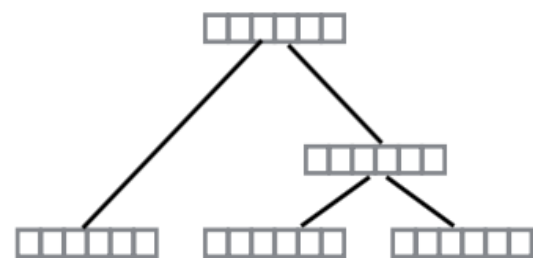


```
SELECT DISTINCT ?X  
WHERE{  
  ?y 国籍 ?x.  
  res:姚明 配偶 ?y.  
}
```

查询

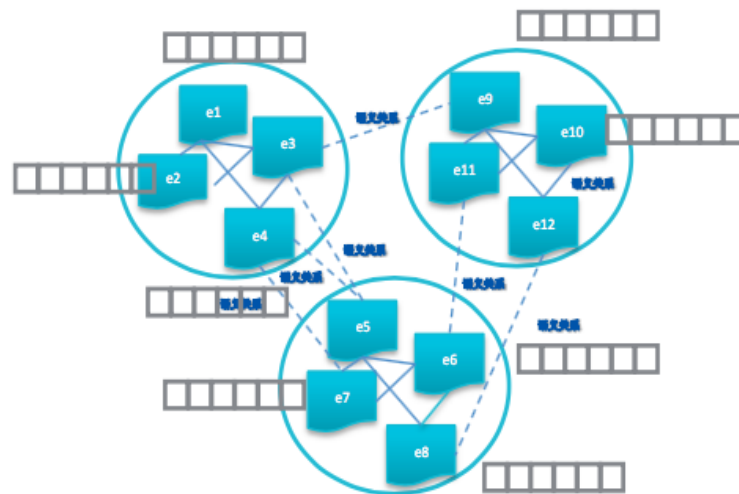


基于分布式表示 (DL) 的知识库问答



姚明的老婆的国籍是？

Similarity



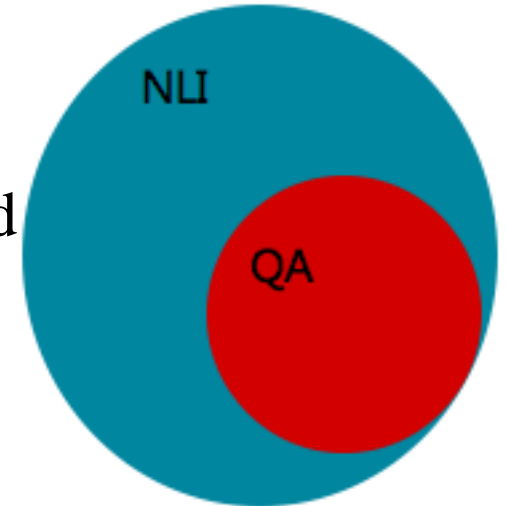
NLI & QAs

□ Natural Language Interfaces (NLI)

■ 输入: Natural language queries

■ 输出: Either processed or unprocessed answers

- Processed: Direct answers (QA).
- Unprocessed: Database records, text snippets, documents.



基本概念与术语

- Categorization of questions and answers.
- Important for:
 - Understanding the challenges before attacking the problem.
 - Scoping the QA system.
- Based on:
 - Chin-Yew Lin: Question Answering.
 - Farah Benamara: Question Answering Systems: State of the Art and Future Directions.

概念一：问句短语

☐ 问句短语定义问的是什么：

■ Wh-words:

☐ who, what, which, when, where, why, and how

■ Wh-words + nouns, adjectives or adverbs:

☐ “which party ...”, “which actress ...”, “how long ...”, “how tall ...”.

概念二：问题类型

□ 问题类型决定了后续采用什么样的回答处理策略

■ **FACTOID – 事实型问题：**

□ **PREDICATIVE QUESTIONS – 谓词型问题：**

- “Who was the first man in space?”
- “What is the highest mountain in Korea?”
- “How far is Earth from Mars?”
- “When did the Jurassic Period end?”
- “Where is Taj Mahal?”

□ **LIST – 列表型问题：**

- “Give me all cities in Germany.”

□ **SUPERLATIVE – 最高级型问题：**

- “What is the highest mountain?”

□ **YES-NO – 对错型问题：**

- “Was Margaret Thatcher a chemist?”
-

概念二:问题类型

- 问题类型决定了后续采用什么样的回答处理策略
 - **OPINION - 观点型问题:**
 - “What do most Americans think of gun control?”
 - **CAUSE & EFFECT-因果型问题:**
 - “What is the most frequent cause for lung cancer?”
 - **PROCESS – 方法型问题:**
 - “How do I make a cheese cake?”
 - **EXPLANATION & JUSTIFICATION – 解释型问题:**
 - “Why did the revenue of IBM drop?”
 - **ASSOCIATION QUESTION – 关联型问题:**
 - “What is the connection between Barack Obama and Indonesia?”
 - **EVALUATIVE OR COMPARATIVE QUESTIONS – 比较型问题:**
 - “What is the difference between impressionism and expressionism?”

概念三：答案类型

The class of object sought by the question:

☐ **Abbreviation**

☐ **Entity:** event, color, animal, plant, . . .

☐ **Description, Explanation & Justification :** definition, manner, reason, . . . ("How, why ...")

☐ **Human:** group, individual, . . . ("Who ...")

☐ **Location:** city, country, mountain, . . . ("Where ...")

☐ **Numeric:** count, distance, size, . . . ("How many how far, how long ...")

☐ **Temporal:** date, time, ...(from "When ...")

概念四：问题主题

□ Question focus is the **property** or **entity** that is being sought by the question

- “In which **city** was Barack Obama born?”
- “What is the **population** of Galway?”

□ 问题主题：问题是关于哪方面的：

- “What is the height of Mount Everest?”
 - (geography, mountains)
 - “Which organ is affected by the Meniere’s disease?”
 - (medicine)
-

概念五：问答来源类型

□ Structure level:

- 结构化数据 (databases).
- 半结构化数据 (e.g. XML).
- 非结构化数据.

□ 数据源:

- Single dataset (centralized).
 - Enumerated list of multiple, distributed datasets.
 - Web-scale.
-

概念六：领域类型

☐ Domain Scope:

- 开放域
- 特定域

☐ Data Type:

- 文本
- 图片
- 音频
- 视频

☐ 多模态问答

☐ Visual QA

概念七：答案格式

- Long answers
 - Definition/justification based.
- Short answers
 - Phrases.
- Exact answers
 - Named entities, numbers, aggregate, yes/no.

问答质量的评估原则

- ❑ **Relevance-相关度**: The level in which the answer addresses users information needs.
- ❑ **Correctness-正确度**: The level in which the answer is factually correct.
- ❑ **Conciseness-精炼度**: 答案不包含不相关信息
- ❑ **Completeness-完备度**: 答案应该完整
- ❑ **Simplicity-简单度**: 答案易于解释
- ❑ **Justification-合理度**: Sufficient context should be provided to support the data consumer in the determination of the query correctness.

答案的评估

- ☐ **Right:** The answer is correct and complete.
 - ☐ **Inexact:** The answer is incomplete or incorrect.
 - ☐ **Unsupported:** The answer does not have an appropriate evidence/justification.
 - ☐ **Wrong:** The answer is not appropriate for the question.
-

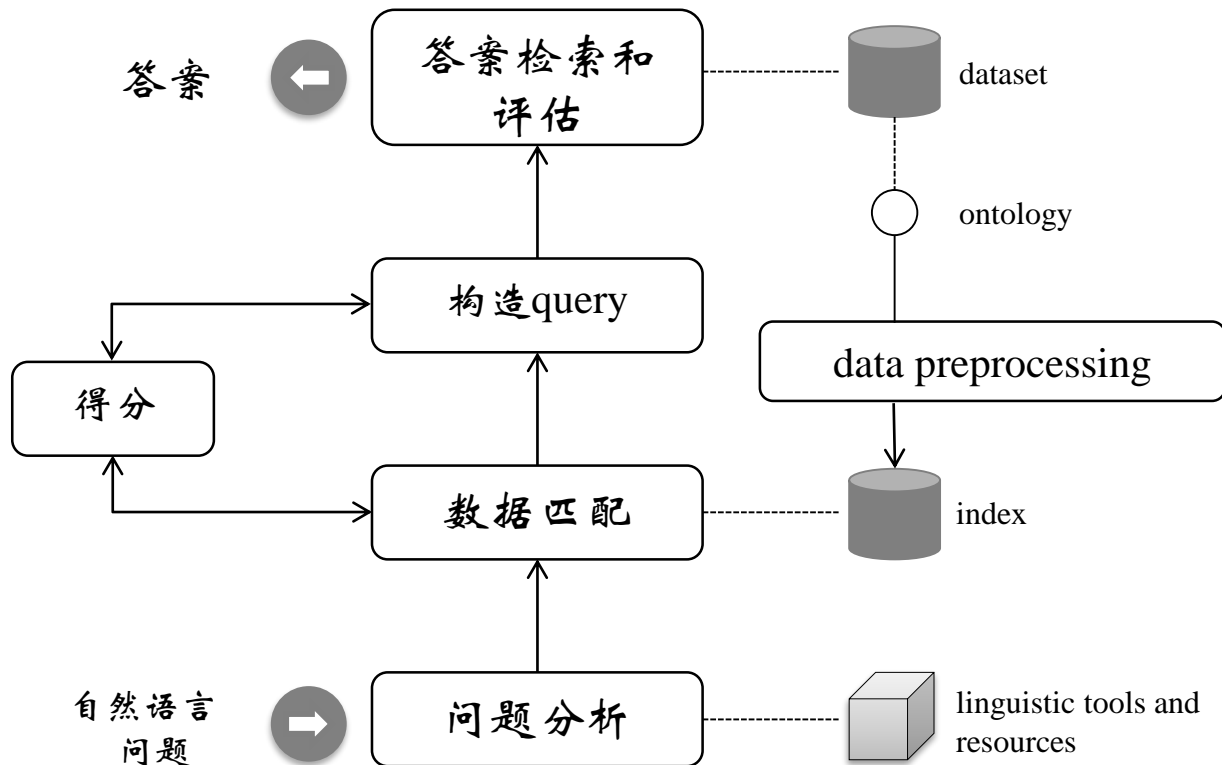
答案的处理

- ❑ **简单抽取:** Direct extraction of snippets from the original document(s) / data records.
 - ❑ **Combination:** Combines excerpts from multiple sentences, documents / multiple data records, databases.
 - ❑ **Summarization:** Synthesis from large texts / data collections.
 - ❑ **Operational/functional:** Depends on the application of functional operators.
 - ❑ **推理:** Depends on the application of an inference process over the original data.
-

问答任务的复杂性

- ❑ **Semantic Tractability** (Popescu et al., 2003): Vocabulary overlap/distance between the query and the answer.
 - ❑ **Answer Locality** (Webber et al., 2003): Whether answer fragments are distributed across different document fragments / documents or datasets/dataset records.
 - ❑ **Derivability** (Webber et al, 2003): Dependent if the answer is explicit or implicit. Level of reasoning dependency.
 - ❑ **Semantic Complexity:** Level of ambiguity and discourse/data heterogeneity.
-

问答系统的基本组件



问答系统的基本组件

- ❑ **数据预处理-Data pre-processing:** Preprocesses the database data (includes indexing, data cleaning, feature extraction).
 - ❑ **问题分析-Question Analysis:** Performs syntactic analysis and detects/extracts the core features of the question (NER, answer type, etc).
 - ❑ **数据匹配-Data Matching:** Matches terms in the question to entities in the data.
 - ❑ **查询创建-Query Constuction:** Generates structured query candidates considering the question-data mappings and the syntactic constraints in the query and in the database.
 - ❑ **排序-Scoring:** Data matching and the query construction components output several candidates that need to be scored and ranked according to certain criteria.
 - ❑ **结果返回与生成-Answer Retrieval & Extraction:** Executes the query and extracts the natural language answer from the result set.
-

基于知识图谱的问答：基本需求

- **High usability:**
 - Supporting natural language queries.
 - **High query expressivity:**
 - Path, conjunctions, disjunctions, aggregations, conditions.
 - **Accurate & comprehensive semantic matching:**
 - High precision and recall.
 - **Low maintainability:**
 - Easily transportable across datasets from different domains (minimum adaptation effort/low adaptation time).
 - **Low query execution time:**
 - Suitable for interactive querying.
 - **High scalability:**
 - Scalable to a large number of datasets (Organization-scale, Web-scale).
-

技术挑战: 如何将问题映射到答案

Which mountains
are higher than
the Nanga Parbat?

```
PREFIX dbo: <http://dbpedia.org/ontology/>  
PREFIX res: <http://dbpedia.org/resource/>
```

```
SELECT DISTINCT ?uri WHERE {  
  ?uri a dbo:Mountain .  
  res:Nanga_Parbat dbo:elevation ?e1 .  
  ?uri dbo:elevation ?e2 .  
  FILTER (?e2 > ?e1) .  
}
```



Eight mountains
are higher than
the Nanga Parbat:
Mount Everest,
Makalu, K2, ...

```
resource:Nanga_Parbat a dbpedia:Mountain .  
resource:Nanga_Parbat dbpedia:elevation '8126' .  
resource:K2 a dbpedia:Mountain .  
resource:K2 dbpedia:elevation '8611' .  
resource:Annapurna a dbpedia:Mountain .  
resource:Annapurna dbpedia:elevation '8091' .  
resource:Mount_Everest a dbpedia:Mountain .  
resource:Mount_Everest dbpedia:elevation '8848' .  
resource:Amsterdam a dbpedia:City .  
resource:Amsterdam dbpedia:elevation '2' .  
...
```

```
<http://dbpedia.org/resource/Mount_Everest>  
<http://dbpedia.org/resource/Makalu>  
<http://dbpedia.org/resource/K2>  
...
```

自然语言问题与知识图谱之间的鸿沟

Example: *What is the currency of the Czech Republic?*

```
SELECT DISTINCT ?uri WHERE {  
  res:Czech_Republic dbo:currency ?uri .  
}
```

Main challenges:

- ▶ Mapping natural language expressions to vocabulary elements (accounting for lexical and structural differences).
 - ▶ Handling meaning variations (e.g. ambiguous or vague expressions, anaphoric expressions).
-

映射自然语言表达式到知识图谱元素词汇

- URIs are language independent identifiers.
- Their only actual connection to natural language is by the labels that are attached to them.

`dbo:spouse rdfs:label “spouse”@en , “echtgenoot”@nl .`

- Labels, however, do not capture lexical variation:

wife of
husband of
married to

...

映射自然语言表达式到知识图谱元素词汇

Which Greek cities have more than 1 million inhabitants?

```
SELECT DISTINCT ?uri
WHERE {
    ?uri rdf:type dbo:City .
    ?uri dbo:country res:Greece .
    ?uri dbo:populationTotal ?p .
    FILTER (?p > 1000000)
}
```

映射自然语言表达式到知识图谱元素词汇

- Often the conceptual granularity of language does not coincide with that of the data schema.

When did Germany [join the EU](#)?

```
SELECT DISTINCT ?date
WHERE {
    res:Germany dbp:accessioneudate ?date .
}
```

Who are the [grandchildren](#) of Bruce Lee?

```
SELECT DISTINCT ?uri
WHERE {
    res:Bruce_Lee dbo:child ?c .
    ?c dbo:child ?uri .
}
```


映射自然语言表达式到知识图谱元素词汇

- In addition, there are expressions with a fixed, dataset-independent meaning.

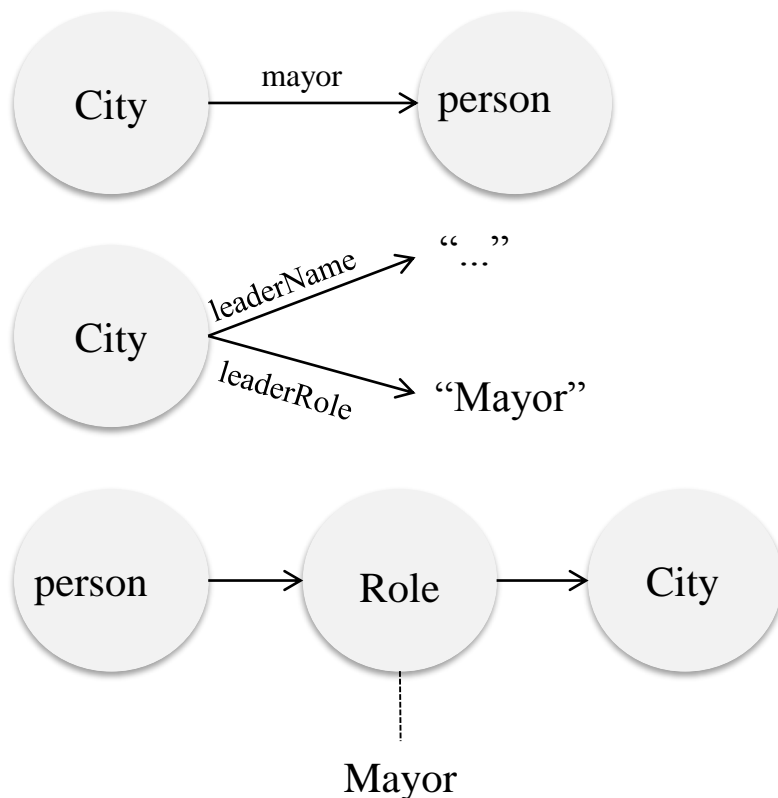
Who produced **the most** films?

```
SELECT DISTINCT ?uri
WHERE {
    ?x rdf:type dbo:Film .
    ?x dbo:producer ?uri .
}
ORDER BY DESC(COUNT(?x))
OFFSET 0 LIMIT 1
```

不同的知识表示增加了映射难度

- Different datasets usually follow different schemas, thus provide different ways of answering an information need.

Example:



关系或属性隐含表述

- The meaning of expressions like the verbs *to be*, *to have*, and prepositions *of*, *with*, etc. strongly depends on the linguistic context.

Which museum **has** the most paintings?

?museum **dbo:exhibits** ?painting .

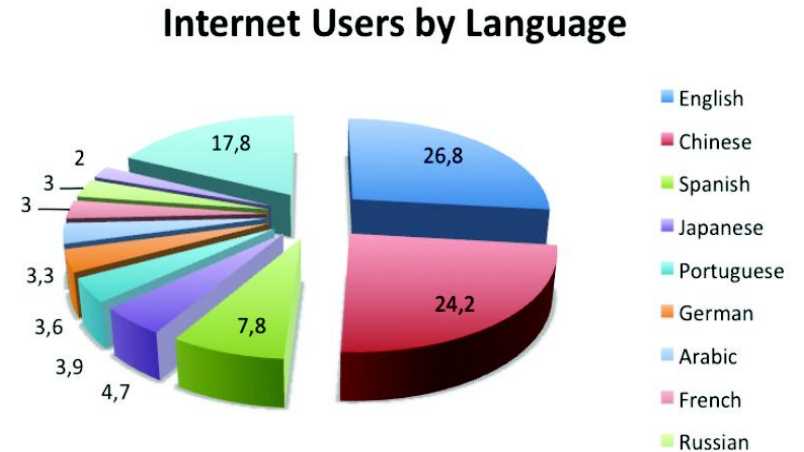
Which country **has** the most caves?

?cave **dbo:location** ?country .

知识库如何支持多语言问答

- The number of non-English actors on the web is growing substantially.

- Accessing data.
- Creating and publishing data.



Knowledge Graph:

- In principle very well suited for multilinguality, as URIs are language-independent.
- But adding multilingual labels is not common practice (less than a quarter of the RDF literals have language tags, and most of those tags are in English).

数据质量和异构性

- Requirement: Completeness and accuracy
(Wrong answers are worse than no answers)

In the context of the Knowledge Graph:

- QA systems need to deal with heterogeneous and imperfect data.
 - Datasets are often incomplete.
 - Different datasets sometimes contain duplicate information, often using different vocabularies even when talking about the same things.
 - Datasets can also contain conflicting information and inconsistencies.

分布式和互联数据

- Data is distributed among a large collection of interconnected datasets.

Example: What are side effects of drugs used for the treatment of Tuberculosis?

```
SELECT DISTINCT ?x
WHERE {
    disease:1154 diseasome:possibleDrug ?d1.
    ?d1 a drugbank:drugs .
    ?d1 owl:sameAs ?d2.
    ?d2 sider:sideEffect ?x.
}
```

回答性能和可扩展性

- Requirement: Real-time answers, i.e. low processing time.

In the context of the Knowledge Graph:

- Datasets are huge.
 - There are a lot of distributed datasets that might be relevant for answering the question.
 - Reported performance of current QA systems amounts to ~20-30 seconds per question (on one dataset).

KBQA挑战总结

- 怎样缩小自然语言和规范化结构化数据之间的鸿沟
- 怎样处理不完全、充满噪音和异构的数据集.
- 怎样处理大规模的知识图谱.
- 怎样处理分布式数据集上的QA
- 怎样融合结构化和非结构化的数据
- 怎样降低维护成本
- 怎样能快速的复制到不同的领域

大纲

- 知识问答概述和相关数据集
- KBQA基本概念及挑战
- 知识问答主流方法介绍

KBQA 主流方法

- 基于模板的方法
- 基于语义解析的方法
- 基于深度学习的方法

基于模板的方法

□ 模板定义

□ 模板生成

□ 模板匹配

本节以TBSL (Unger et al., 2012)方法为例详细讲解

TBSL (Unger et al., 2012)

□ 核心贡献:

- Constructs a **query template** that directly mirrors the **linguistic structure of the question**
- Instantiates the template by matching **natural language expressions** with **ontology concepts**

□ 评估: QALD 2012

动机

- In order to understand a user question, we need to understand:

The words (dataset-specific)

Abraham Lincoln \rightarrow res:Abraham Lincoln
died in \rightarrow dbo:deathPlace

The semantic structure (dataset-independent)

who \rightarrow SELECT ?x WHERE { ... }
the most $N \rightarrow$ ORDER BY DESC(COUNT(?N)) LIMIT 1
more than $i \ N \rightarrow$ HAVING COUNT(?N) > i

基于模板的问答

- 目标: An approach that combines both an analysis of the **semantic structure** and a **mapping** of words to URIs.
 - 方法的两个重要步骤:
 - 1. Template generation (模板生成)
 - Parse question to produce a SPARQL template that directly mirrors the **structure** of the question, including **filters and aggregation operations**.
 - 2. Template instantiation (模板实例化)
 - Instantiate SPARQL template by matching natural language expressions with ontology concepts using statistical **entity identification** and **predicate detection**.
-

示例: Who produced the most films?

SPARQL template:

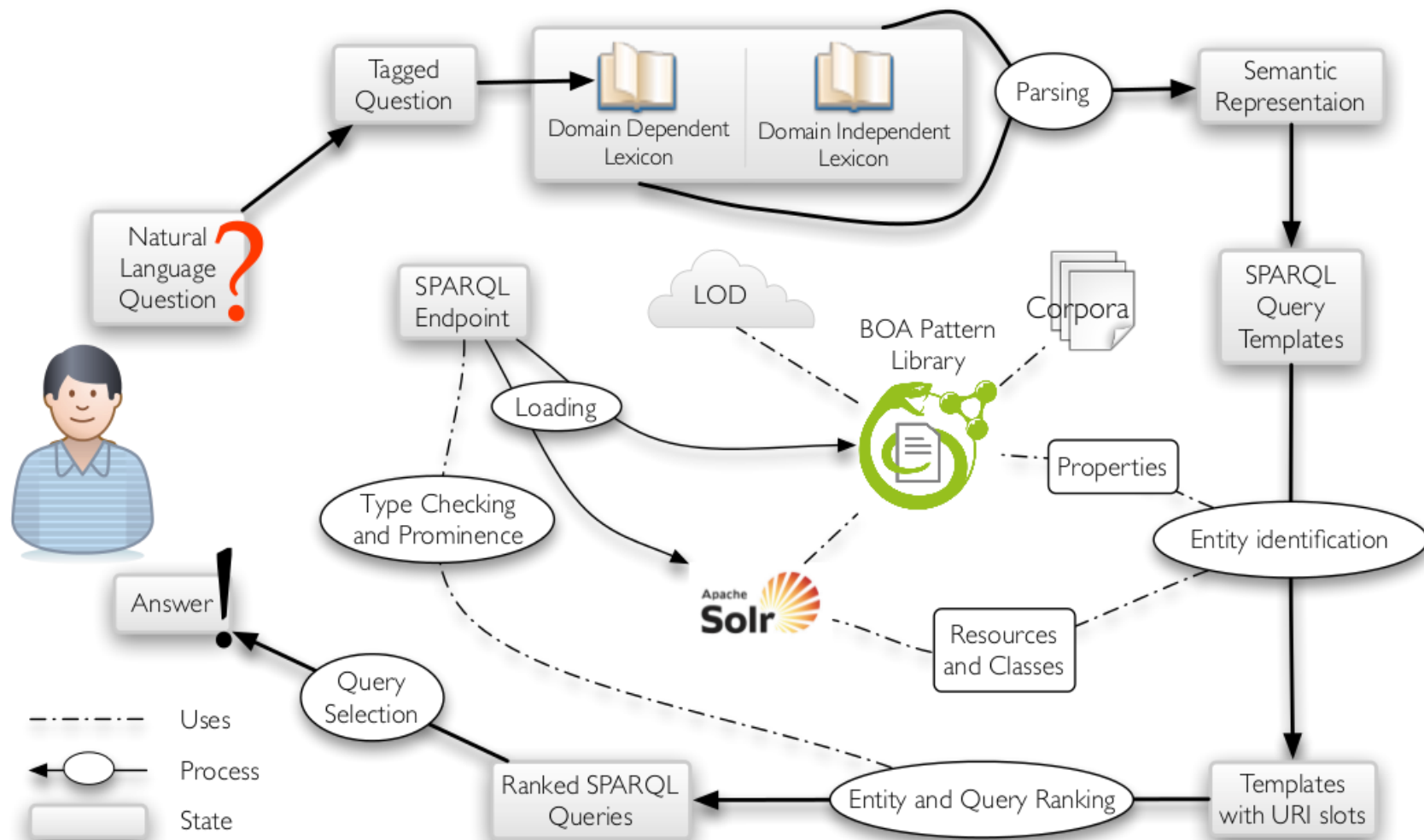
```
SELECT DISTINCT ?x WHERE {  
    ?y rdf:type ?c .  
    ?y ?p ?x .  
}  
ORDER BY DESC(COUNT(?y))  
OFFSET 0 LIMIT 1
```

```
?c CLASS [films]  
?p PROPERTY [produced]
```

Instantiations:

```
?c = <http://dbpedia.org/ontology/Film>  
?p = <http://dbpedia.org/ontology/producer>
```

TBSL 架构



模板定义

- 结合KG的结构，以及问句的句式，进行模板定义。通常没有统一的标准或格式
- TBSL的模板定义为SPARQL query模板，将其直接与自然语言相映射

Step 1: 模板生成 – Linguistic processing

1. 首先，获取自然语言问题的POS tags信息
2. 其次，基于POS tags, 语法规则表示问句
3. 然后利用domain-dependent词汇和domain-independent词汇辅助分析问题
4. 最后，将语义表示转化为一个SPARQL模板

示例: Who produced the most films?

☐ 领域无关: who, the most

☐ 领域相关: produced/VBD, films/NNS

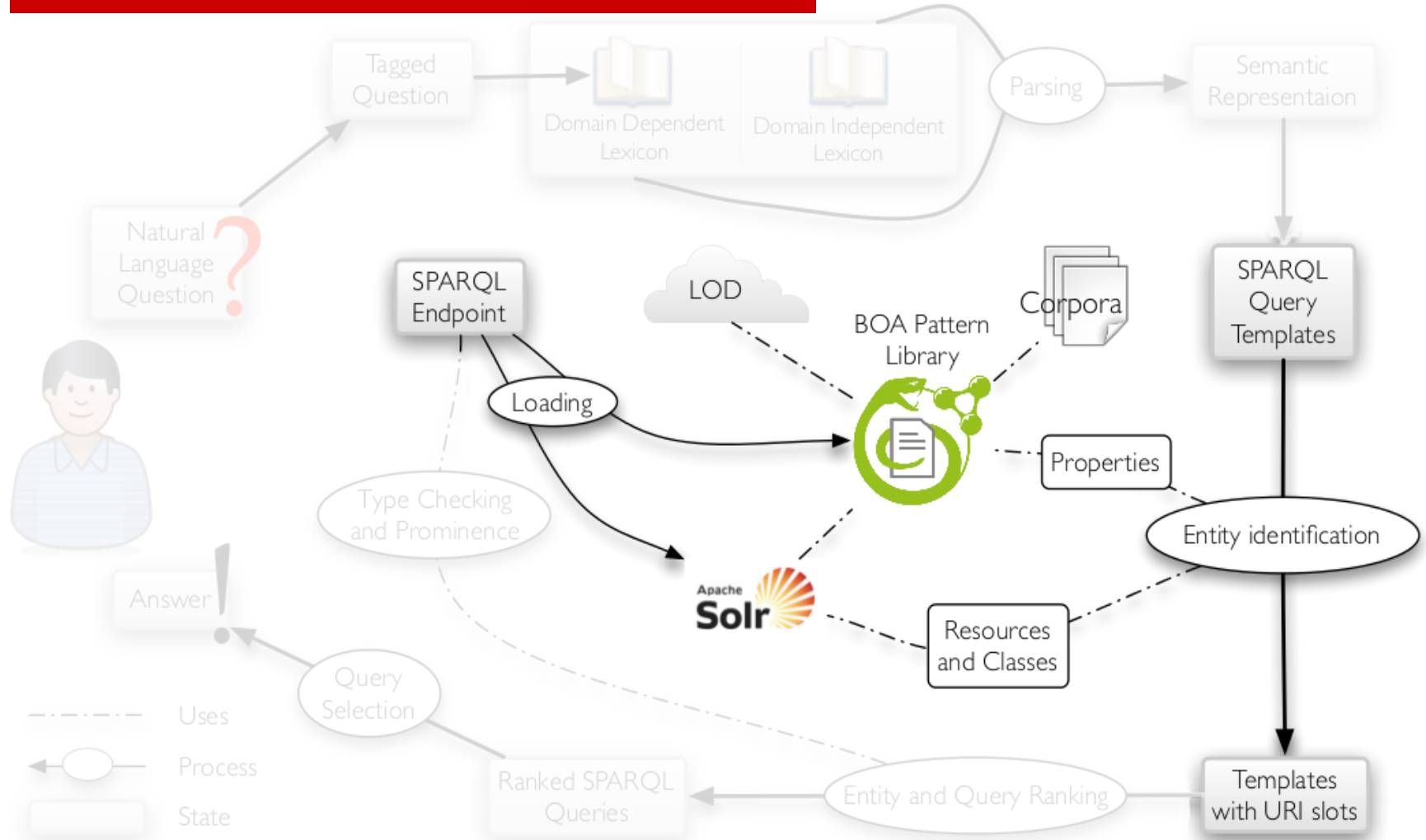
SPARQL template 1:

```
SELECT DISTINCT ?x WHERE {  
    ?x ?p ?y .  
    ?y rdf:type ?c .  
}  
ORDER BY DESC(COUNT(?y)) LIMIT 1  
?c CLASS [films]  
?p PROPERTY [produced]
```

SPARQL template 2:

```
SELECT DISTINCT ?x WHERE {  
    ?x ?p ?y .  
}  
ORDER BY DESC(COUNT(?y)) LIMIT 1  
?p PROPERTY [films]
```

Step 2: Template instantiation Entity identification and predicate detection



模板匹配与实例化-实体识别与属性检测

- 有了SPARQL模板以后，需要进行实例化与具体的自然语言问句相匹配。即将自然语言问句与知识库中的本体概念相映射的过程。
- 对于resources和classes,实体识别常用方法:
 - 用WordNet定义知识库中标签的同义词
 - 计算字符串相似度 (trigram, Levenshtein 和子串相似度)
- 对于property labels,将还需要与存储在BOA模式库中的自然语言表示进行比较
- 最高排位的实体将作为填充查询槽位的候选答案

示例: Who produced the most films?

?c CLASS [films]

<http://dbpedia.org/ontology/Film>

<http://dbpedia.org/ontology/FilmFestival>

...

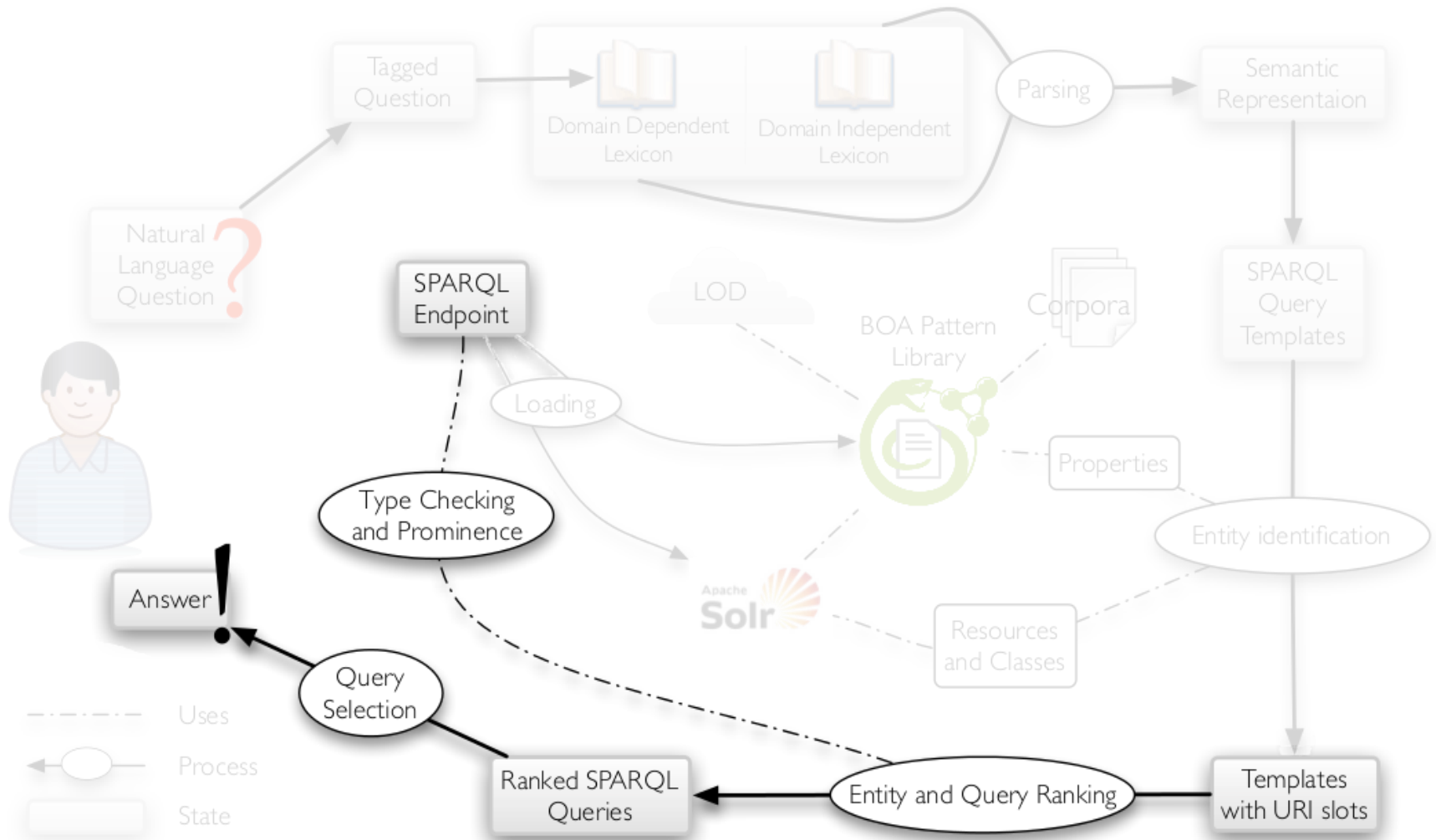
?p PROPERTY [produced]

<http://dbpedia.org/ontology/producer>

<http://dbpedia.org/property/producer>

<http://dbpedia.org/ontology/wineProduced>

Step 3: Query ranking and selection



排序打分

1. 每个entity根据string similarity和prominence获得一个打分
2. 一个query模板的分值根据填充slots的多个entities的平均打分
3. 另外,需要检查type类型:
 - 对于所有的三元组?x rdf:type <class>,对于查询三元组?x p e和 e p ?x需要检查p的domain/range是否与<class>一致
4. 对于全部的查询集合, 仅返回打分最高的.

示例: Who produced the most films?

```
SELECT DISTINCT ?x WHERE {  
  ?x <http://dbpedia.org/ontology/producer> ?y .  
  ?y rdf:type <http://dbpedia.org/ontology/Film> .  
}  
ORDER BY DESC(COUNT(?y)) LIMIT 1
```

Score: 0.76

```
SELECT DISTINCT ?x WHERE {  
  ?x <http://dbpedia.org/ontology/producer> ?y .  
  ?y rdf:type <http://dbpedia.org/ontology/FilmFestival> .  
}  
ORDER BY DESC(COUNT(?y)) LIMIT 1
```

Score: 0.60

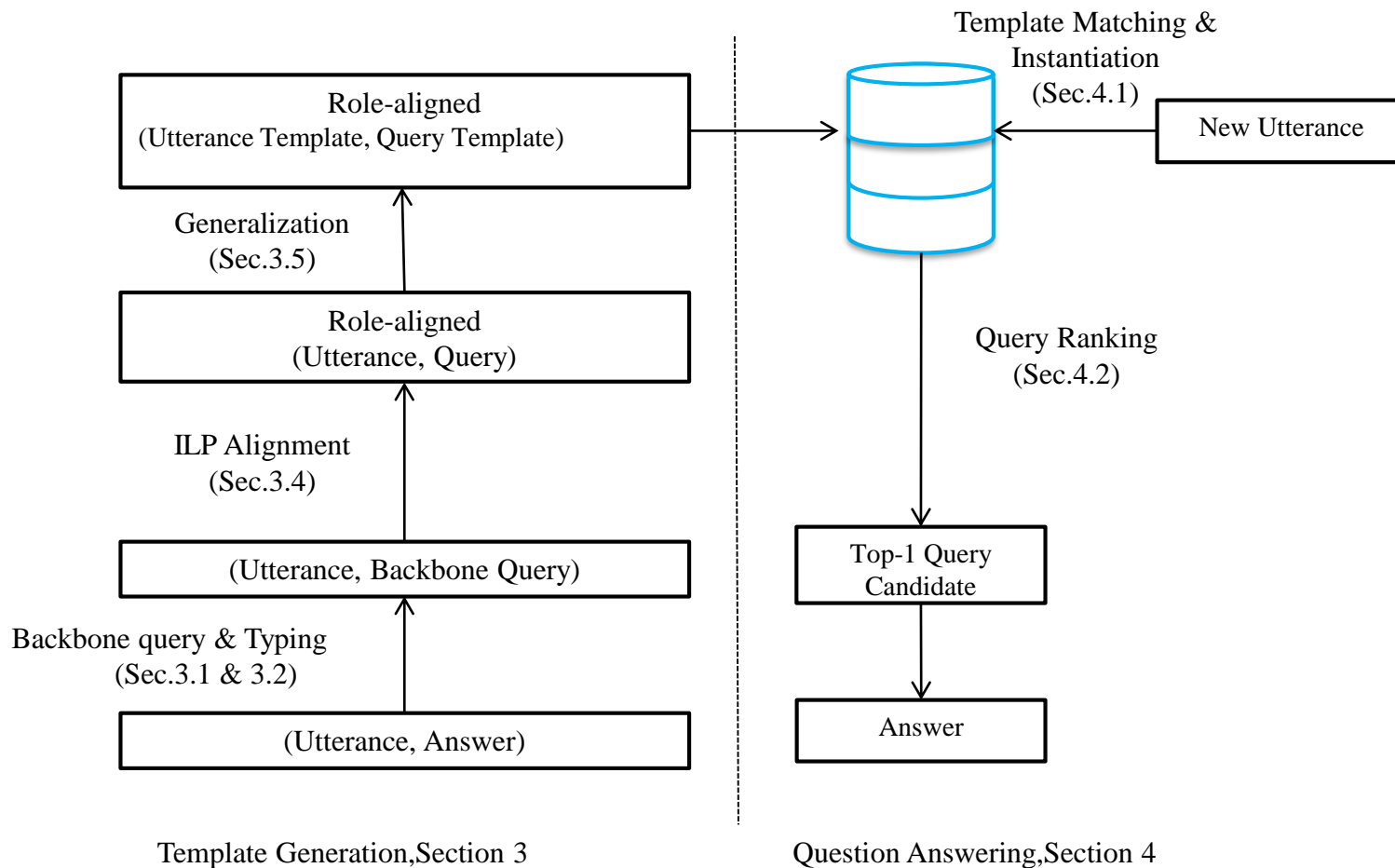
TBSL 主要缺点

- ❑ 创建的模板结构未必和知识图谱中的数据建模相契合
- ❑ 考虑到数据建模的各种可能性，对应到一个问题的潜在模板数量会非常多，同时手工准备海量模板的代价也非常大

模板是否可以自动生成呢？

- ❑ Automated Template Generation for Question Answering over Knowledge Graphs (Abujabal et al, www2017)
 - 提出了QUINT，能够根据utterance-answer对，根据依存树自动学习utterance-query模板
 - 利用自然语言的组成特点，可以使用从简单问题中学到的模板来解决复杂问题

GUINT 架构



方法与主要贡献点

1. 提出了QUINT能够根据utterance-answer pair, 使用依存树自动学习utterance-query模板。模板的学习使用远程监督的方法。模板支持自动识别问题答案的类型
 - 其中, 使用Integer Linear Programming (ILP)学习utterance-query之间的对齐
2. 利用自然语言的组成特点, 可以使用从简单问题中学到的模板来解决复杂问题(多个predicate)复杂问题解决流程:
 - 将问题分解为子句
 - 使用模板回答每一个子句
 - 结合子句答案获取最终答案

模板生成-问句依存分析

□ 训练阶段的输入是问题utterance u 和它对应的答案集合 A_u

□ 输入示例：

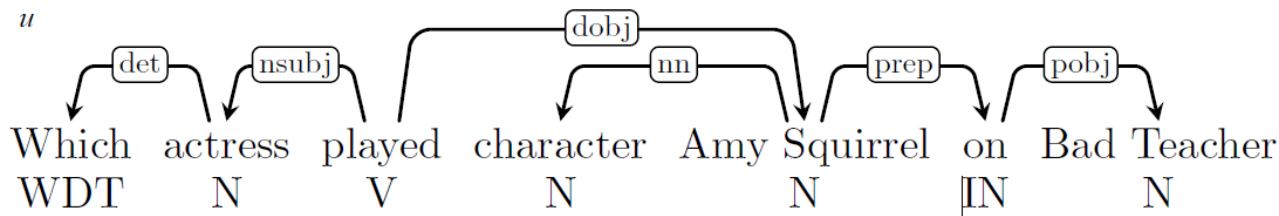
■ utterance: u = “Which actress played character Amy Squirrel on Bad Teacher?”

■ $A_u = \{\text{LucyPunch}\}$

□ 对应依存树：

使用依存树的好处：

- 依存树能够捕捉远距离依赖关系, 适合复杂问题
- 能够跳过无用的tokens, 更好的容错性



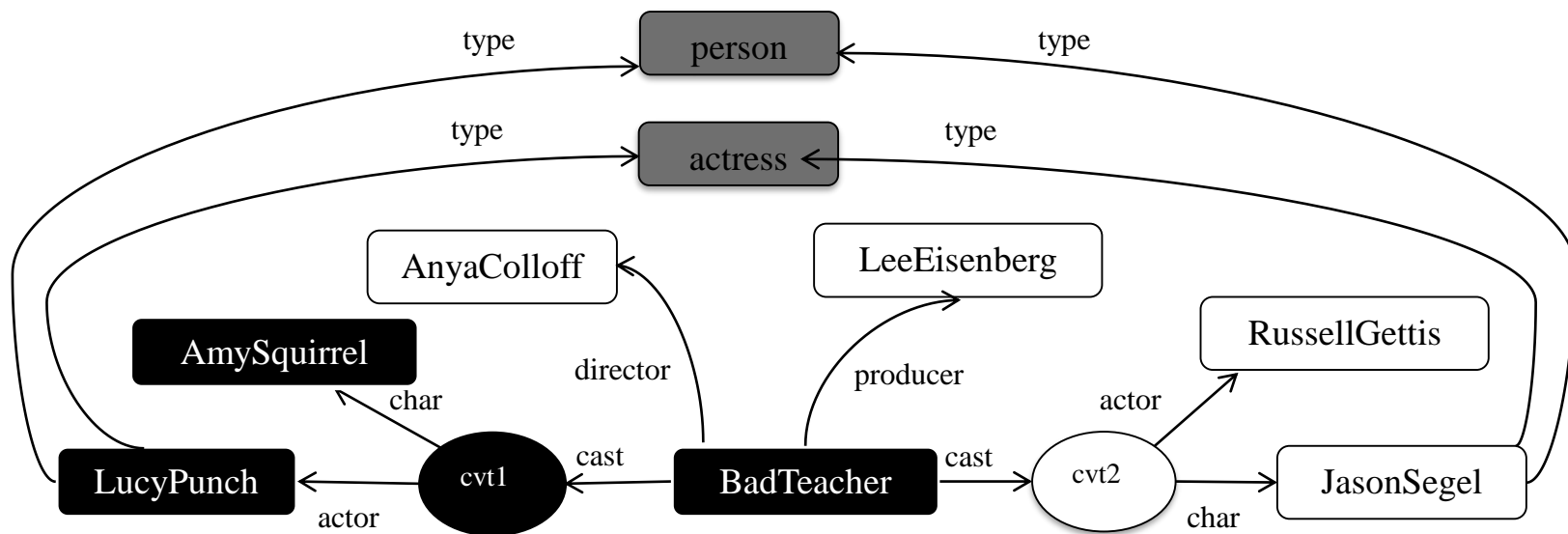
$A_u = \{\text{LucyPunch}\}$

模板生成-为utterance构建query

- 根据utterance和answer中的entity（使用工具S-MART进行NERL，与freebase链接），从KG中得到最小子图

utterance: u = “Which actress played character **Amy Squirrel** on **Bad Teacher**?”

Au = {**LucyPunch**}



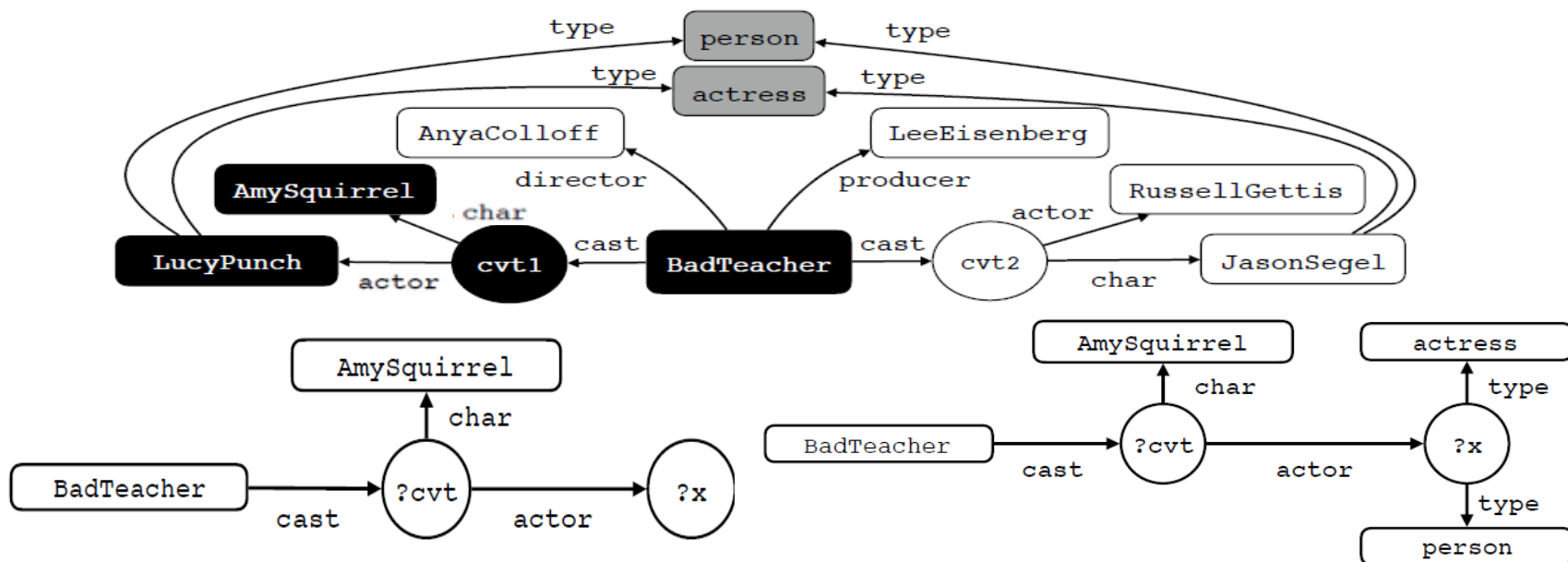
S-MART 来自： novel tree-based structured learning algorithms applied to tweet entity linking. In ACL, 2015.

模板生成-为utterance构建query

utterance: u = “Which actress played character **Amy Squirrel** on **Bad Teacher**?”

$$\text{Au} = \{\text{LucyPunch}\}$$

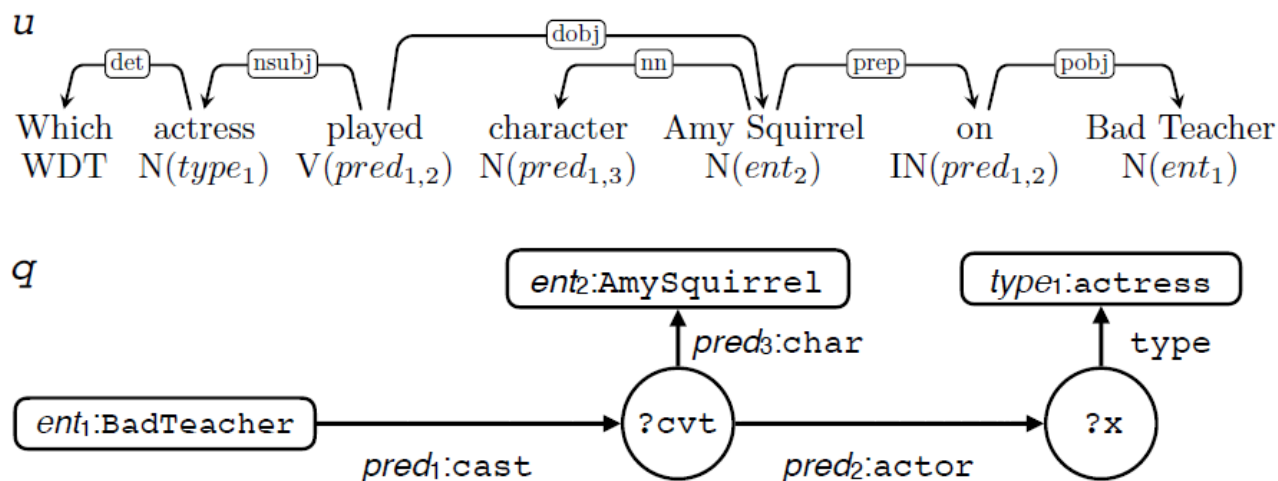
黑点构成最小子图，进一步将answer和cvt替换为变量?x, ?cvt, 并添加answer上的type边，即问题的答案类型（图中灰色节点）。得到utterance在KG中的子图为：



通过ILP将问题与query进行对齐

utterance: u = “Which actress played character **Amy Squirrel** on **Bad Teacher**?”

$Au = \{\text{LucyPunch}\}$



Aligned utterance query pair: (u, q, m) :

m 通过共享的 ent, pred 和 type 标注表示 (如 “played on” 与 cast.actor 对齐)

词典L构建 (详见语义解析-EMNLP 2013)

包括 L_P (Predicate lexicon) 和 L_C (type lexicon)

- 使用distant supervision方法构建
- 使用的语料: ClueWeb09-FACC1, 500M Web pages annotated with Freebase entities.

L_P 构建:

- 语料中: “[[Albert Einstein | AlbertEinstein]] was born in [[Ulm | Ulm]] ...”
- KG中: (AlbertEinstein birthPlace Ulm)
- 将“was born in” -> birthPlace添加到 L_P 中, 并添加权重, 即在语料中出现的次数

L_C 构建:

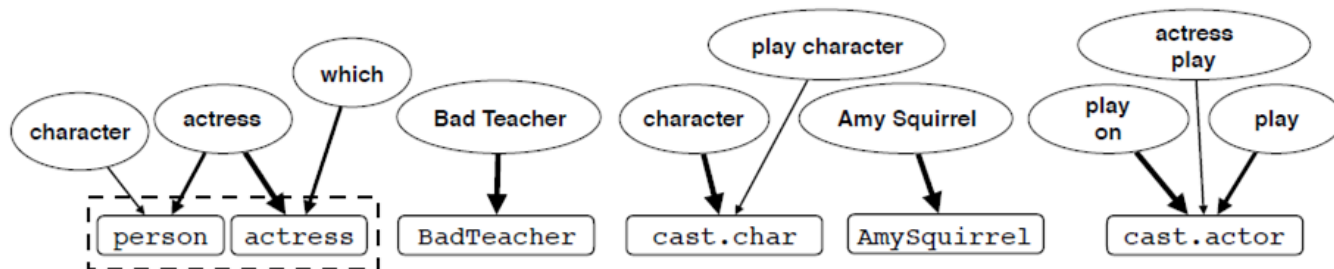
- 语料中: “[[Albert Einstein | AlbertEinstein]] and other scientists...”
- KG中: (scientists type c)
- 添加scientists -> c到 L_C 中, 并添加权重, 即在语料中出现的次数。

Ph.	S. Item	w	Ph.	S. Item	w
“play on”	cast.actor	0.5	“role”	cast.char	0.6
“play”	cast.actor	0.3	“actress”	actress	0.7
“character”	cast.char	0.6	“actress”	person	0.3

示例 L_P 和 L_C 片段

词典L构建(详见语义解析-EMNLP 2013)

- 将问题分块, 得到多个phrase
- 将问题中短语与KG对齐
- 使用词典L对齐, 出现歧义(问题歧义, 词典噪声)使用ILP解决



上面是utterance中的phrase, $Ph=\{ph1, ph2, phi...\}$ 来自utterance的子串

下面是query (即utterance在KG中对应子图)中的semantic items。 $Sq=\{s1, s2, sj.....\}$

$$\max \sum_{i,j} W_{ij} X_{ij}$$

添加边:

根据词典添加边 $phi \rightarrow sj \in L_p \cup L_c$ 边的权重来自字典

将每个semantic items和每个utterance中的phrase进行连接

使用ILP优化函数确定最优边:

函数限制条件: 每个semantic items都需要一条边。每个phrase只能对应一个semantic items。Type边只能选择一个。

注: 其中 W_{ij} 来自词典的权重, X_{ij} 表示是否保留这条边 (0,1)

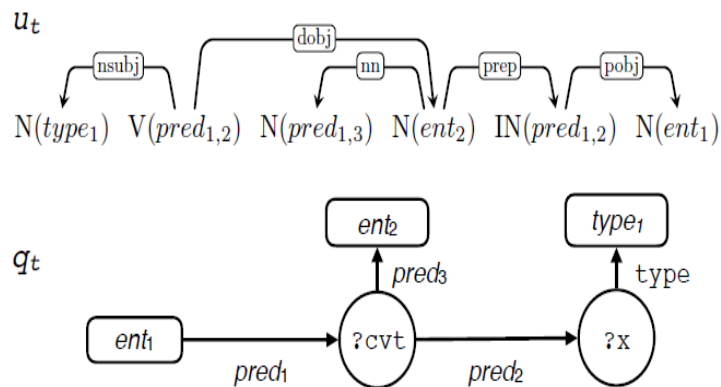
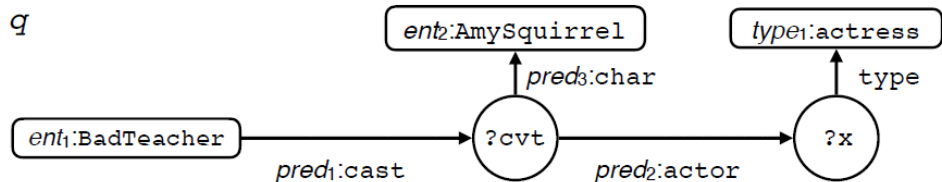
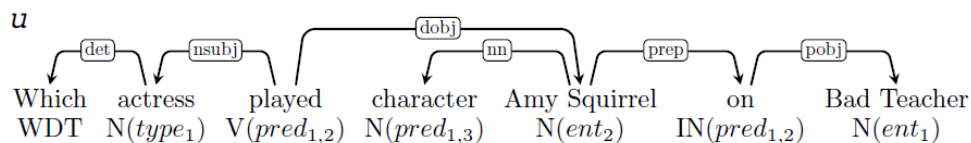
模板定义与生成

定义一个三元组的模板: (u_t, q_t, m_t) :

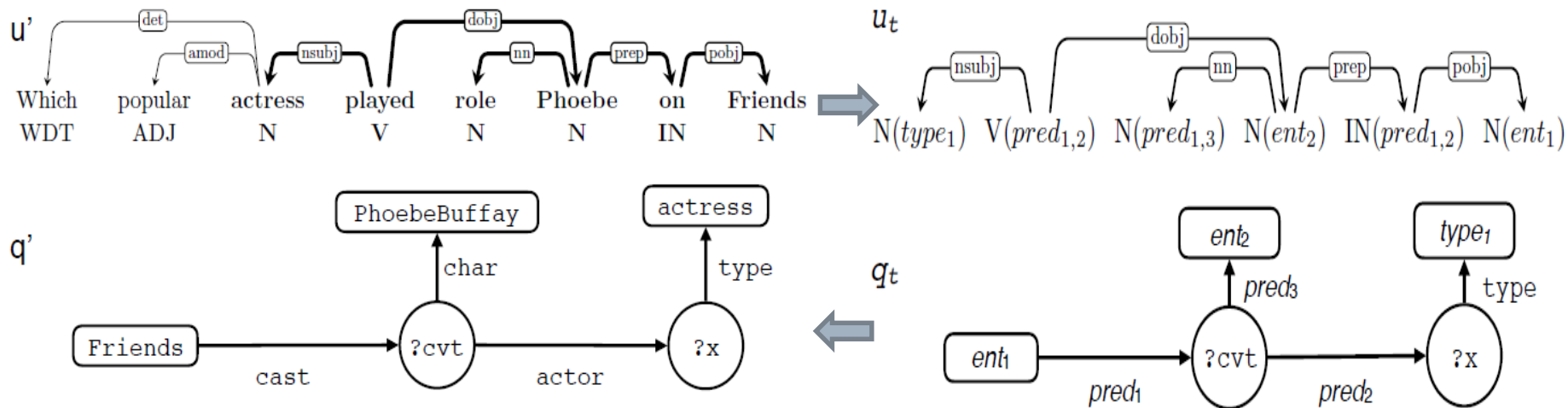
其中 u_t 为问题模板

q_t 为 query 模板

m_t 为问题和 query 映射方法模板



模板匹配和实例化



- 对于新问题进行依存分析，并使用工具S-MART进行NERL (freebase)
- 去模板库中进行匹配 (u' 中加粗的黑线与 u_t 匹配，使用子图同构匹配)
- 再使用词典 L 对 m_t (m_t 为 utterance 和 query 对齐关系) 进行实例化

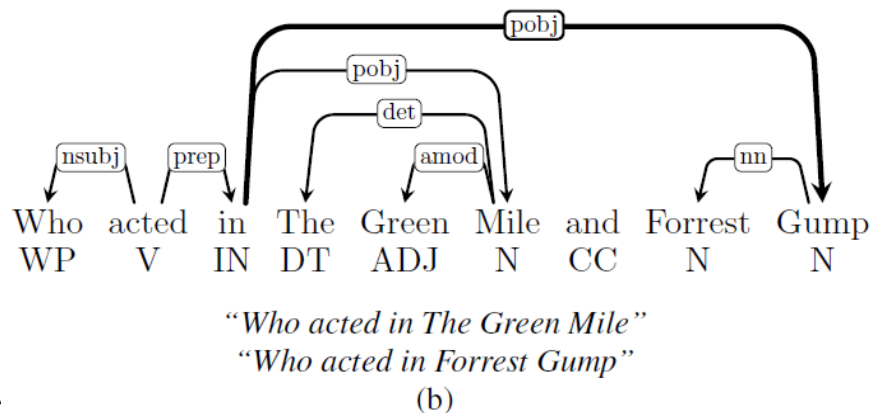
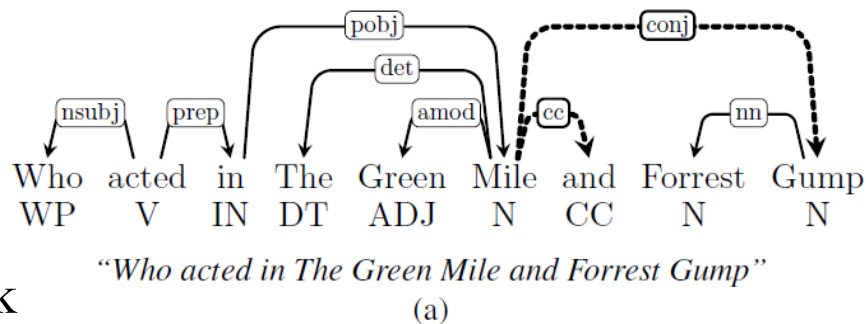
候选Query排序

- 从问题产生多个候选query的原因
 - 模板可能匹配多个
 - NERL
- 使用Random Forest进行学习两个query对之间的顺序

Category	#	Description
Alignment	1,2	Average & sum of lexicon weights for predicates
	3,4	Average & sum of lexicon weights for entities
	5	# of utterance tokens literally matching their predicate
	6	# of entity mentions matching their canonical name in the KG
	7	Indicator: question n -gram \wedge predicate p , $n = 1, 2$ and 3
Semantic	8,9	Average & sum of entity popularity scores
	10	# of predicates in the query
	11	# of entities in the query
Template	12	Indicator: t captures a type constraint
	13	# of training utterances that generate t
	14	t coverage: % of tokens in utterance matched by u_t
	15	Indicator: utterance node with $type$ annotation \wedge semantic answer type (if t is typed)
Answer	16	Indicator: answer set size (=1, =2, =3, $\in [4-10]$, > 10)

复杂问题处理

- 依存树重写
 - 并列连词
 - 关系从句
- 子问题回答
 - 匹配模板、实例化, query rank
 - 返回query list
- 答案拼接
 - 对子问题的query list打分 $1/r$, 其中 r 表示位置。
 - 对每个子问题的答案取交集
 - 选择交集不为空, 且组合后得分最高的query组合



实验结果

Method	WebQuestions Average $F1$	Free917 Accuracy
Cai and Yates [10] (2013)	-	59.0
Berant et al. [4] (2013)	35.7	62.0
Kwiatkowski et al. [19] (2013)	-	68.0
Yao and Van Durme [46] (2014)	33.0	-
Berant and Liang [5] (2014)	39.9	68.6
Bao et al. [2] (2014)	37.5	-
Bordes et al. [8] (2014)	39.2	-
Yao [45] (2015)	44.3	-
Dong et al. [12] (2015)	40.8	-
Bast and Haussmann [3] (2015)	49.4	76.4
Berant and Liang [21] (2015)	49.7	-
Yih et al. [47] (2015)	52.5	-
Reddy et al. [28] (2016)	50.3	78.0
This Work		
QUINT-untyped	50.8	78.6
QUINT	51.0	72.8

Table 4: Results on the WebQuestions and Free917 test sets.

模板匹配方法的优缺点

□ 模板方法的优点：

- 模板查询响应速度快
- 准确率较高，可以回答相对复杂的复合问题

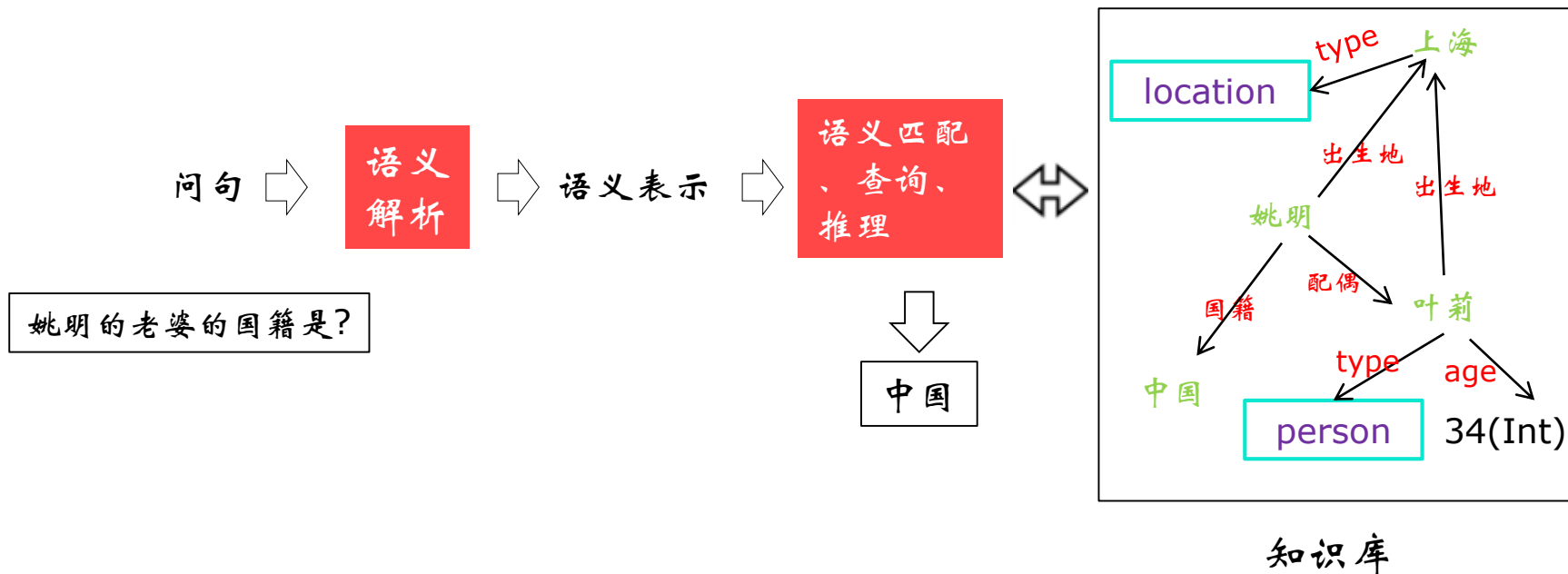
□ 模板方法的缺点：

- 人工定义的模板结构经常无法与真实的用户问题进行匹配。
- 如果为了尽可能匹配上一个问题的多种不同表述，则需要建立庞大的模板库，耗时耗力且查询起来效率降低。

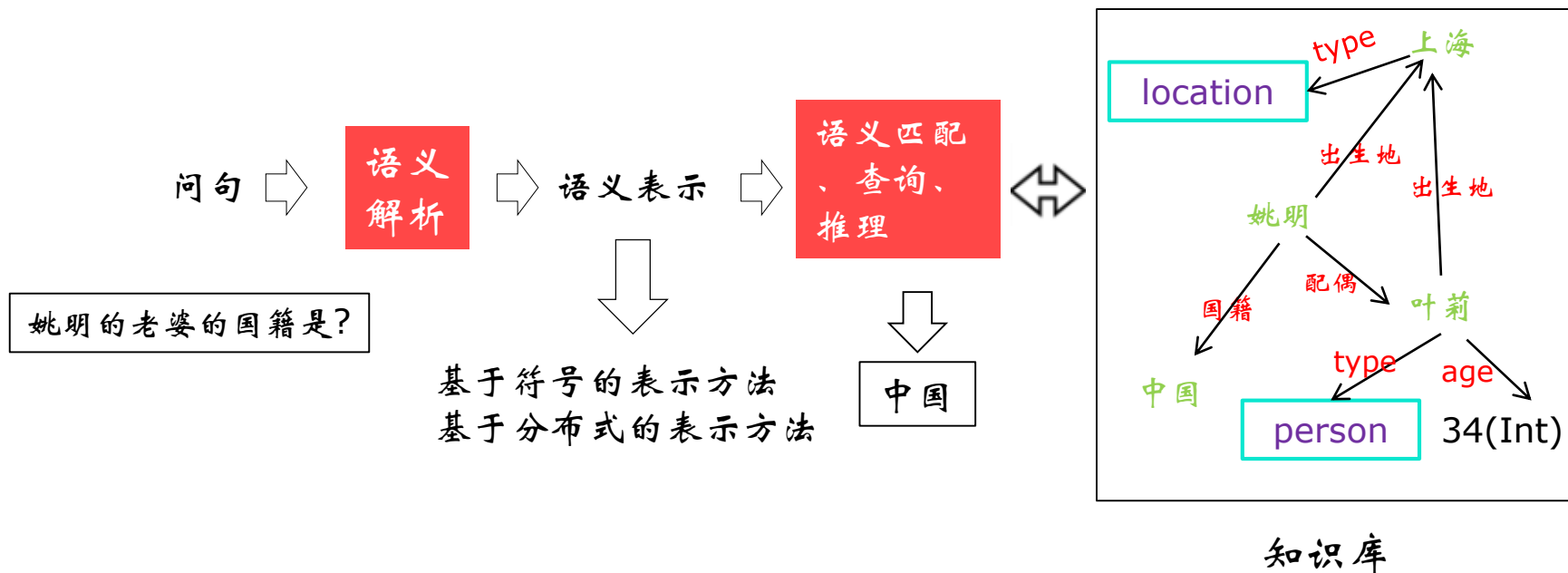
基于语义解析的方法

- 资源映射
- Logic Form
- 候选答案生成
- 排序

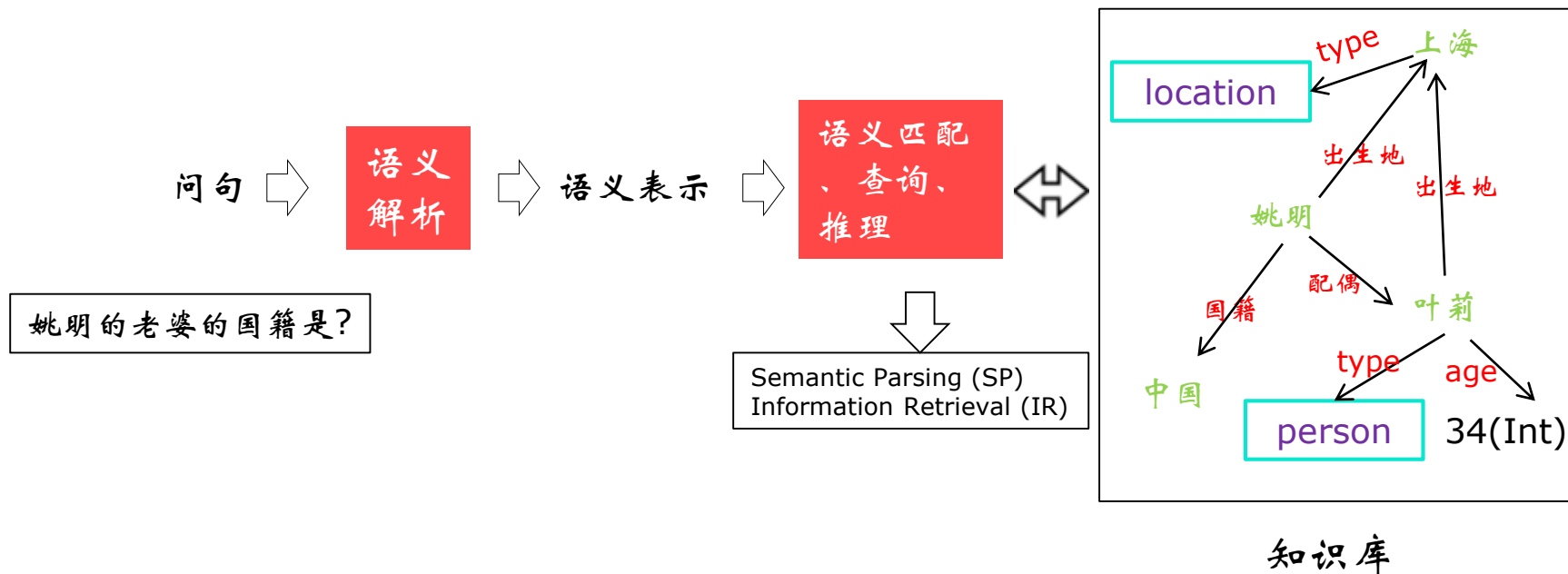
传统语义分析方法



传统语义分析方法

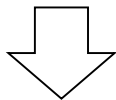


传统语义分析方法

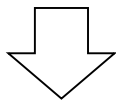


基于符号表示(传统)的知识库问答-Semantic Parsing

姚明的老婆的国籍是?

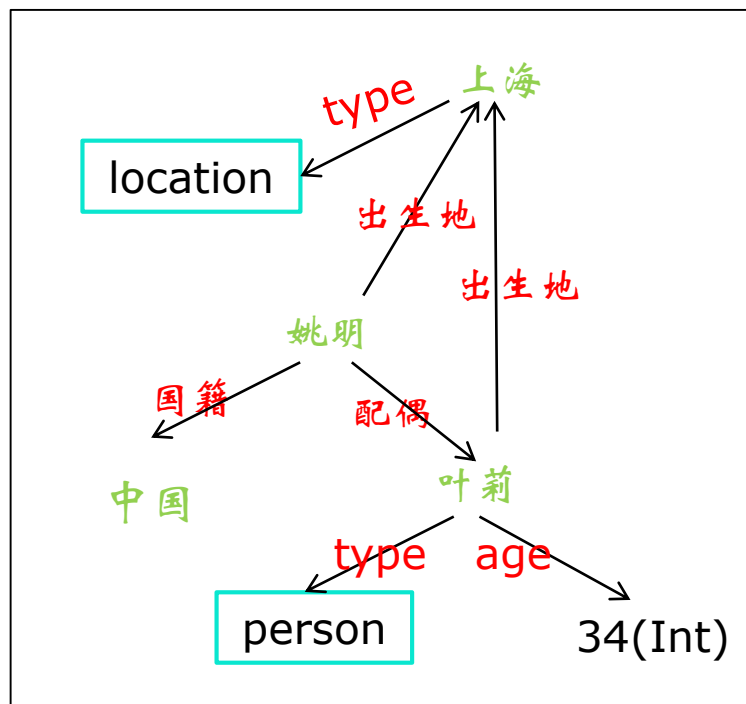
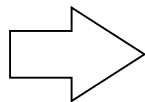


语义解析



```
SELECT DISTINCT ?X
WHERE{
  ?y 国籍 ?x.
  res:姚明 配偶 ?y.
}
```

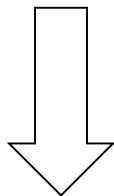
查询



知识库

问句的形式化表示

姚明的老婆出生在哪里？



$\lambda x. \text{配偶}(\text{姚明}, y) \wedge \text{出生地}(y, x)$

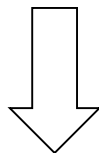
逻辑表达式

Lambda Calculus

DCS-Tree

Fun-QL

...



```
SELECT DISTINCT ?x
WHERE {
  ?y 出生地 ?x.
  res:姚明 配偶 ?y.
}
```

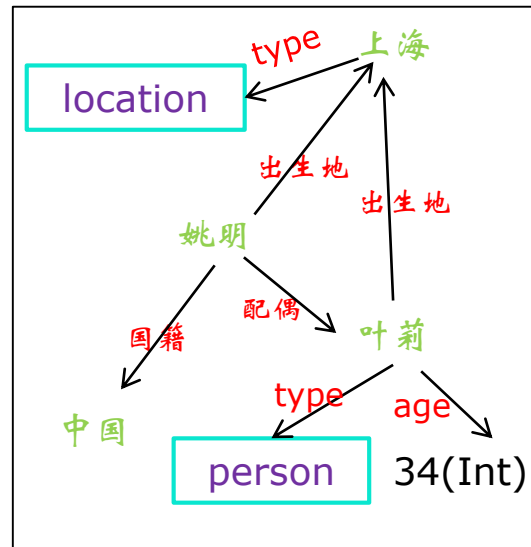
SQL

SPARQL

Prolog

FunQL

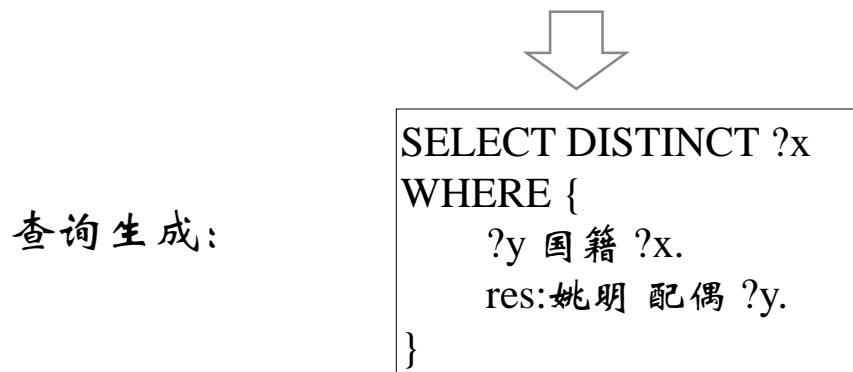
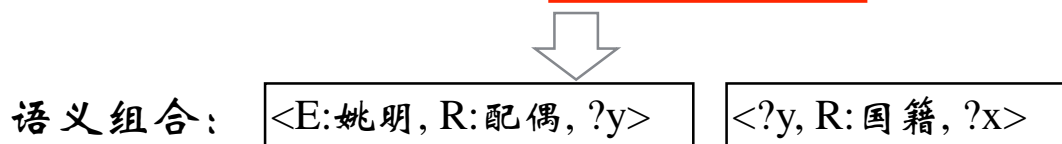
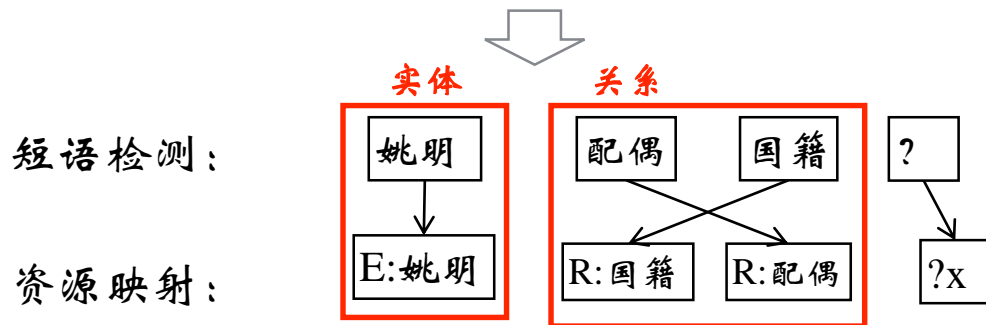
...



知识库

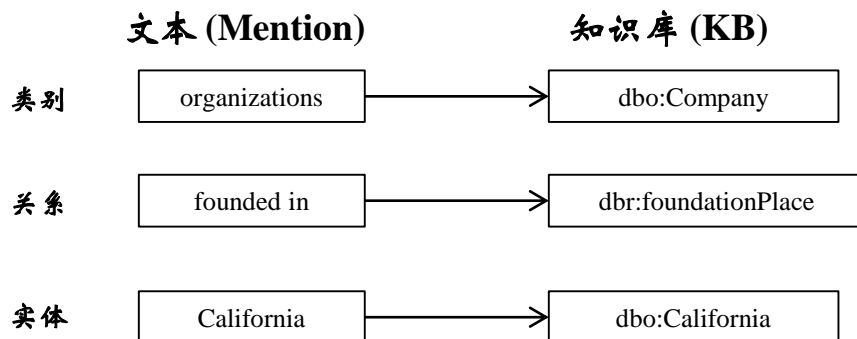
基本步骤

姚明的老婆的国籍是？

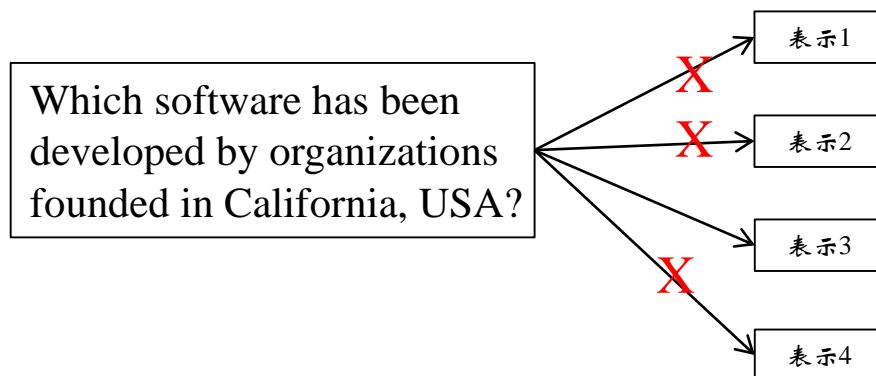


两个关键问题

- 获得短语到资源的映射 - 资源映射

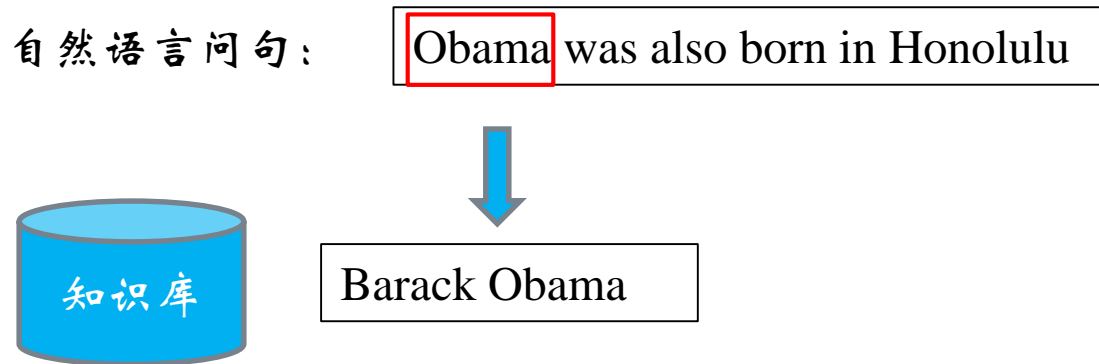


- 如何解决文本的歧义 - 逻辑表达式



语义分析—资源映射

资源映射：将自然语言短语或单词节点映射到知识库的**实体或实体关系**。可以通过构造一个词汇表（Lexicon）来完成这样的映射



简单映射：

- 字符串相似度匹配

语义分析—资源映射

- 复杂映射：“was also born in” → PlaceOfBirth
- 较难通过字符串匹配的方式建立映射，可采用统计方法
- 实体对 (entity1, entity2) 常作为主语和宾语出现在 was also born in 两侧，并且也存在在知识库中 PlaceOfBirth 的三元组中，那么我们可以认为“was also born in”这个短语可以和 PlaceOfBirth 建立映射

自然语言问句： Obama was also born in Honolulu



Barack Obama

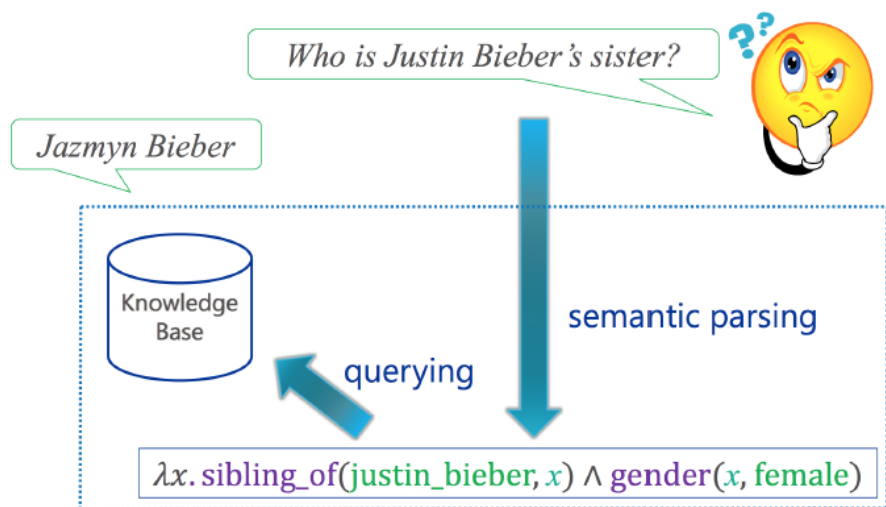
复杂映射：
• 统计方法

语义分析-逻辑表达式

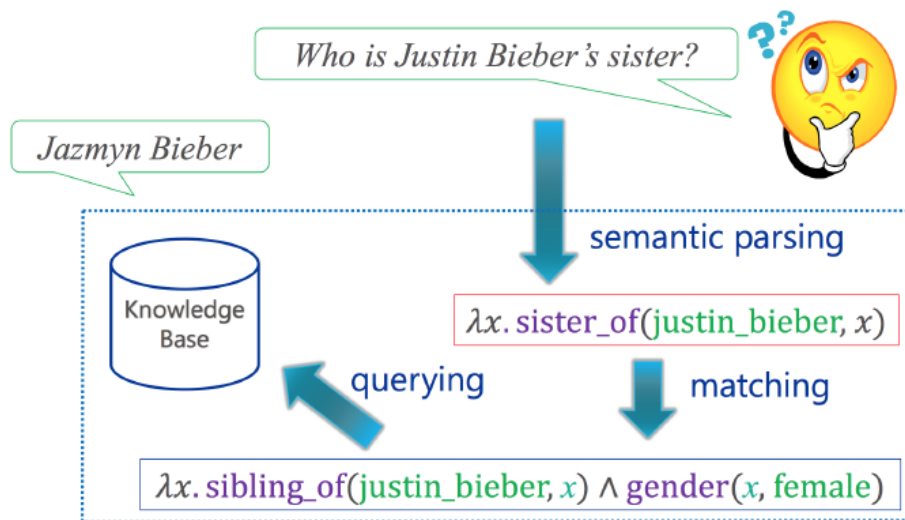
- **逻辑表达式**: 即一种能让知识库“看懂”的表示。可以表示知识库中的实体、实体关系
- 逻辑形式通常可分为一元形式 (unary) 和二元形式 (binary)
 - 一元实体: 对应知识库中的实体
 - 二元实体关系, 对应知识库中所有与该实体关系相关的三元组中的实体对。
- 并且, 可以像数据库语言一样, 进行连接Join, 求交集Intersection和聚合Aggregate (如计数, 求最大值等等) 操作

利用Logic Form表示问题语义

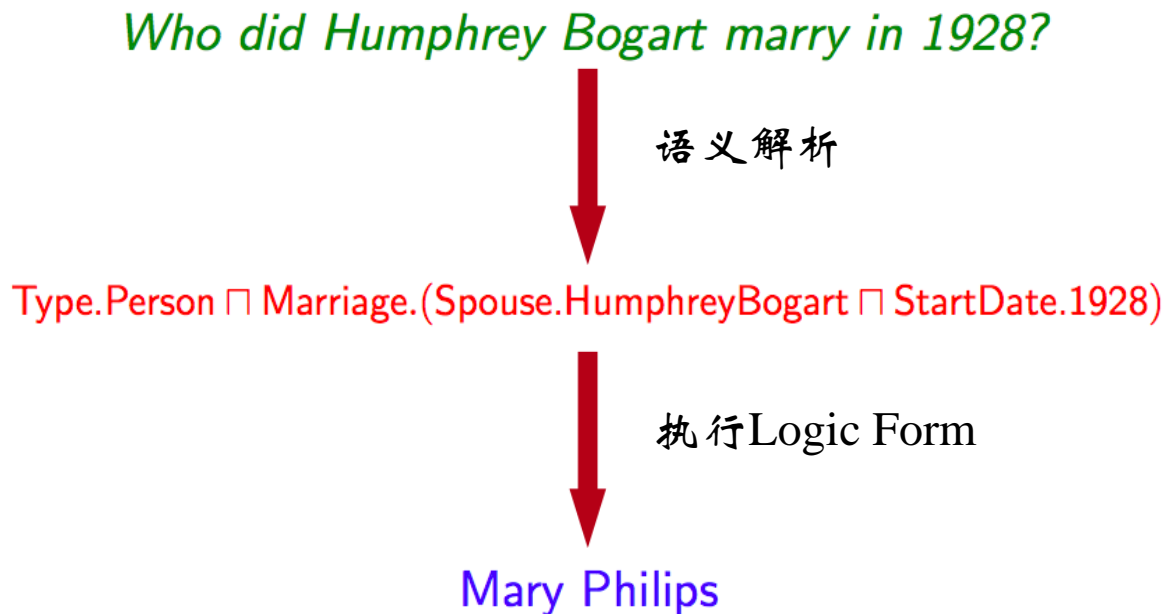
- 一步到位：直接获得与知识库相关的语义表示



- 两步到位：先通用语义表示，再实现知识库映射



语义解析经典方法 – Semantic Parsing on Freebase from Question-Answer Pairs. EMNLP 2013



Motivation: Natural language interface to large structured knowledge-bases (Freebase, DBPedia, Yelp...)

传统语义解析 VS 弱监督语义解析

□ 监督数据来源：手工编写规则

□ 监督数据来源：问题/答案对

What's California's capital?

Capital.California

How long is the Mississippi river?

RiverLength.Mississippi

...

...

What's California's capital?

Sacramento

How long is the Mississippi river?

3,734km

...

...

• 局限

- 需要专家编写—进度缓慢、代价昂贵、不可扩展
- 仅可在受限领域推广

• 优势

- 可以较为轻松地从普通民众获得

应对大规模KB的语义解析

- 无监督系统 (不需要任何训练数据)
 - Unger et al., 2012; Yahya et al., 2012

- 远距离监督 (适应于有限的小规模语义关系)
 - Krishnamurthy and Mitchell, 2012

- 通过问题/逻辑表达式对集合训练得到的Parser
 - Cai and Yates, 2013

目标：通过大规模知识库上的问题/答案对集合训练Parser

文本(问题)映射到KB的若干挑战



BarackObama
TopGun
MichelleObama
Type.Country
Type.city
Profession.Lawyer
PeopleBornHere
InventorOf
CapitalOf
⋮

字符串匹配往
往是不精确的

Type.HumanLanguage
SpeechLanguagePathology
LanguageAcquisition
⋮

What

languages

do

people

in

Brazil

use

穷举不可行

BarackObama
TopGun
MichelleObama
Type.Country
Type.city
Profession.Lawyer
PeopleBornHere
InventorOf
CapitalOf
⋮

字符串匹配存在
覆盖率低的问题

LanguagesSpoken
⋮

应对文本(问题)映射到KB的若干挑战



- ❑ **Alignment:** 从文本短语到KB predicates映射的辞典 (Lexicon)
- ❑ **Bridging:** 使用上下文 (context) 生成KB predicates

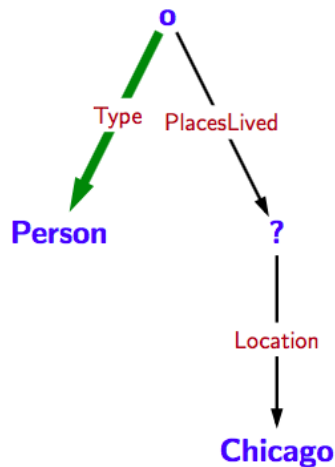
语义解析 – 问题定义

- ❑ 问题答案对 (question/answers pairs) 数据集来训练分析器
- ❑ 数据集: QALD, WebQuestions, Free917 等等 ; 也可以人工从 KB 中抽取构建

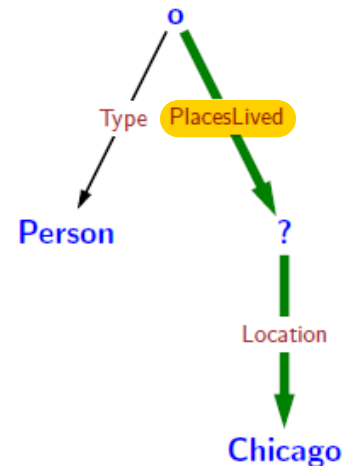
输入:
Knowledge-base K
问答对训练集合 $\{(x_i, y_i)\}_1^n$
问答对示例:
What's California's capital? Sacramento
How long is the Mississippi river? 3,734km
输出:
通过语义分析, 构建逻辑形式, 将问题 x 与答案 y 相映射。
What's California's capital? \Rightarrow <u>Capital.California</u>
\Rightarrow Sacramento

Logic Form 可以视为图模板 (Graph Template)

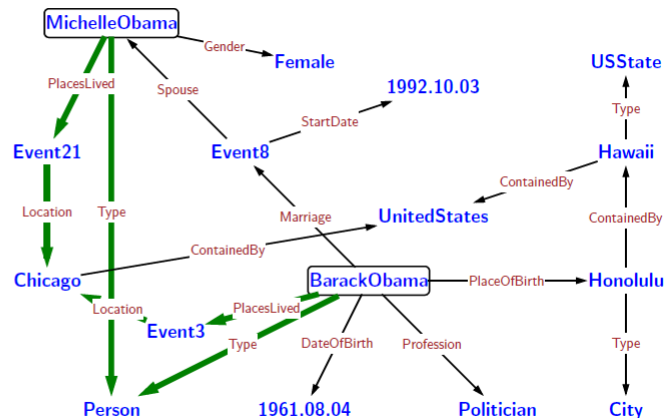
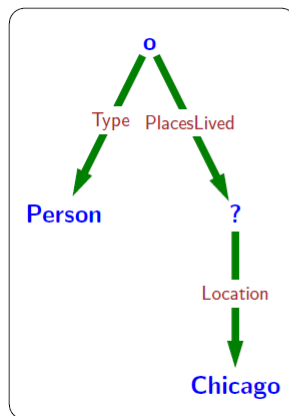
Type.Person \sqcap PlacesLived.Location.Chicago



Type.Person \sqcap PlacesLived.Location.Chicago



Type.Person \sqcap PlacesLived.Location.Chicago



Alignment: 文本短语

ReVerb on ClueWeb09 [Thomas Lin]:



(Barack Obama, *was born in*, Honolulu)
(Albert Einstein, *was born in*, Ulm)
(Barack Obama, *lived in*, Chicago)
... 15M triples ...

- Entities are linked to Freebase
- Hearst patterns used for unaries 15,000 text phrases



(Albert Einstein, PlaceOfBirth, Ulm)

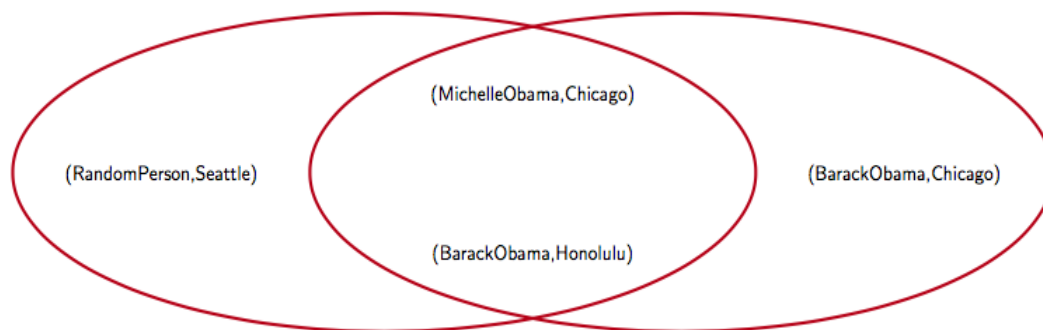
... 600M triples ...

Alignment: 短语到KB Predicates的匹配

Alignment features

phrase-count: 15,765
predicate-count: 9,182
intersection-count: 6,048
KB-best-match: 0

grew up in[Person,Location] DateOfBirth
born in[Person,Date] PlaceOfBirth
married in[Person,Date] Marriage.StartDate
born in[Person,Location] PlacesLived.Location



Lexicon: Mapping from phrases to predicates with features

Bridging操作

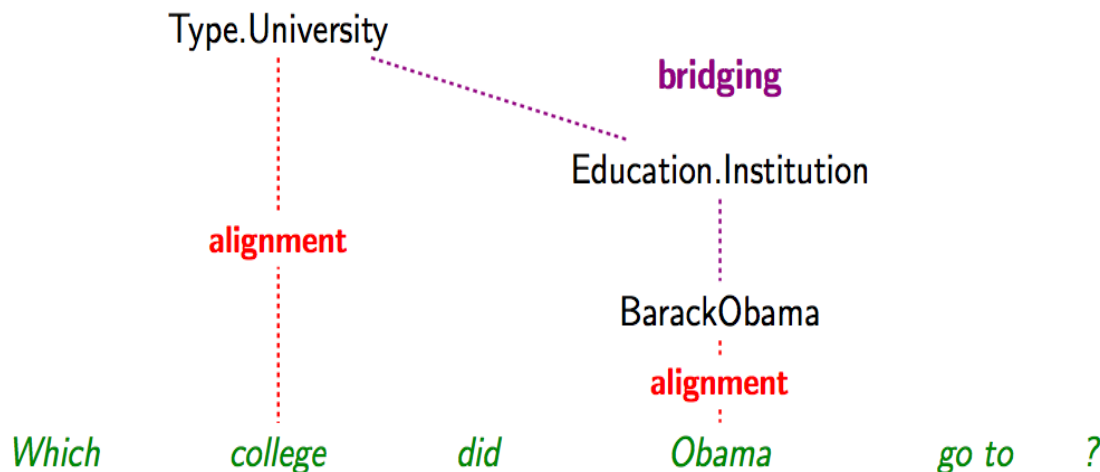
Often predicates are not expressed explicitly:

- What government does Chile have?
- What is Italy's language?
- Where is Beijing?
- What is the cover price of X-men?
- Who did Humphrey Bogart marry in 1928?

Alignment: build coarse mapping from raw text

Bridging: use neighboring predicates / type constraints

Bridging操作：连接2个Unaries

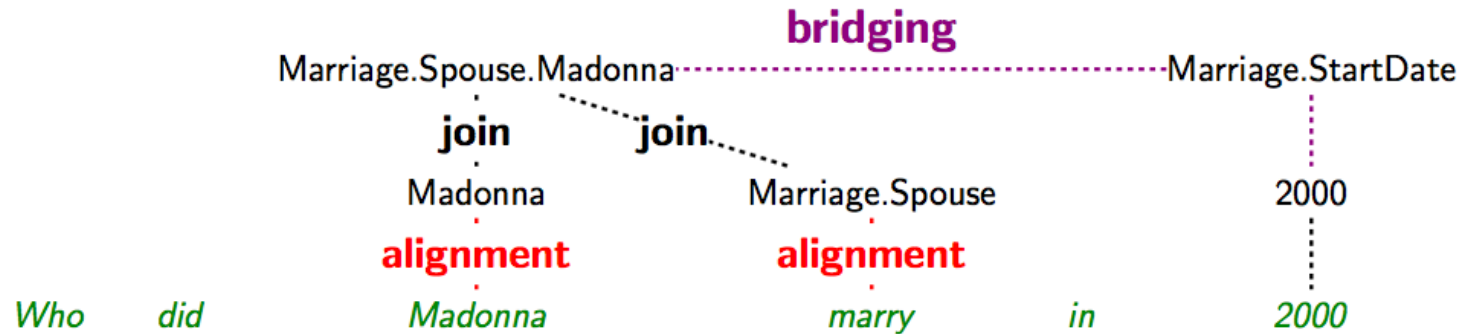


features

br-popularity	:11.37
br-two-unaries	: 1
br-education.institution:	1

Type.University \sqcap Education.Institution.BarackObama

Bridging操作: Event Modifiers

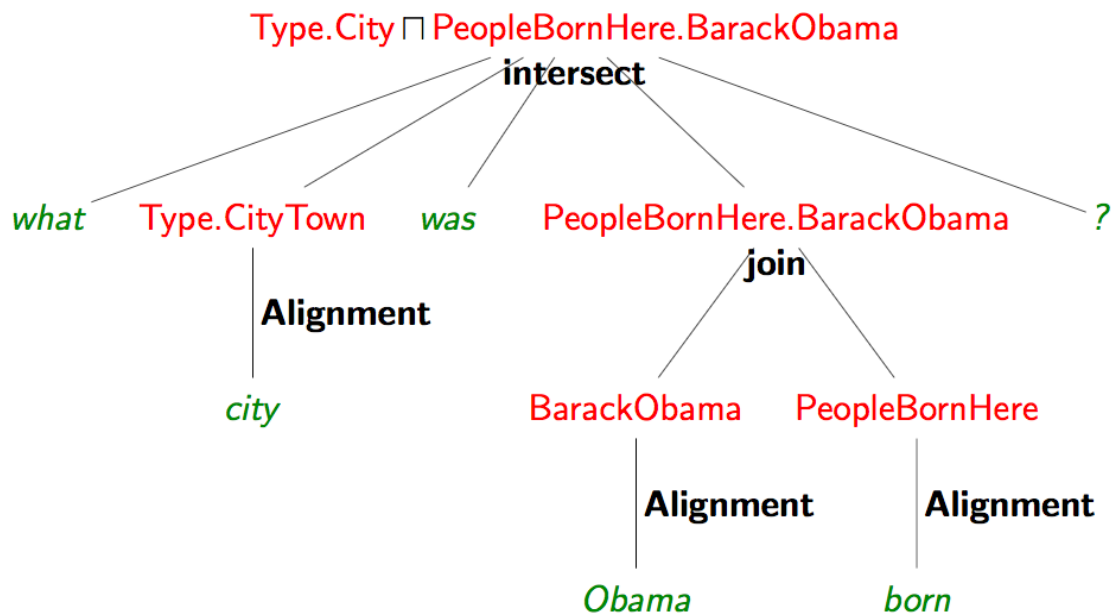


Marriage.(Spouse.Madonna \sqcap StartDate.2000)

features

br-popularity:	7.11
br-inject	: 1
br-startdate	: 1

通过Composition产生候选Logic Form (Derivation)



Derivations are constructed using an over-general grammar

训练模型来估计给定问句生成一个逻辑表达式的概率

Candidate derivations: $\mathcal{D}(x)$

Model: distribution over derivations d given utterance x

$$p(d \mid x, \theta) = \frac{\exp(\phi(x, d) \cdot \theta)}{\sum_{d' \in \mathcal{D}(x)} \exp(\phi(x, d') \cdot \theta)}$$

Training (estimating θ):

- Stochastic gradient descent (AdaGrad)

Features:

- Alignment and bridging
- lexicalized
- syntactic
- denotation

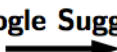
WebQuestions: 通过 Web 获得更多问题

Strategy: breadth-first search over Google Suggest graph

Where was *Barack Obama* born?

Where was _ born?  Barack Obama
Lady Gaga
Steve Jobs

Where was *Steve Jobs* born?

Where was *Steve Jobs* _?  born
raised
on the Forbes list

Where was *Steve Jobs* raised?

Result: popular web questions

Answers were obtained through crowdsourcing (AMT)

WebQuestions vs Free917

Free917 [Cai & Yates, 2013]: 917 examples, 2,036 word types

What is the engine in a 2010 Ferrari California?

What was the cover price of the X-men Issue 1?

- Generate questions based on Freebase facts

WebQuestions [our work]: 5,810 examples, 4,525 word types

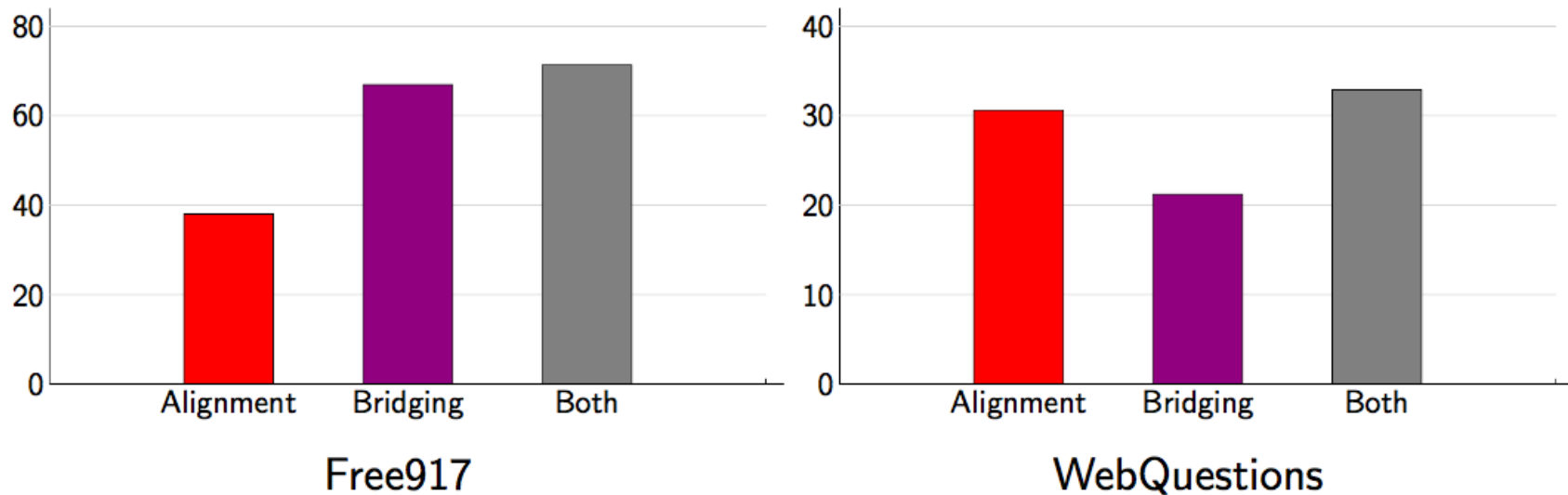
What character did Natalie Portman play in Star Wars?

What kind of money to take to Bahamas?

What did Edward Jenner do for a living?

- Generate questions from Google \Rightarrow less formulaic
-

Alignment和Bridging对Parsing性能的影响

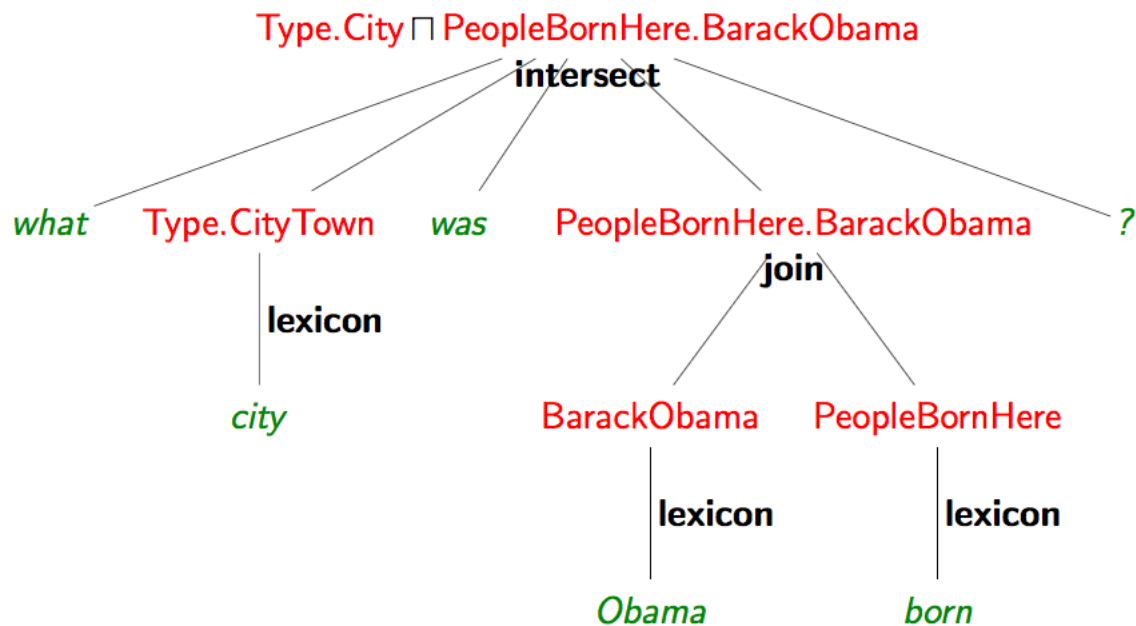


Conclusions:

Bridging more important for Free917

Alignment more important for WebQuestions

传统的语义解析



Lexicon depends on KB

Constructing a lexicon is challenging for large scale KBs

KB不完整问题

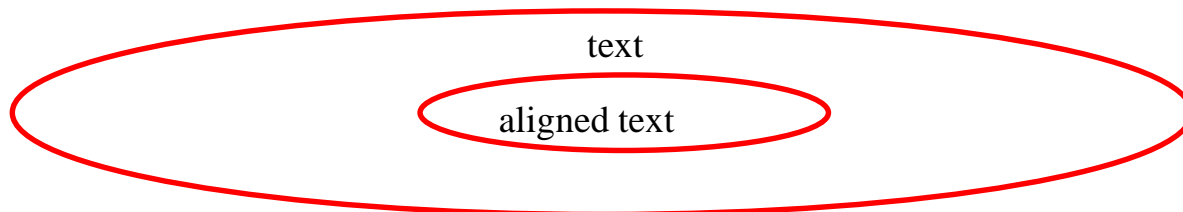
<i>St. Patrick's day happens on</i>	<i>March 17th</i>		<i>HolidayDate</i>
<i>St. Patrick's day occurs on</i>	<i>March 17th</i>	<i>StPatricksDay</i>	<i>March 17th</i>
<i>Purim happens on</i>	<i>March this year</i>	<i>Purim</i>	<i>Adar 14th</i>
<i>Purim occurs on</i>	<i>March</i>		
<i>Spring break happens usually around</i>	<i>March</i>		
<i>Spring break usually occurs around</i>	<i>March</i>		

happen \Rightarrow *HolidayDate* (1)

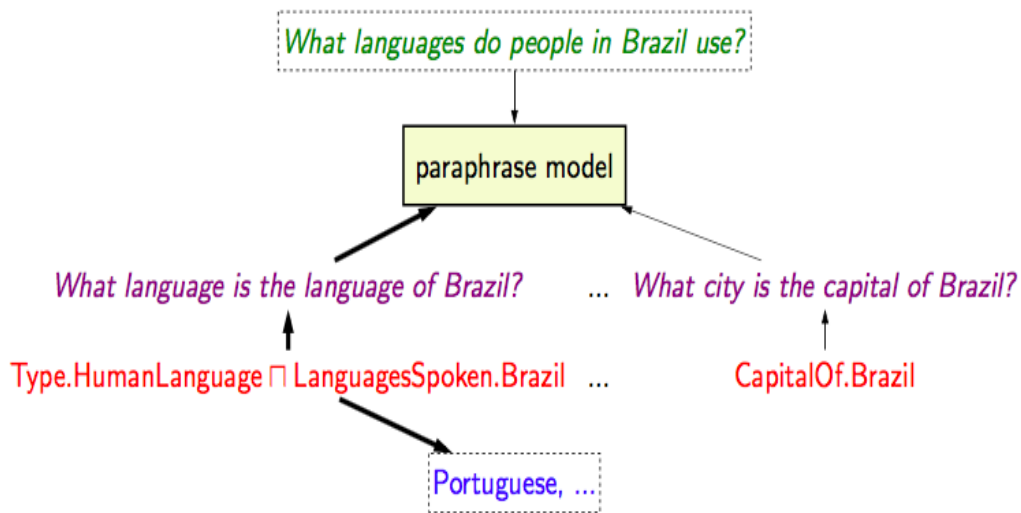
occur \Rightarrow *HolidayDate* (1)

occur \Leftrightarrow *happen* (3)

Only 2% of relation phrases on Reverb can be aligned to Freebase



KB不完整问题-使用Paraphrasing来解决



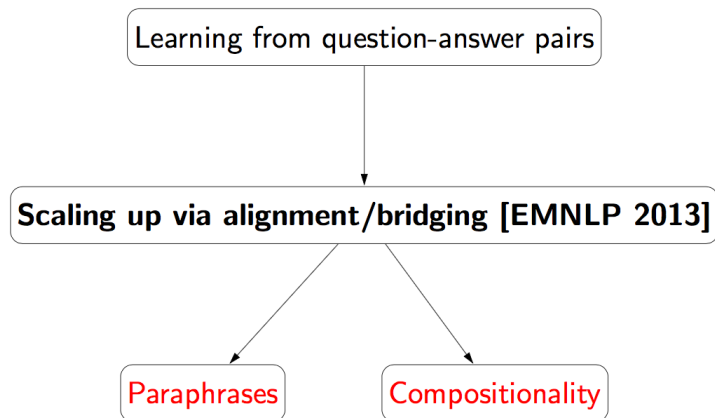
x : input question

Z_x : candidate logical forms

C_z : generated canonical utterances

y : answer

- Simple model suggests candidate logical forms
- Simple model generates canonical utterances
- Ranking of canonical utterances



模型定义

Model: distribution over logical forms and canonical utterances

$$p_{\theta}(c, z \mid x) = \frac{\exp(\phi(x, c, z)^{\top} \theta)}{\sum_{z' \in Z_x, c' \in C_z} \exp(\phi(x, z', c')^{\top} \theta)}$$

Decomposition to paraphrase model and logical form model:

$$\phi(x, c, z) = \phi_{\text{pr}}(x, c) + \phi_{\text{lf}}(z)$$

x : input question

Z_x : candidate logical forms

C_z : generated canonical utterances

y : answer

Need to estimate parameters θ_{pr} and θ_{lf}

模型学习

Training data: $\{(x_i, y_i)\}_{i=1}^n$

Objective function:

$$p_{\theta}(y \mid x) = \sum_{z \in Z_x: y=[z]_{\mathcal{K}}} \sum_{c \in C_z} p_{\theta}(c, z \mid x)$$

$$O(\theta) = \sum_{i=1}^n \log p_{\theta}(y_i \mid x_i) - \lambda \|\theta\|_1$$

x : input question

Z_x : candidate logical forms

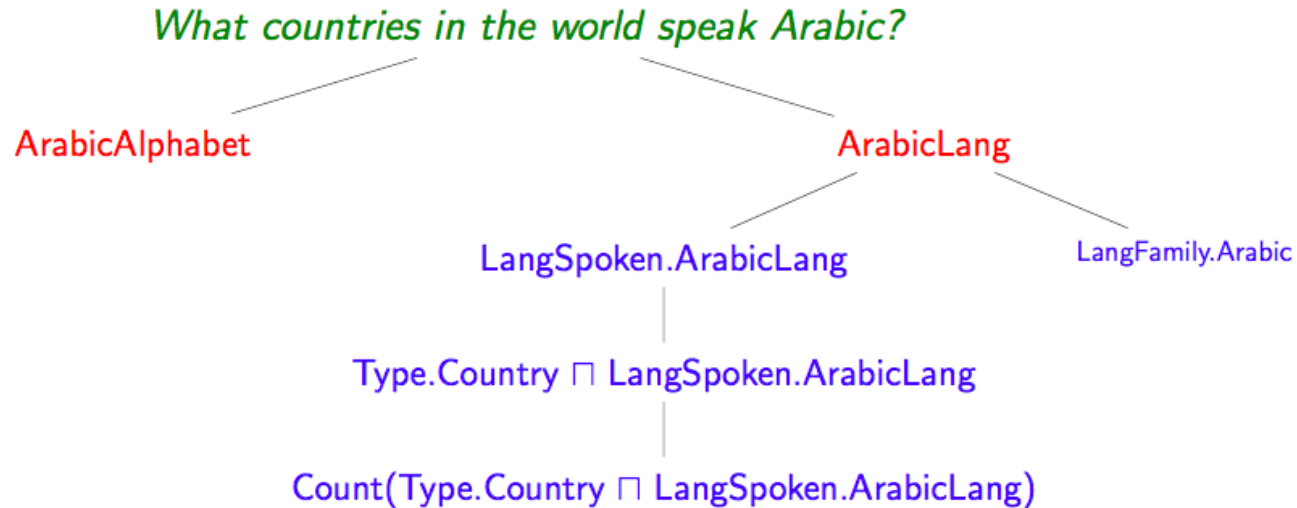
C_z : generated canonical utterances

y : answer

Training: AdaGrad (Duchi et al., 2010)

候选逻辑表达式

Grow logical forms around entities



候选逻辑表达式

Template	Example	Question
$p.e$	Directed.TopGun	<i>who directed Top Gun</i>
$p_1.p_2.e$	Employment.EmployerOf.SteveBalmer	<i>Where does Steve Balmer work?</i>
$p.(p_1.e_1 \sqcap p_2.e_2)$	Character.(Actor.BradPitt \sqcap Film.Troy)	<i>Who did Brad Pitt play in Troy?</i>
$\text{Type}.t \sqcap z$	Type.Composer \sqcap SpeakerOf.French	<i>What composers spoke French?</i>
$\text{count}(z)$	count(BoatDesigner.NatHerreshoff)	<i>How many ships were designed by Nat Herreshoff?</i>

650 logical forms on average per example

65% recall

候选问句生成

Type.Country □ LangSpoken.ArabicLang
country spoken in languages spoken Arabic language



syntactic analysis

What country is Arabic language spoken in?

What country spoken the languages Arabic language?

候选问句生成

	$d(p)$ Categ.	Rule	Example
$p.e$	NP	WH $d(t)$ has $d(e)$ as NP ?	What <i>election contest</i> has <i>George Bush</i> as winner?
	VP	WH $d(t)$ (AUX) VP $d(e)$?	What <i>radio station</i> serves area <i>New-York</i> ?
	PP	WH $d(t)$ PP $d(e)$?	What <i>beer</i> from region <i>Argentina</i> ?
	NP VP	WH $d(t)$ VP the NP $d(e)$?	What <i>mass transportation system</i> served the area <i>Berlin</i> ?
$R(p).e$	NP	WH $d(t)$ is the NP of $d(e)$?	What <i>location</i> is the <i>place of birth</i> of <i>Elvis Presley</i> ?
	VP	WH $d(t)$ AUX $d(e)$ VP ?	What <i>film</i> is <i>Brazil</i> featured in?
	PP	WH $d(t)$ $d(e)$ PP ?	What <i>destination</i> <i>Spanish</i> steps near travel destination?
	NP VP	WH NP is VP by $d(e)$?	What <i>structure</i> is <i>designed</i> by <i>Herod</i> ?

generation based on

- Entity is subject or object
- Parse tree of predicates' descriptions

1500 canonical utterances on average per example

Paraphrase模型

□ 映射同义句

What countries in the world speak Arabic?

What country is Arabic language spoken in?

□ 简单的paraphrase模型利用大量文本做统计

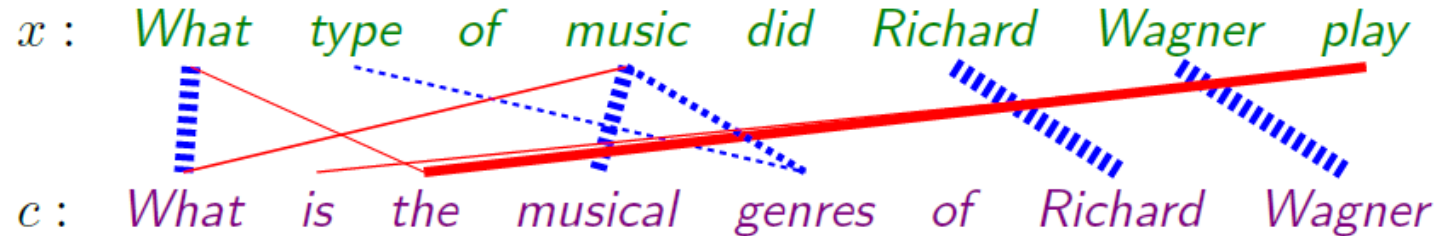
- Association model (关联模型)—Paralex

- Vector space model (向量空间模型)—Wikipedia

Association model (关联模型)

Association: pair of spans $(x_{ij}, c_{i'j'})$

type of music \Leftrightarrow musical genre



Generate all associations and extract features:

identical lemma	3
type of music \wedge musical genre	1
play \wedge the	1
WN derivation	1
delete IN	1
delete of	1
...	...

关联生成

Paralex dataset (Fader et al., 2013)

- 18M word aligned question pairs
- Generated through links in WikiAnswers

Who wrote the Winnie the Pook books? Who is poohs creator?

What relieves a hangover?

What is the best cure for a hangover?

How do you say Santa Clause in Sweden? Say santa clause in sweden?

Consistent phrase pair heuristic (Och and Ney, 2004):

type of music ⇔ *musical genre*

born in ⇔ *birth place*

Associate words with same POS tag or that are linked through WordNet

Vector space model (向量空间模型)

Associations disadvantage: coverage

Train word vectors $v(w)$:

C: content words in utterance x

$$v(x) = \frac{1}{|C|} \sum_{x_i \in C} v(x_i)$$

Learn a matrix W to estimate “similarity” score

$$s(x, c) = v(x)^\top W v(c)$$

Options for W

- Identity: dot product
 - Diagonal: dot product with scaling
 - Full matrix: interactions between dimensions
-

Association vs. vector space

x : What type of music did Richard Wagner play?

as : What is the musical genres of Richard Wagner?

vs : What composition has Richard Wagner as lyricist?

x : Where is made Kia car?

as : What place is founded by Kia motors?

vs : What city is Kia motors a headquarters of?

Building a Semantic Parser Overnight

Domain



(1) by builder (~30 minutes)

Seed lexicon

article → TYPENP[article]
 publication date → RELNP[publicationDate]
 cites → VP/NP[cites]
 ...



(2) via domain-general grammar

Logical forms and canonical utterances

article that has the largest publication date
 $\text{argmax}(\text{type.article}, \text{publicationDate})$
person that is author of the most number of article
 $\text{argmax}(\text{type.person}, \mathbf{R}(\lambda x. \text{count}(\text{type.article} \sqcap \text{author}.x)))$
 ...



(3) via crowdsourcing (~5 hours)

Paraphrases

what is the newest published article?
who has published the most articles?
 ...



(4) by training a paraphrasing model

Semantic parser

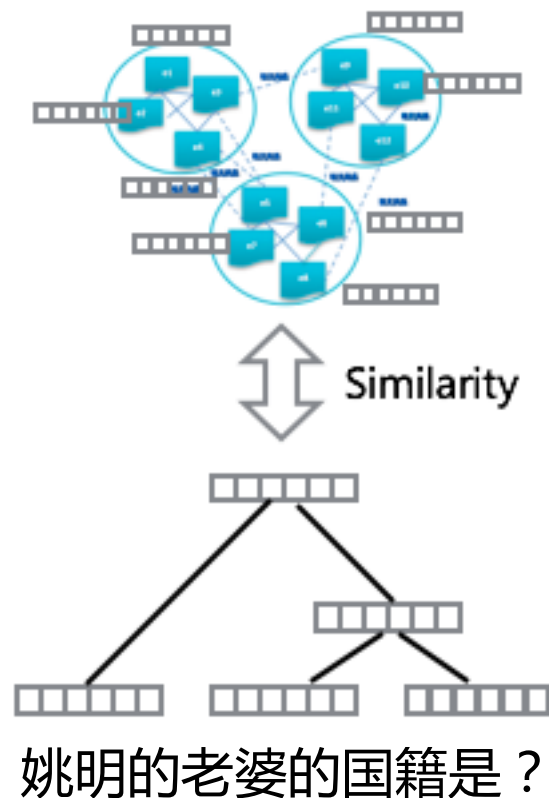
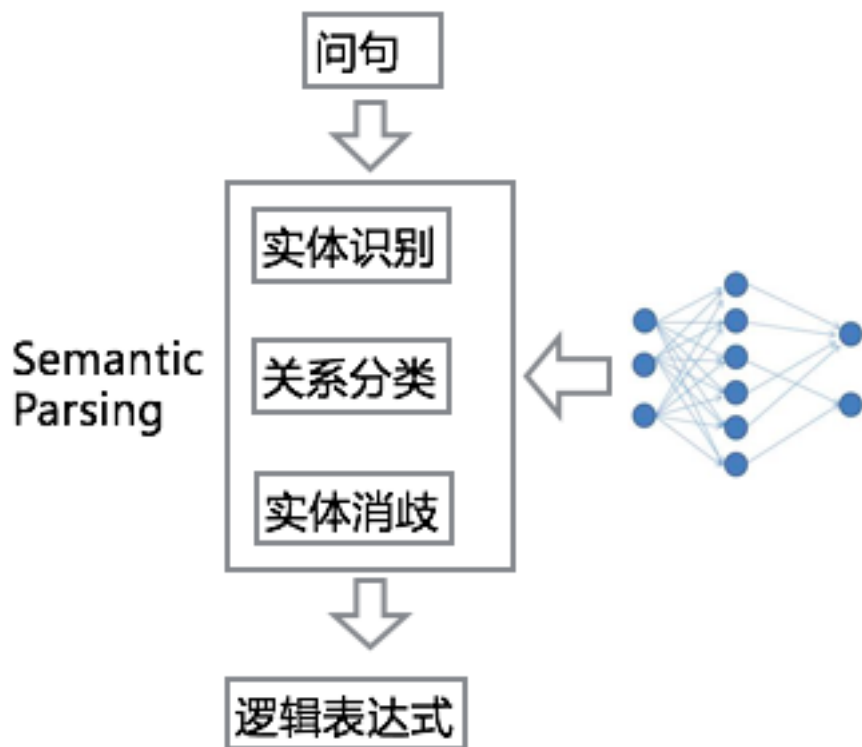
	[glue]	
(G1)	ENTITYNP[x]	→ NP[x]
(G2)	TYPENP[x]	→ NP[type.x]
(G3)	NP[x] CP[f] (and CP[g])*	→ NP[x \sqcap f \sqcap g]
	[simple]	
(R0)	that VP[x]	→ CP[x]
(R1)	whose RELNP[r] CMP[c] NP[y]	→ CP[r.c.y]
	is is not is smaller than is larger than is at least is at most	→ CMP[= \neq < > \leq \geq]
(R2)	that (not)? VP/NP[r] NP[y]	→ CP[(\neg)r.y]
(R3)	that is (not)? RELNP[r] of NP[y]	→ CP[(\neg) $\mathbf{R}(r).y$]
(R4)	that NP[y] (not)? VP/NP[r]	→ CP[(\neg)($\mathbf{R}(r).y)$]
	[counting]	
(C1)	that has CNT[c] RELNP[r]	→ CP[$\mathbf{R}(\lambda x. \text{count}(\mathbf{R}(r).x)).c$]
(C2)	that VP/NP[r] CNT[c] NP[y]	→ CP[$\mathbf{R}(\lambda x. \text{count}(y \sqcap \mathbf{R}(r).x)).c$]
(C3)	that is RELNP[r] of CNT[c] NP[y]	→ CP[$\mathbf{R}(\lambda x. \text{count}(y \sqcap r.x)).c$]
(C4)	that CNT[c] NP[y] VP/NP[r]	→ CP[$\mathbf{R}(\lambda x. \text{count}(y \sqcap r.x)).c$]
	(less than more than) NUM[n]	→ CNT[($<$ $>$ \geq)n]
	[superlatives]	
(S0)	NP[x] that has the largest RELNP[r]	→ NP[arg max(x, r)]
(S1)	NP[x] that has the most number of RELNP[r]	→ NP[arg max(x, $\mathbf{R}(\lambda y. \text{count}(\mathbf{R}(r).y))$)]
(S2)	NP[x] that VP/NP[r] the most number of NP[y]	→ NP[arg max(x, $\mathbf{R}(\lambda y. \text{count}(\mathbf{R}(r).y))$)]
(S3)	NP[x] that is RELNP[r] of the most number of NP[y]	→ NP[arg max(x, $\mathbf{R}(\lambda z. \text{count}(y \sqcap r.z))$)]
(S4)	NP[x] that the most number of NP[y] VP/NP[r]	→ NP[arg max(x, $\mathbf{R}(\lambda z. \text{count}(y \sqcap r.z))$)]
	[transformation]	
(T1)	RELPNP[r] of NP[y]	→ NP[$\mathbf{R}(r).y$]
(T2)	RELPNP ₀ [h] CP[f] (and CP[g])*	→ NP[$\mathbf{R}(h).(f \sqcap g)$]
(T3)	RELPNP[r] of RELNP ₀ [h] NP[x] CP[f] (and CP[g])*	→ NP[$\mathbf{R}(r).(h.x \sqcap f \sqcap g)$]
(T4)	NP[x] or NP[y]	→ NP[x \sqcup y]
	[aggregation]	
(A1)	number of NP[x]	→ NP[count(x)]
(A2)	total average RELNP[r] of NP[x]	→ NP[sum average(x, r)]

语义分析方法面临的挑战

- 开放域环境下如何进行Semantic Parsing
 - 词典的获取?
 - 问题答案对的获取、人工模板
 - 消歧?
 - PCCG、MLN
 - 符号匹配?

KBQA与深度学习结合的两个方向

- 利用深度学习对于传统问答方法进行改进
- 基于深度学习的End2End模型



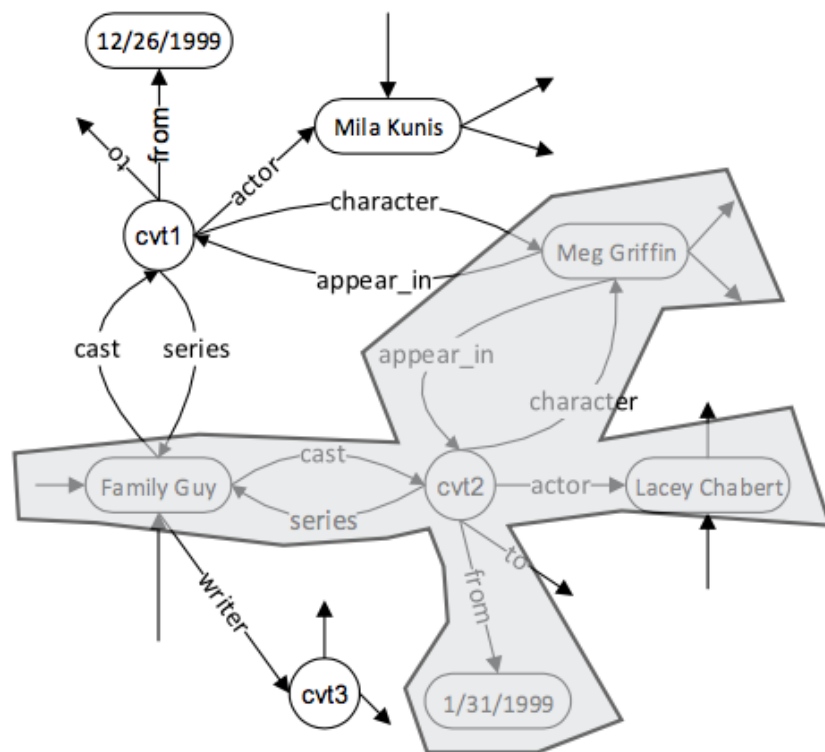
Semantic Parsing via Staged Query Graph Generation

- 问答过程：基于结构化的问句语义表达 (Lambda演算) 在知识图谱匹配最优子图

Who first voiced Meg on Family Guy?

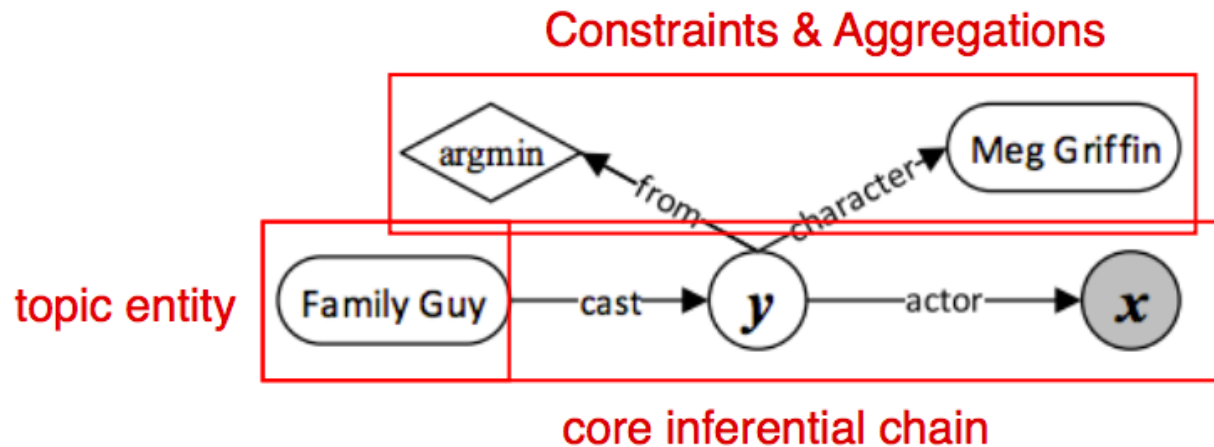


$\text{argmin}(\lambda x. \text{Actor}(x, \text{Family_Guy})$
 $\wedge \text{Voice}(x, \text{Meg_Griffin}), \lambda x. \text{casttime}(x))$



Semantic Parsing via Staged Query Graph Generation

Who first voiced Meg on Family Guy?

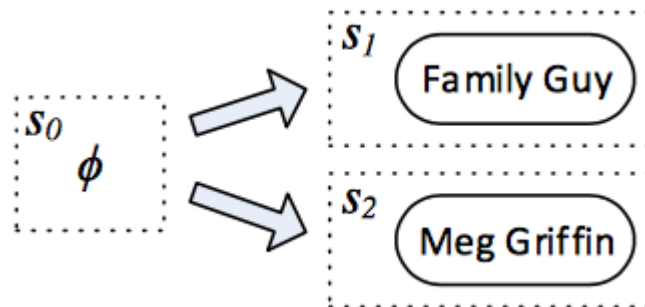


1. 知识库实体，在图中用圆角矩形表示。
2. 中间变量，在图中用白底圆圈表示。
3. 聚合函数，用菱形表示。
4. lambda变量 (答案)，在图中用灰底圆圈表示。

链接Topic Entity并识别核心推断链

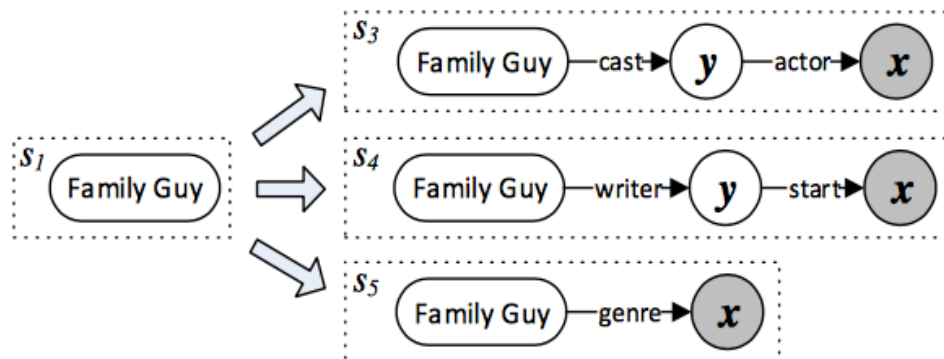
Who first voiced Meg on Family Guy?

候选主题词：S1和S2



将其在KB中周围的节点提取出来作为候选路径。

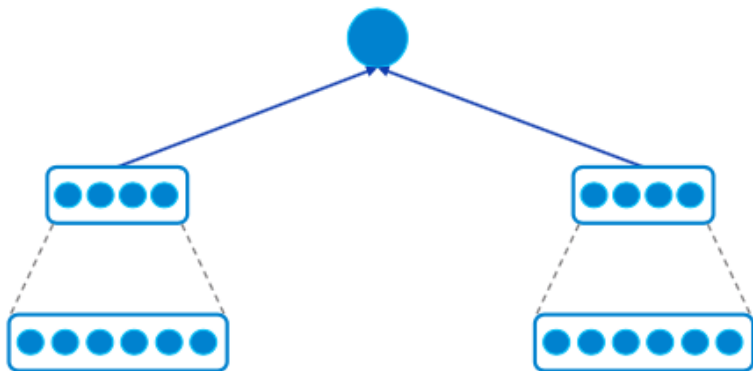
- 周围长度为1的路径 (S5)
- 周围长度为2且包含CVT节点的路径 (S3,S4), 如: cast-actor



采用CNN对候选路径进行打分

- 输入：自然语言和候选路径
- 对二者分别经过两个不同的卷积神经网络，输出得到一个300维的分布式表示
- 然后利用向量间的相似度（如cosine距离）计算自然语言和谓语句序列的相似度得分

$$p(r_1|q) = \frac{\exp(\cos(e_{r_1}, e_q))}{\sum_r \exp(\cos(e_r, e_q))}$$



Cast-Actor

Who first voiced Meg on **Family Guy**?

Semantic layer: y

Semantic projection matrix: W_s

Max pooling layer: v

Max pooling operation

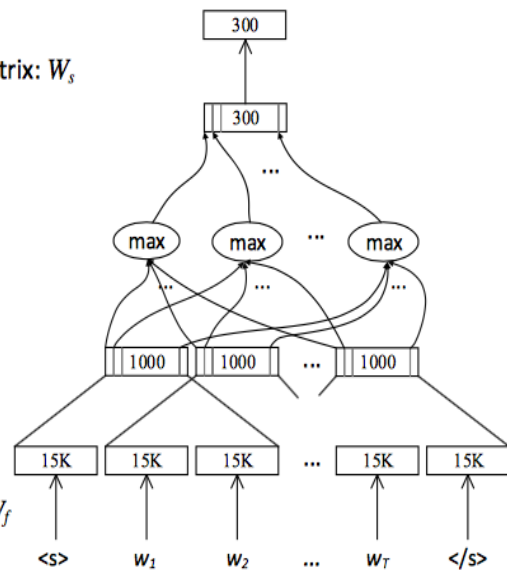
Convolutional layer: h_i

Convolution matrix: W_c

Word hashing layer: f_i

Word hashing matrix: W_f

Word sequence: x_t

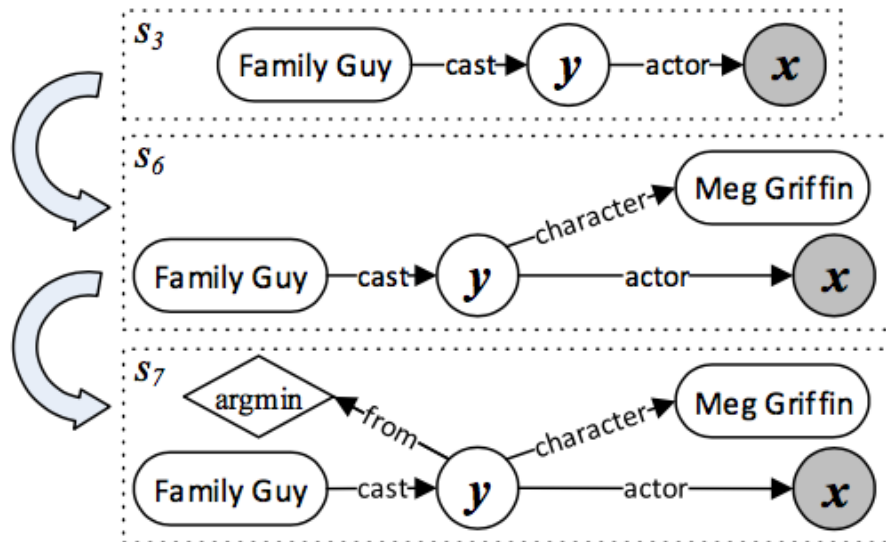


word hash: "cat" \rightarrow "#ca", "cat", "at#"

"cat" \rightarrow [0....1..1....0....1....0]

Argument Constraints

Who first voiced Meg on Family Guy ?



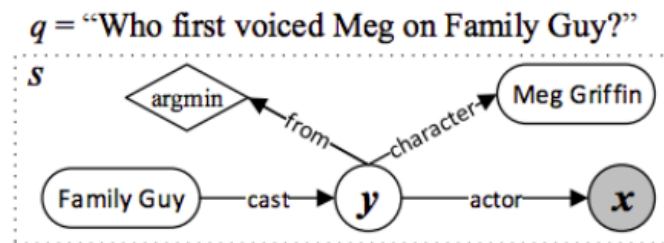
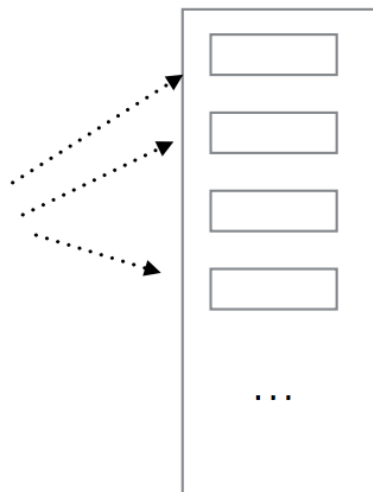
Using rules to add constraints on the core inferential chain

If x is an entity, it can be added as an entity node

If x is such keywords, like "first", "latest", it could be added as aggregation constraints

排序

Who first voiced **Meg** on **Family Guy**?



- (1) EntityLinkingScore(FamilyGuy, "Family Guy") = 0.9
- (2) PatChain("who first voiced meg on <e>", cast-actor) = 0.7
- (3) QuesEP(q , "family guy cast-actor") = 0.6
- (4) ClueWeb("who first voiced meg on <e>", cast-actor) = 0.2
- (5) ConstraintEntityWord("Meg Griffin", q) = 0.5
- (6) ConstraintEntityInQ("Meg Griffin", q) = 1
- (7) AggregationKeyword(argmin, q) = 1
- (8) NumNodes(s) = 5
- (9) NumAns(s) = 1

- Main Features:
 - Topic Entity: Entity Linking Score
 - Core Inferential Chain: Relation Matching Score (NN-based model)
 - Constraints: Keyword and entity matching

关键: 关系分类

Who first voiced Meg on Family Guy?



{cast-actor, writer-start, genre}

Feature
Representation

Labeled
Training Data

传统特征表示

- 传统特征提取需要NLP预处理+人工设计的特征

The [haft]_{e1} of the [axe]_{e2} is made of yew wood

Component-Whole(e1,e2)

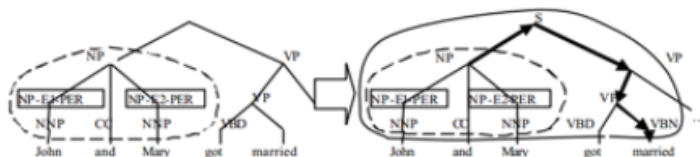
Traditional Features

Words : *haft_{m11}, of_{b1}, the_{b2}, axe₂₁*

Entity Type : *OBJECT_{m1}, PRODUCT_{m2}*

Parse Tree : *OBJECT-NP-PP-PRODUCT*

Kernel Feature:

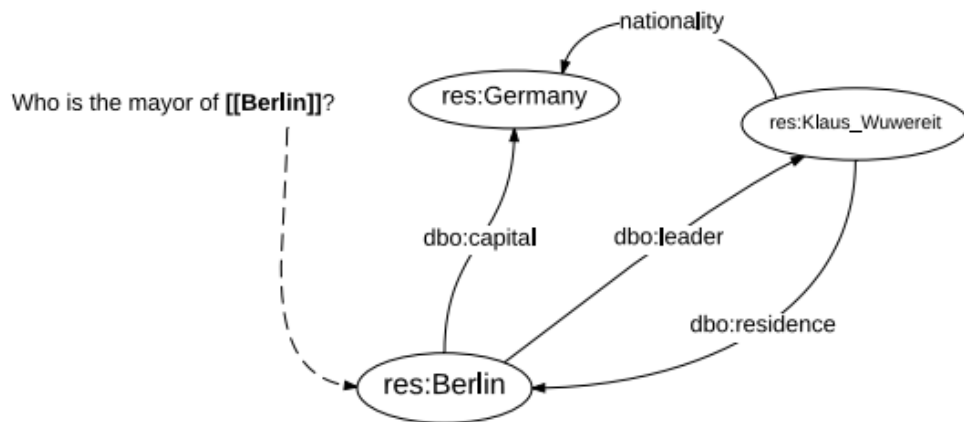


问题1：对于缺少NLP处理工具和资源的语言，无法提取文本特征

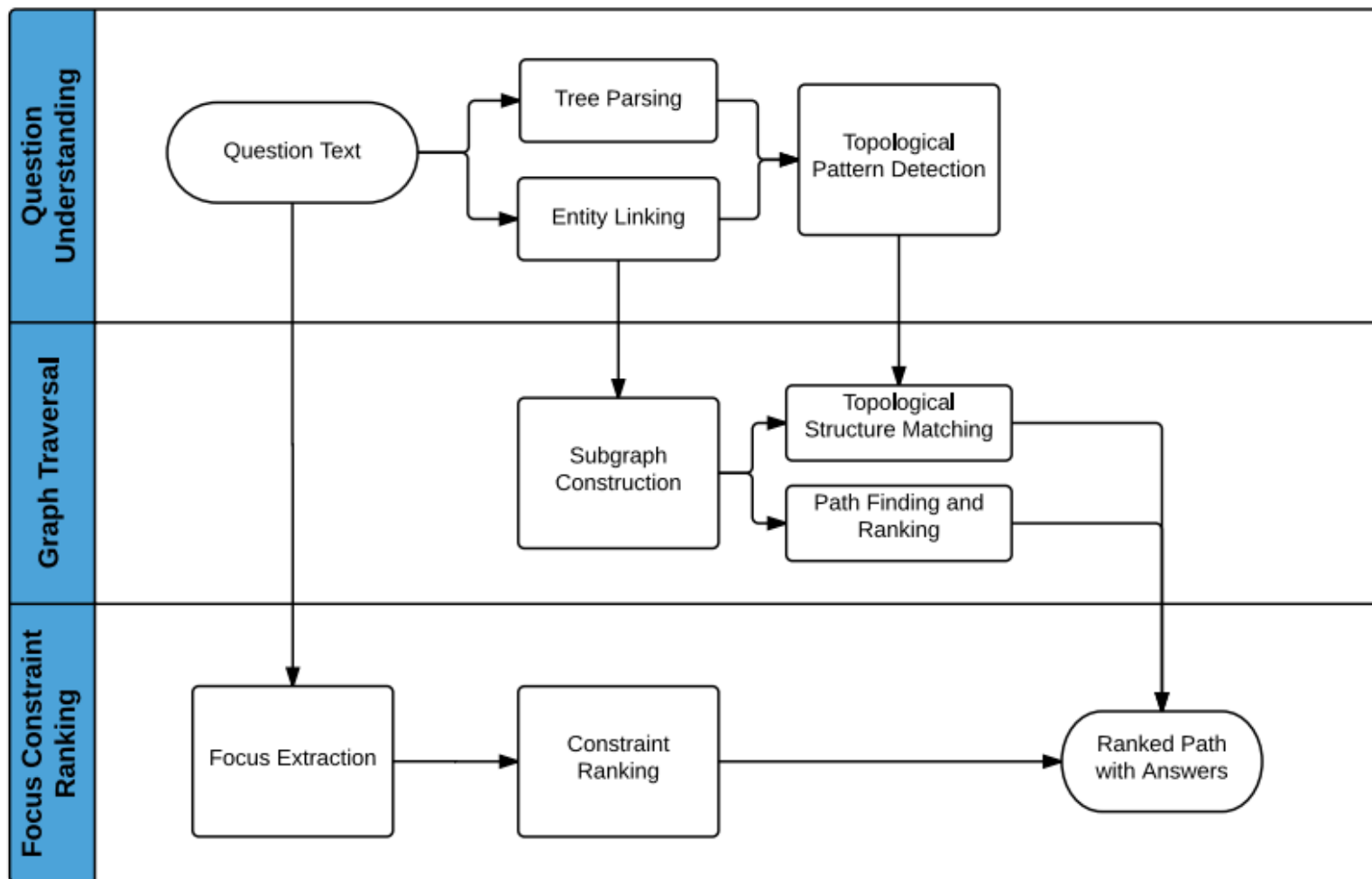
问题2：NLP处理工具引入的“错误积累”

Graph-Traversal-Based QA (图遍历QA)

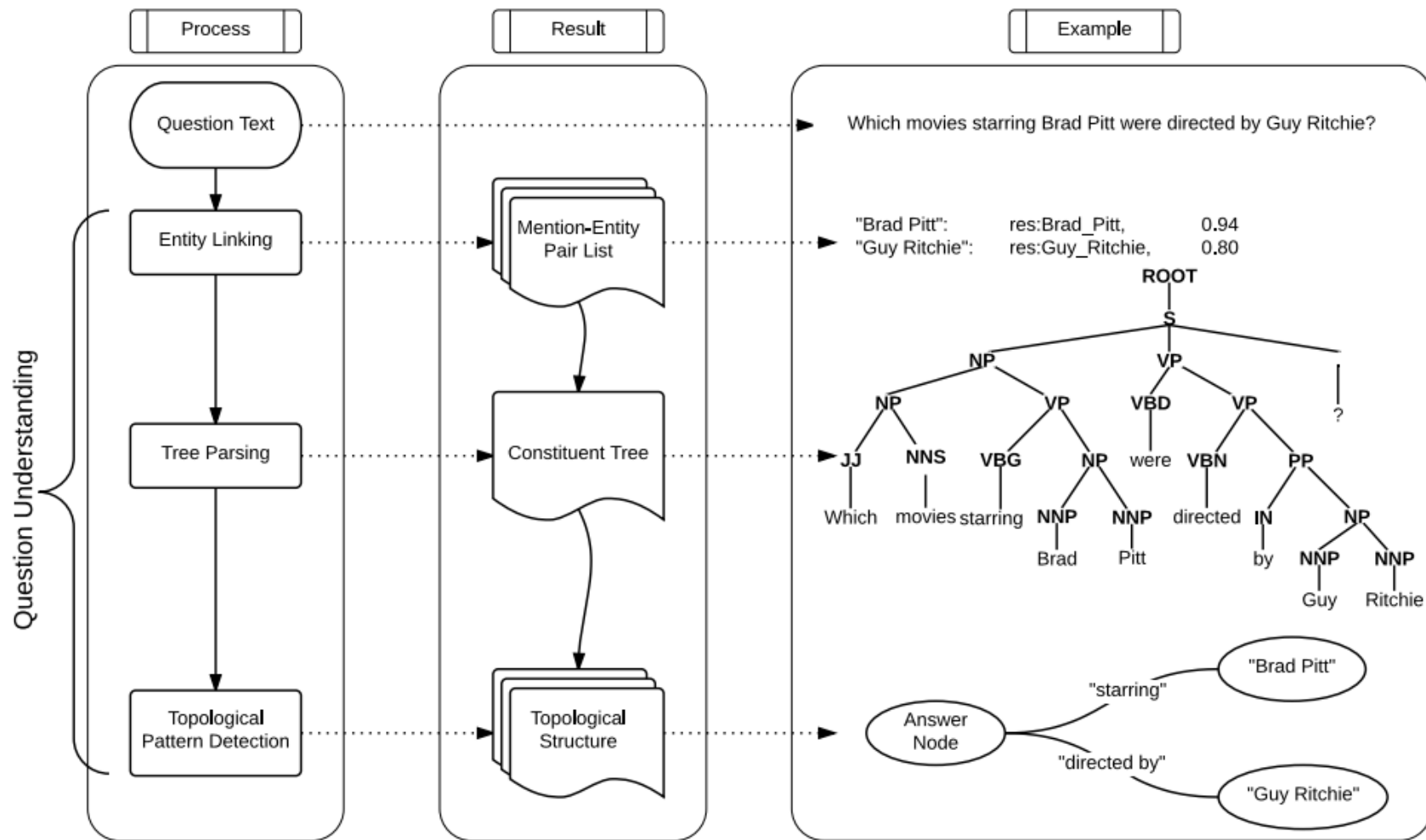
- 深度学习End2End方法前身：基于图遍历的QA方法
- 与End2End方法的区别仅在于改变了ranking和mapping函数



图遍历QA整体架构



问句理解



问句理解

□ 实体链接 (Entity linking):

- To detect mentions in the query and link them to the resources in the knowledge base , with a global threshold of 0.15
- By using Wikipedia Miner tool to do the entity linking

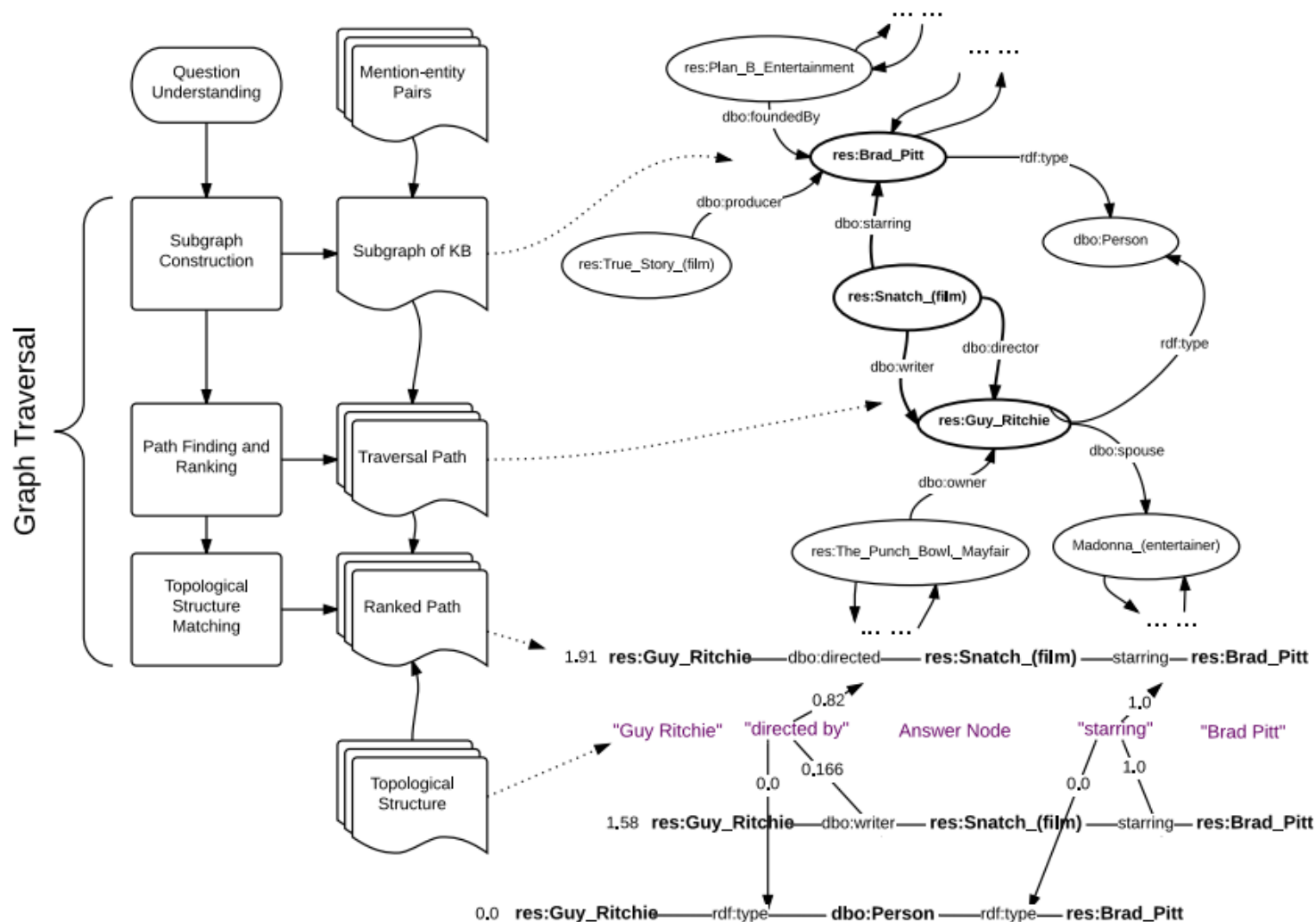
□ 拓扑结构抽取 (Topological structure extraction)

- parse the question text and generate the corresponding constituent tree
- To extracts relationships by using topological patterns

Table 1. Topological Pattern List

ID	Pattern	Example	Extraction Result
1	VB → VB+NP	Who produces Orangina?	ANSNODE - “produces” - “Orangina”
2	VP → VB+PP	Which television shows were created by John Cleese?	ANSNODE - “created by” - “John Cleese”
3	NP → NP+PP	Who is the mayor of Berlin?	ANSNODE - “mayor of” - “Berlin”
4	SQ → VB+NP+VP	When was Alberta admitted as province?	ANSNODE - “admitted as province” - “Alberta”

Graph Traversal (图遍历)



Graph Traversal (图遍历)

☐ Subgraph construction:

- Iteratively construct a subgraph from linked entities in the knowledge base with most K steps

☐ Path finding:

- Start from a known entity e and make one step ahead and obtain many one step paths rooted at e
- Calculates the semantic similarity between the predicates around the entity e and the phrase text from the edge in the topological structure
- Iteratively make more steps outward and get many multiple-length paths
 - ☐ When to stop: When it gets to the outmost layer of the subgraph
 - ☐ Discard the obtained path which does not match the topological structure

Focus Constraint

☐ Focus Definition:

- A focus is a phrase in the question text describing the answer directly
e.g. "television shows" is the focus of "Which television shows were created by John Cleese?"

☐ Rules to find focus:

- Derive focus information from interrogatives
 - ☐ person and organization from "who"
 - ☐ place from "where"
 - ☐ date from "when"
- Extract a focus based on the POS tags of a question
 - ☐ the longest noun phrase after the interrogative part of a question
 - ☐ the interrogative part could be "Give me all" , "What" , "Which" and so on
 - ☐ the first word after the interrogative part with the POS tag "NN" to the last word having continuous "NN" tag as the focus

Path Ranking

Path score

-composed of predicate score and type score

$$PathScore = \frac{1}{m} \sum_{i=1}^m PredicateScore_i + TypeScore$$

- **Predicate ranking:**

- Consider the semantic similarity between a predicate and the phrase text extracted from the topological structure
- Use the UMBC Semantic Similarity Service^[1] to calculate the semantic similarity between two words
- Question: how to add some information from focus constraint to the predicates leading to the final answers is not mentioned in algorithm

- **Path ranking:**

- One step path ranking: score is simply the sum of the predicate ranking score and the answer type score generating from the focus constraint
- Two step path ranking: need more work
- the maximal size of candidate predicates is set to 5

基于端到端知识库问答 (Bordes et al. 2014)

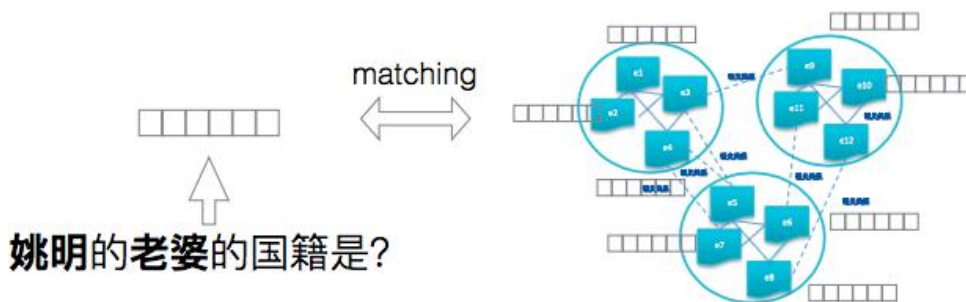
目前只处理单关系 (Single Relation) 的问句 (Simple Question)

基本步骤

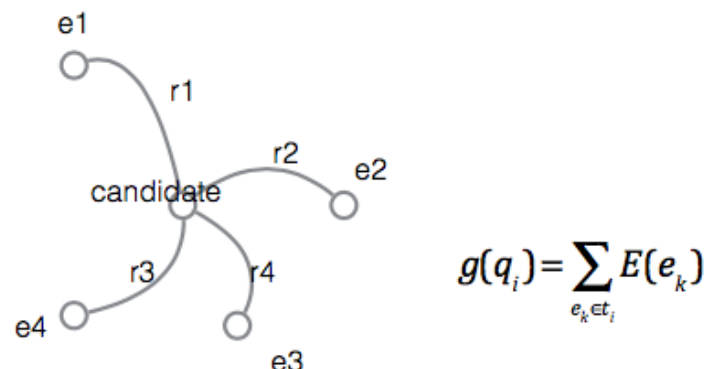
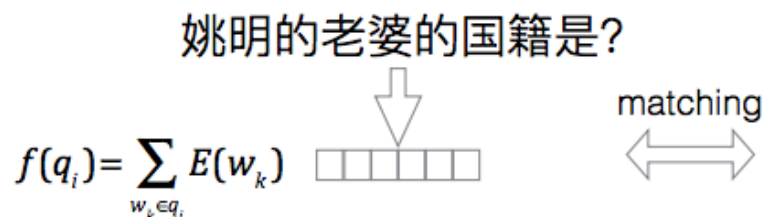
Step1: 候选生成

- 利用Entity Linking找到main entity
- 在KB中main entity周围的entity均是候选

Step2: 候选排序



基于端到端知识库问答 (Bordes et al. 2014)



Object: $L = 0.1 - S(f(q), g(t)) + S(f(q), g(t'))$

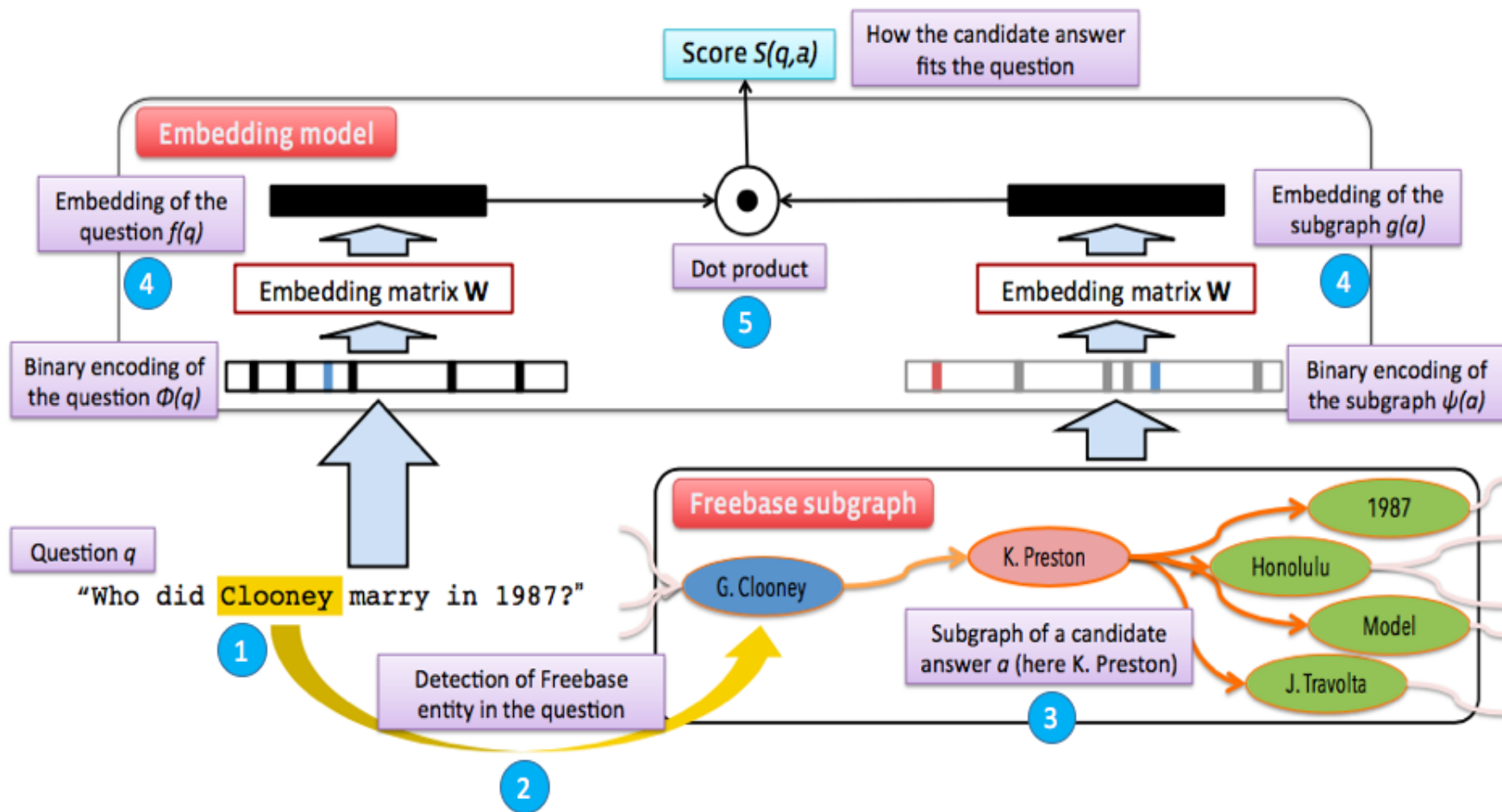
$$S(f(q), g(t)) = f(q)^T g(t)$$

$$S(f(q), g(t)) = f(q)^T M g(t)$$

Multitask learning with paraphrases:

$$S_p(f(q), f(q_p)) = f(q)^T f(q_p)$$

基于端到端知识库问答 (Bordes et al. 2014)



基于端到端知识库问答 (Bordes et al. 2014)

① 利用Entity Linking定位问题中的main entity

$$E(e_k)$$

② 找到从问题实体到答案实体的路径

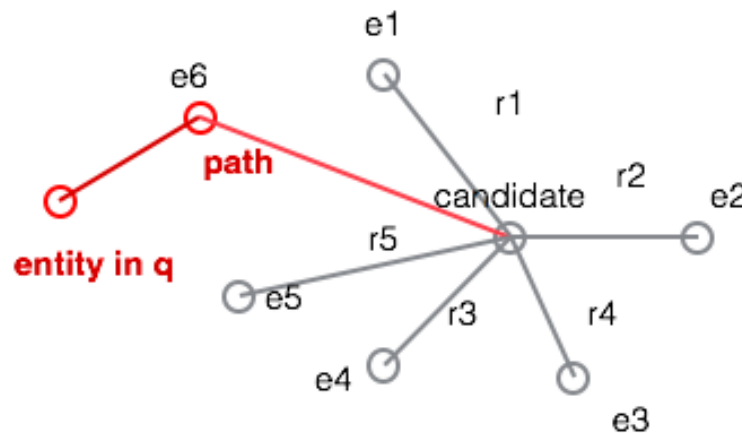
$$g(q_i) = \sum_{e_k \in \text{Path}(t_i)} E(e_k) + \sum_{r_k \in \text{Path}(t_i)} E(r_k)$$

③ 生成候选答案的子图

$$g(q_i) = \sum_{e_k \in \text{Context}(t_i)} E(e_k) + \sum_{r_k \in \text{Context}(t_i)} E(r_k)$$

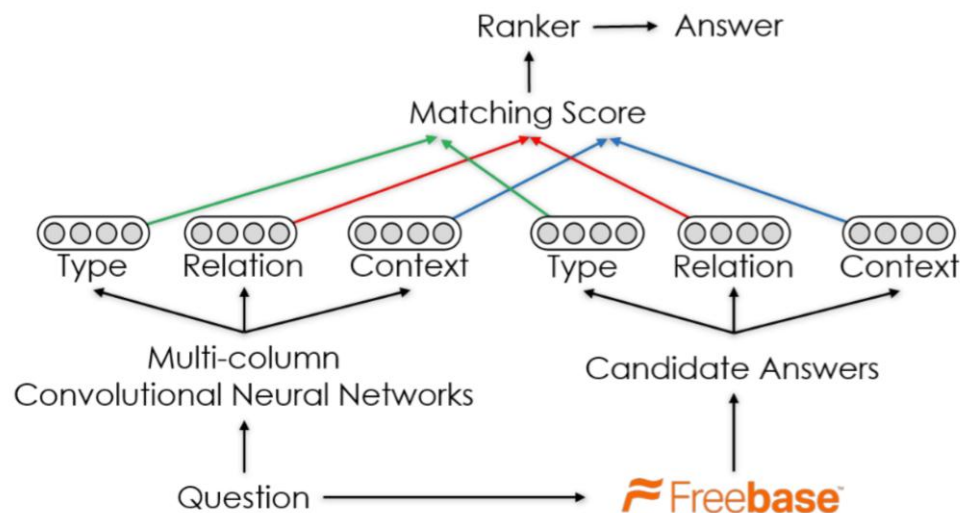
④ 将问题和答案子图分别映射成embedding, 学习出embedding向量;

⑤ 点积操作 (dot product) 获得问题和候选答案之间的相似度分值。



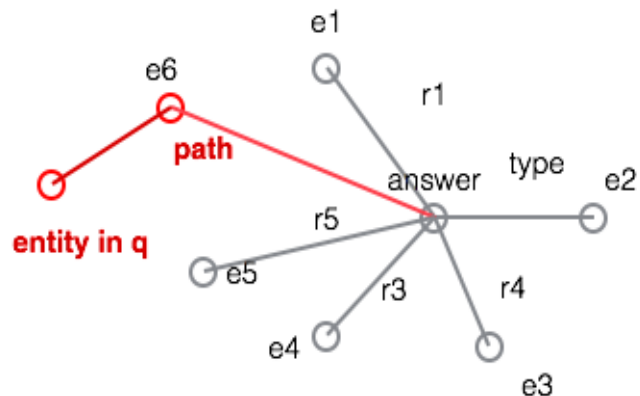
Multi-Column CNN (Dong et al. 2015)

- 依据问答特点，考虑答案不同维度的信息



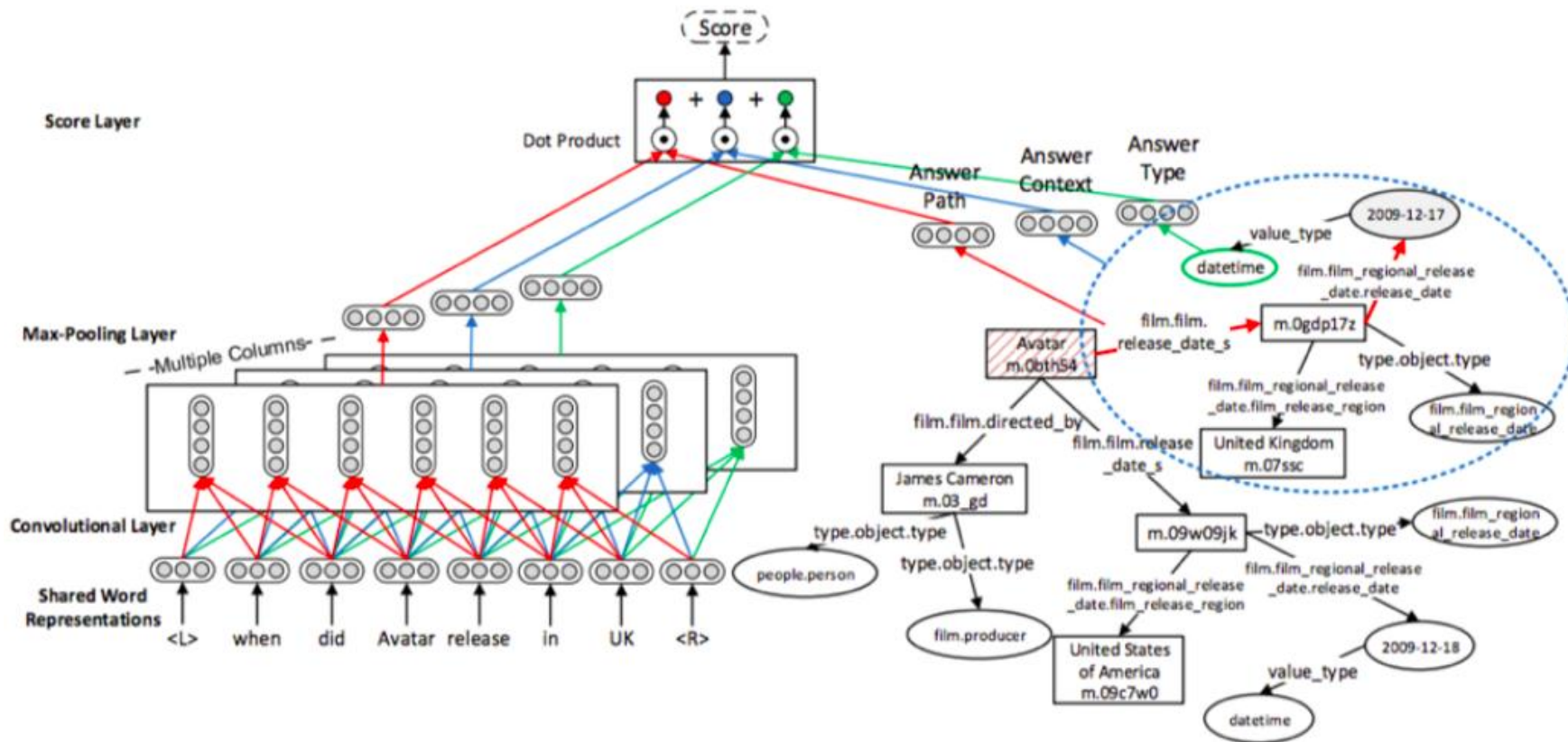
$$S(q, a) = \underbrace{\mathbf{f}_1(q)^T \mathbf{g}_1(a)}_{\text{answer path}} + \underbrace{\mathbf{f}_2(q)^T \mathbf{g}_2(a)}_{\text{answer context}} + \underbrace{\mathbf{f}_3(q)^T \mathbf{g}_3(a)}_{\text{answer type}}$$

Answer Type
Answer Context
Answer Path



Framework

when did Avatar release in UK



Results

Method	F1	P@1
(Berant et al., 2013)	31.4	-
(Berant and Liang, 2014)	39.9	-
(Bao et al., 2014)	37.5	-
(Yao and Van Durme, 2014)	33.0	-
(Bordes et al., 2014a)	39.2	40.4
(Bordes et al., 2014b)	29.7	31.3
MCCNN (our)	40.8	45.1

An End-to-End Model for Question Answering over Knowledge Base with Cross-Attention Combining Global Knowledge Information (Hao et al. ACL, 2017)

□ 问题

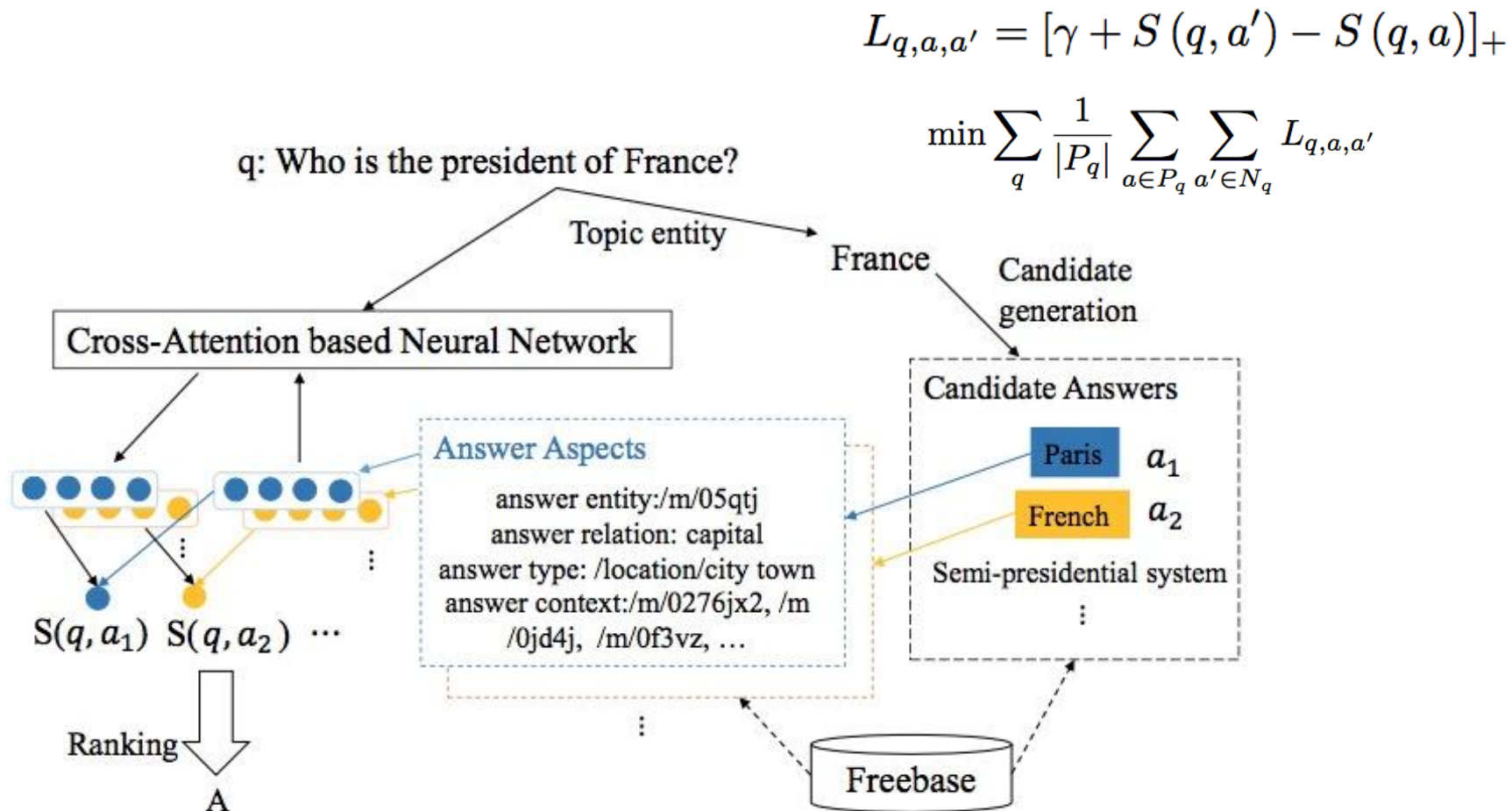
■ 问句表示

- 已有方法多采用Word Embedding的平均对于问句进行语义表示，问句表示过于简单
- 关注答案不同的部分，问句的表示应该是不一样的

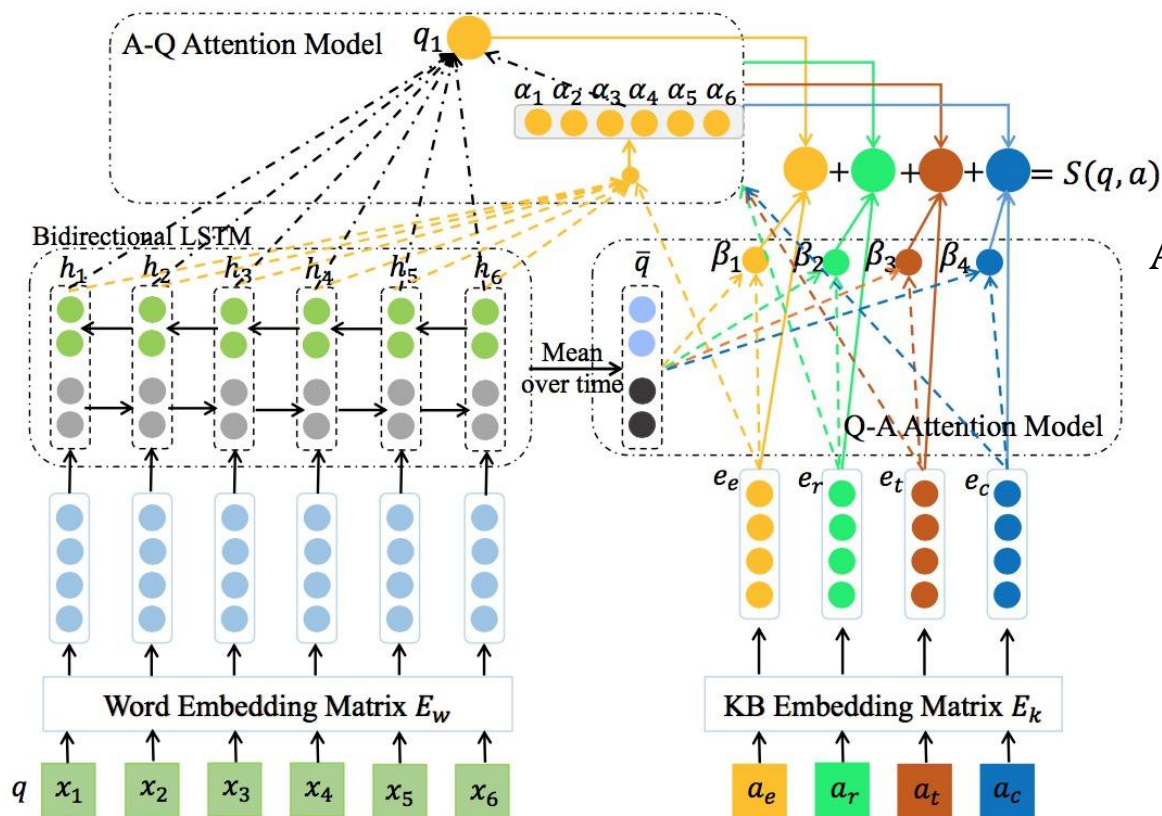
■ 知识表示

- 受限于训练语料
- 未考虑全局信息

(Hao et al. ACL, 2017)



(Hao et al. ACL, 2017)



$$\alpha_{ij} = \frac{\exp(\omega_{ij})}{\sum_{k=1}^n \exp(\omega_{ik})}$$

$$\omega_{ij} = f(W^T[h_j; e_i] + b)$$

A-Q Attention

$$q_i = \sum_{j=1}^n \alpha_{ij} h_j$$

$$S(q, e_i) = h(q_i, e_i)$$

$$S(q, a) = \sum_{e_i \in \{e_e, e_r, e_t, e_c\}} \beta_{e_i} S(q, e_i)$$

$$\beta_{e_i} = \frac{\exp(\omega_{e_i})}{\sum_{e_k \in \{e_e, e_r, e_t, e_c\}} \exp(\omega_{e_k})}$$

$$\omega_{e_i} = f(W^T[\bar{q}; e_i] + b)$$

Q-A Attention

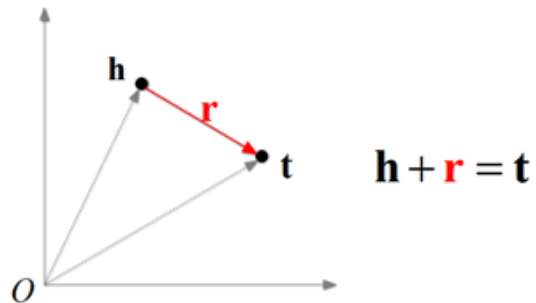
$$\bar{q} = \frac{1}{n} \sum_{j=1}^n h_j$$

(Hao et al. ACL, 2017)

- 融入全局信息
- 利用TransE得到knowledge embedding
- 多任务学习：

$$L_{q,a,a'} = [\gamma + S(q, a') - S(q, a)]_+$$

$$L_k = \sum_{(s,p,o) \in S} \sum_{(s',p,o') \in S'} [\gamma_k + d(s+p, o) - d(s'+p, o')]_+$$



中国+首都=北京

法国+首都=巴黎

俄罗斯+首都=莫斯科

TransE

(Hao et al. ACL, 2017) 实验结果

Methods	Avg F_1
Bordes et al., 2014b	29.7
Bordes et al., 2014a	39.2
Yang et al., 2014	41.3
Dong et al., 2015	40.8
Bordes et al., 2015	42.2
our approach	42.9

Methods	Avg F_1
LSTM	38.2
Bi-LSTM	39.1
Bi-LSTM+A-Q-ATT	41.6
Bi-LSTM+C-ATT	41.8
Bi-LSTM+GKI	40.4
Bi-LSTM+A-Q-ATT+GKI	42.6
Bi-LSTM+C-ATT+GKI	42.9

	where	is	the	carpathian	mountain	range	located
answer entity							
answer type							
answer relation							
answer context							

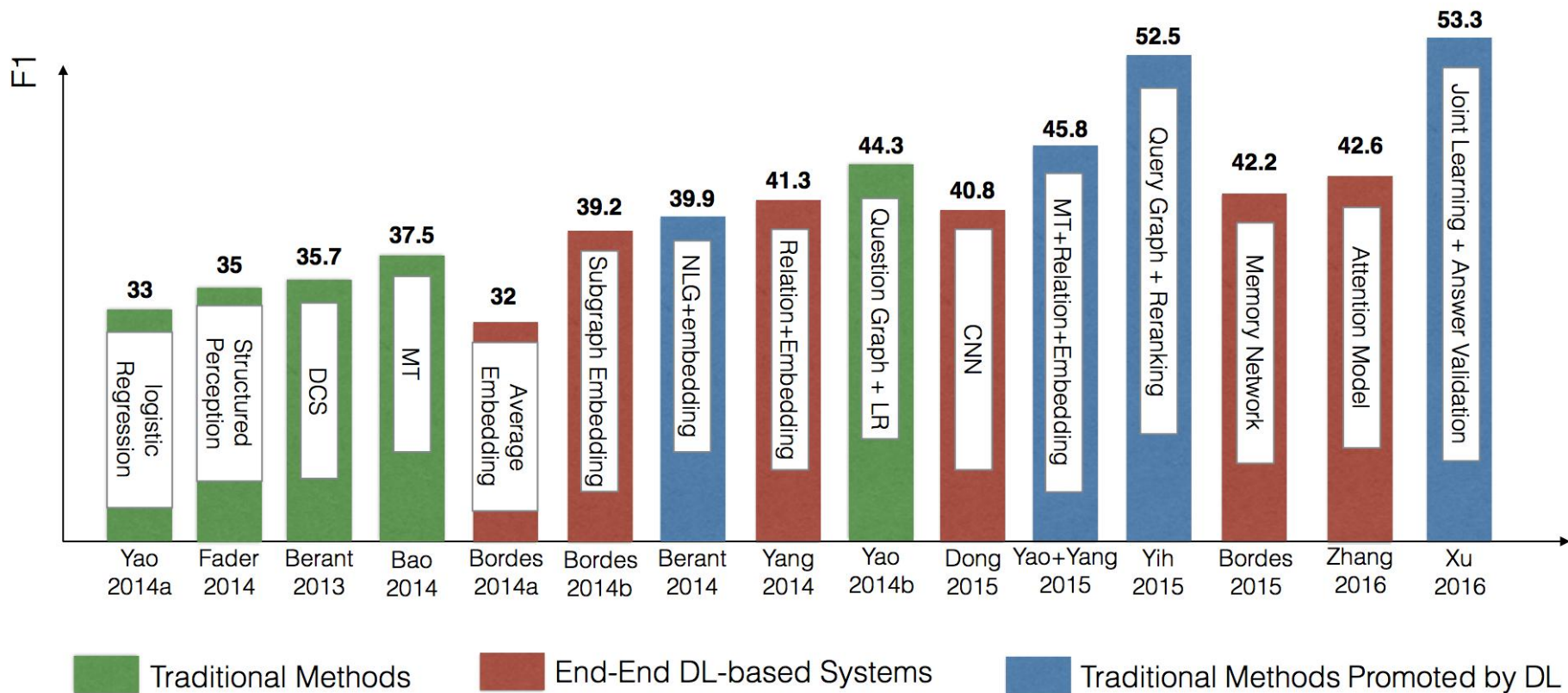
entity: Slovakia

type: /location/country

relation: partially/containedby

context: /m/04dq9kf, /m/01mp...

各种KBQA系统的效果比对



深度学习方法优缺点

□ 优点

- 无须像模板方法那样人工编写大量模板，也无须像语义分析方法中人工编写大量规则，整个过程都是自动进行

□ 缺点

- 目前也只能处理简单问题和单边关系问题，对于复杂问题不如两种传统方法效果好
- 由于DL方法通常不包含聚类操作，因此对于一些时序敏感性问题无法很好的处理，例如：“who is johnny cash’s first wife”，答案可能给出的是second wife的名字（模型只关注到了wife而忽略了first的含义，并没有进行额外的推理）。对于这种情况，可能需要定义专门（ad-hoc）的操作

三类方法总结对比

- ❑ **模板方法**：模板查询响应快、准确率高，可以回答相对复杂的问题。但人工定义模板费时费力，且经常无法与用户真实问题匹配。
- ❑ **语义方法**：可以回答较为复杂的问题，例如时序性问题。但人工编写规则工程量大。
- ❑ **深度学习**方法：无需人工编写规则定义模板，整个学习过程都是自动进行，但目前只能处理简单题和单边关系问题，且深度学习方法通常不包含聚类操作，因此时序性问题无法应对。

参考文献

- [Abujabal, et al. 2017] Abujabal, et al., Automated Template Generation for Question Answering over Knowledge Graphs, *www2017*
 - [Yao, et al. 2014] Yao, et al. A Graph Traversal Based Approach to Answer Non-Aggregation Questions Over DBpedia, *ACL2014*
 - [Bordes, et al., 2014] Antoine Bordes, Sumit Chopra, and Jason Weston, Question Answering with Subgraph Embedding, *EMNLP 2014*
 - [Dong, et al. 2015] Dong et al. Question Answering over Freebase with Multi-Column Convolutional Neural Networks. *ACL 2015*
 - [Hao, et al. 2017] Hao et al., An End-to-End Model for Question Answering over Knowledge Base with Cross-Attention Combining Global Knowledge Information, *ACL 2017*.
 - [Berant, 2013] Berant J, Chou A, Frostig R, et al. Semantic parsing on freebase from question-answer pairs. *Proceedings of Emnlp*, 2013.
 - [Yih,2015] Yih W T, Chang M W, He X, et al. Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base, *ACL 2015*.
 - [Christina, 2012] Christina et al., Template-based question answering over RDF data, *WWW 2012*
 - [Berant, 2014] Berant et al. Semantic parsing via paraphrasing. *ACL*, 2014.
-

谢谢大家！
