

# 基于知识图谱的自然语言问答



# 本章大纲

---

- KBQA(Question Answering over Knowledge Bases)概述
- 基于模板的方法
- 基于图模型的方法
- 基于神经网络的方法

# KBQA概述

---

# 背景

- 问答系统主要功能是回答人提出的**自然语言问题**



典型案例：IBM Watson

2011年，Watson战胜了其他人类竞争者，并获得答题比赛Jeopardy! 的一百万美金奖金。

# 背景

- 问答系统主要功能是回答人提出的**自然语言问题**



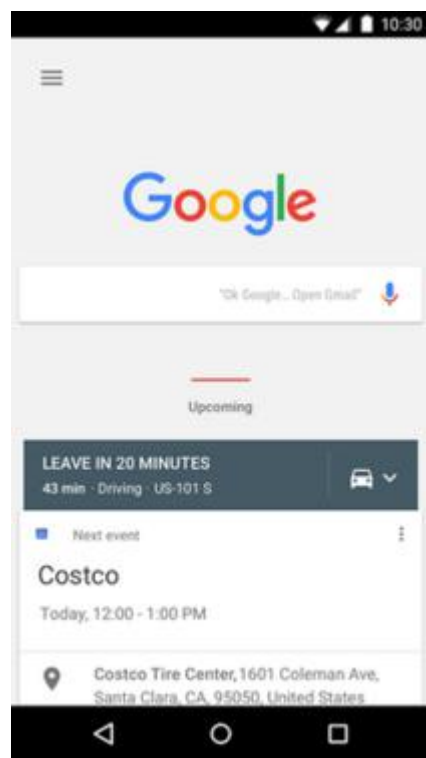
典型案例  
Apple Siri

Siri是WWDC 2016展上的明星产品。

# 背景：相关产品

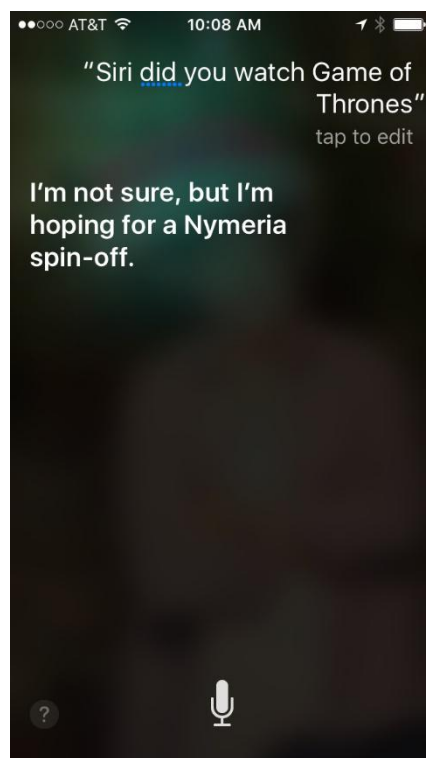
- 巨头公司纷纷加入问答系统大战。

Google  
Google Now



2021-4-26

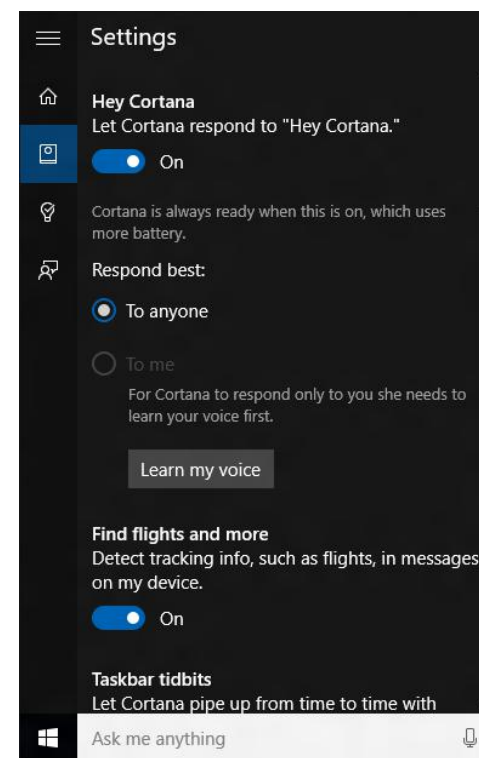
Apple  
Siri



Amazon  
Alexa



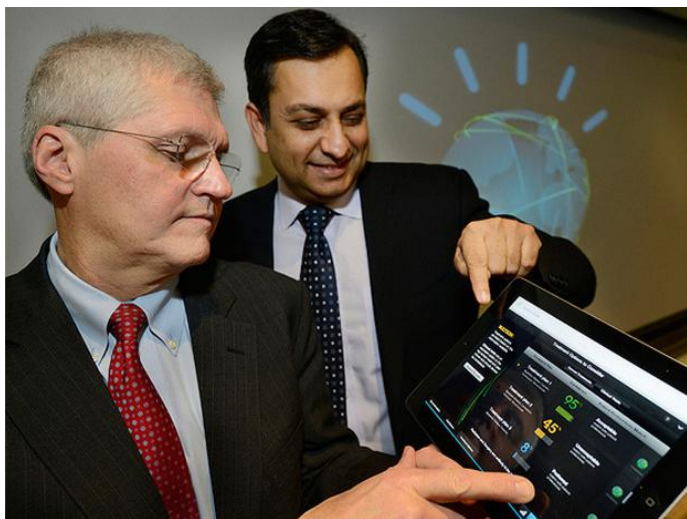
Microsoft  
Cortana



# 应用场景

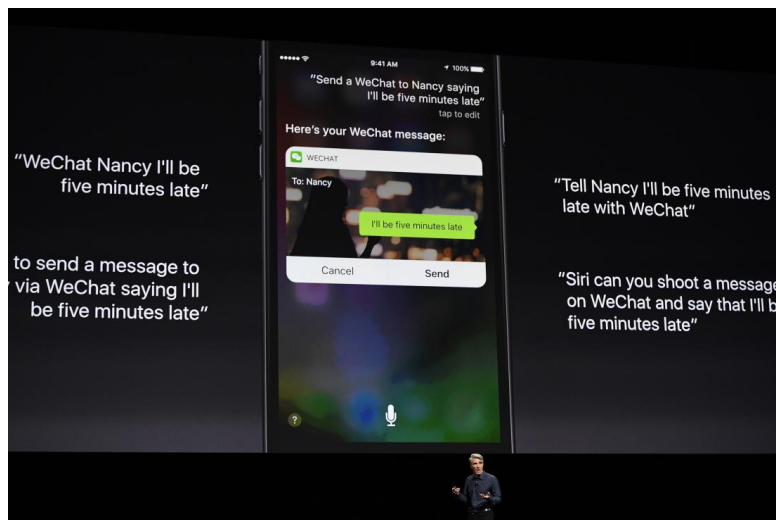
- 问答系统作为多个应用领域的知识入口。

## 健康咨询



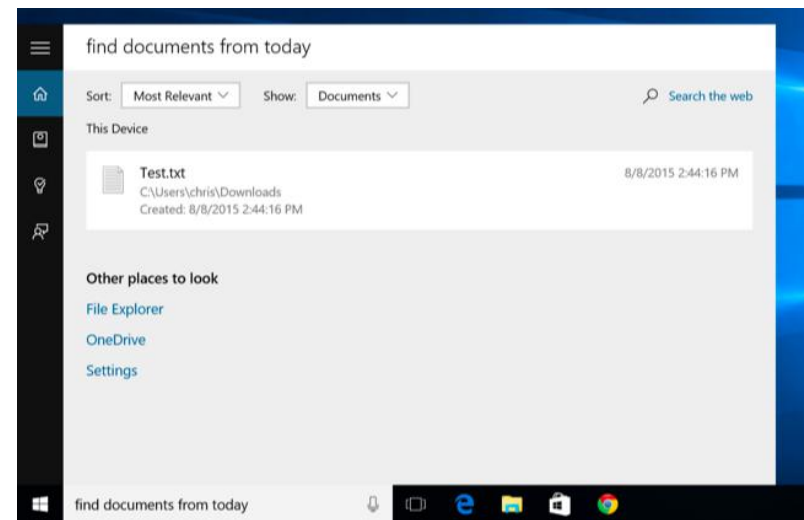
Bumrungrad Hospital and Watson Improve Cancer Care.

## 数字助理



Craig Federighi speaks during the Apple WWDC 2016.

## 自然语言搜索



Cortana supports natural language search for files on your computer.



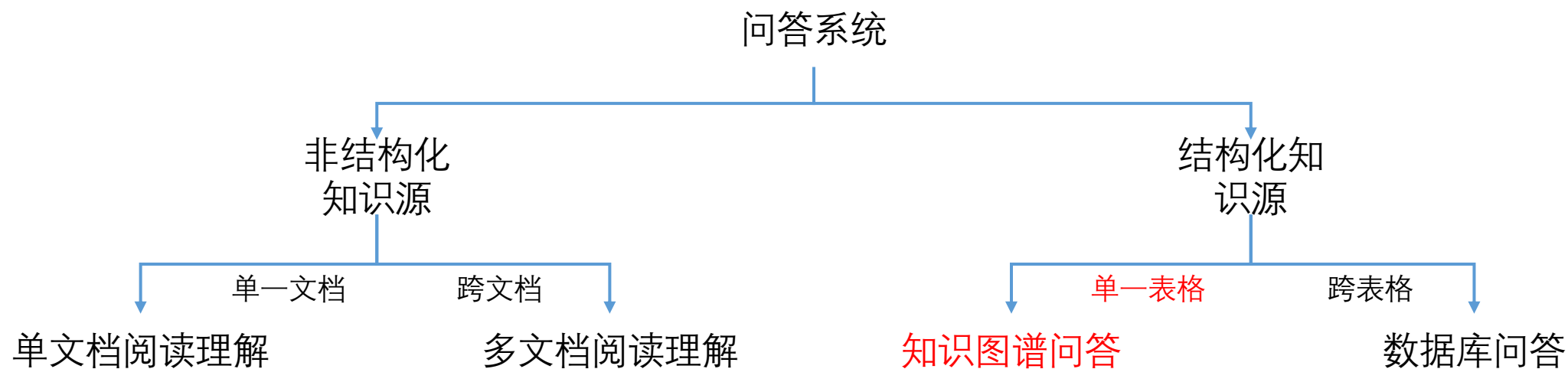
# 问答系统的研究与应用价值

---

- 问答正成为人机知识交互的主要形式之一
- 问答系统的研究是跨越人机语义鸿沟的重要尝试之一
- 问答系统是检验机器智能的重要方式之一



# 问答系统的分类



- 问答系统可以基于其知识源进行划分。
- 知识图谱是结构化知识源。

# 常见问题类型

---

- 事实型问题
  - When was Barack Obama born?
- 是非型问题
  - Is Beijing the capital of China?
- 对比型问题
  - Which city is larger, Shanghai or Beijing?
- 原因/结果/方法型问题
  - How to open the door?
- 观点型问题
  - What is Chinese opinion about Donald Trump?
- 对话型问题

# 常见问题类型

- 事实型问题

- When was Barack Obama born?

- 是非型问题

- Is Beijing the capital of China?

- 对比型问题

- Which city is larger, Shanghai or Beijing?

- 原因/结果/方法型问题

- How to open the door?

- 观点型问题

- What is Chinese opinion about Donald Trump?

- 对话型问题



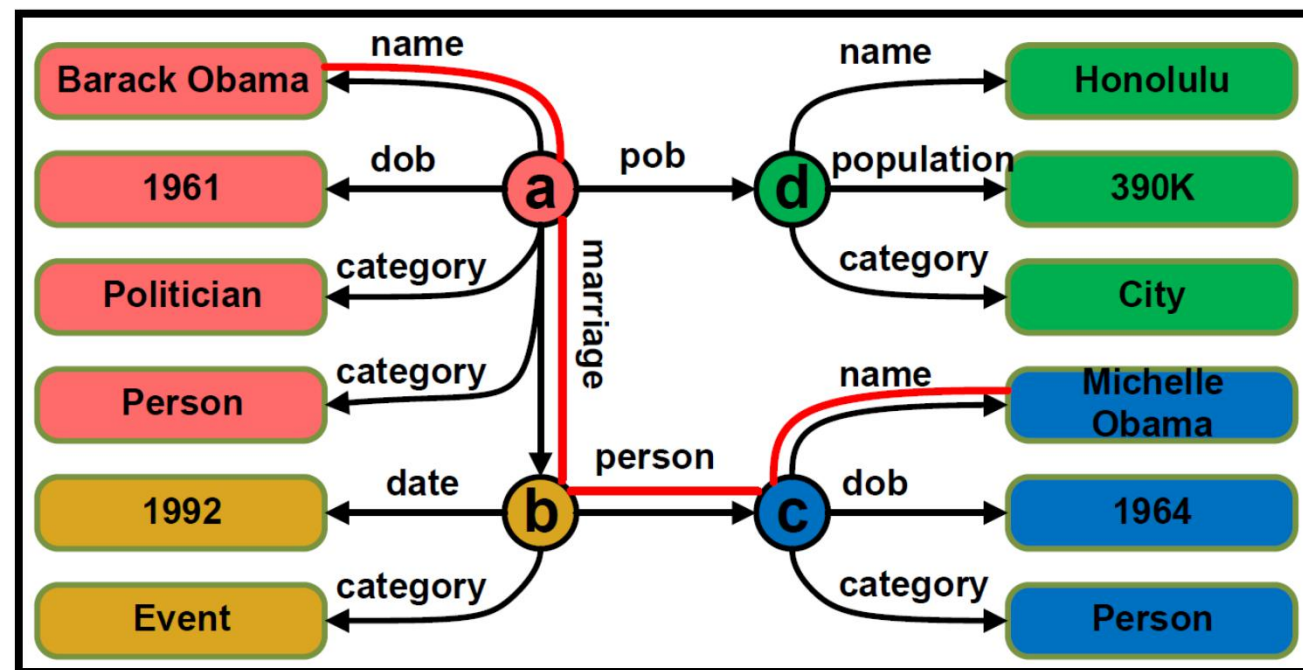
以事实类问题为核心

# 一个简单知识图谱示例

- 结构化、关联化数据表示
- 每个结点表示一个实体
- 每条边表示一条知识

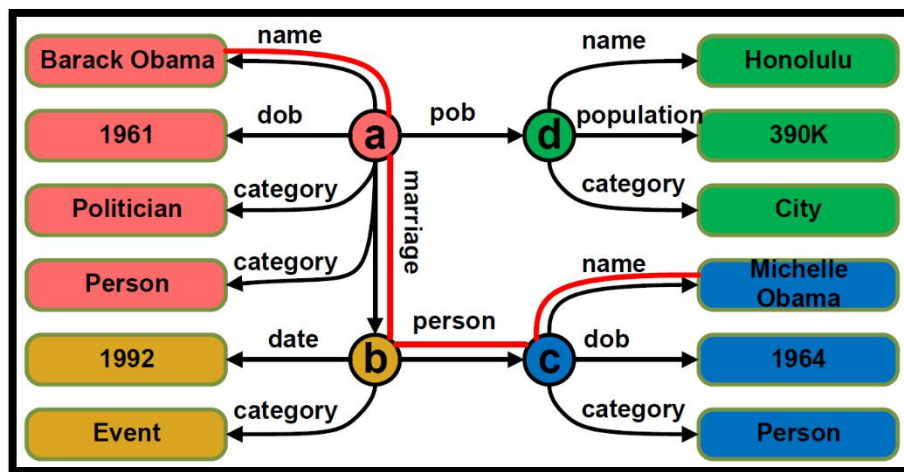
- 知识表示例子

- 檀香山的人口是39万
- 表示为
- $(d, \text{population}, 390k)$



# KBQA的优势（与文本对比）

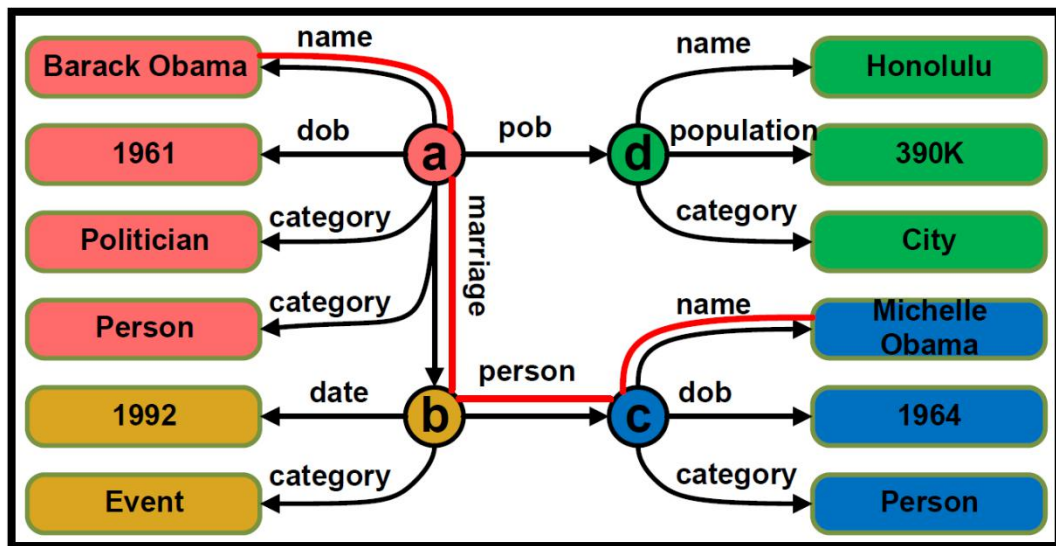
- 为问题的语义理解提供了丰富的背景知识
  - 纯文本: 字符、词法与语义理解
  - 知识图谱: 关联性数据, 提供文本背后的知识信息
- 提供了初步的推理能力
  - 基于知识图谱的关系推理, 问答系统可以回答知识图谱没有直接表达的事实。
    - When was Barack Obama's wife born?



# KBQA工作方式

- 将自然语言问题转化为知识图谱上的结构化查询（如SPARQL或SQL）
  - 查询语言由数据库决定，不是核心问题
- 核心问题：属性理解

How many people live in Honolulu?



SPARQL

```
Select ?number
Where {
  Res:Honolulu dbo:population ?num
}
```

SQL

```
Select value
From KB
Where subject='d' and
predicate='population'
```

# 属性理解

---

- 语义表示
  - 将问题表示为模板、图、向量等
- 属性关联
  - 由语义表示到知识图谱属性



# KBQA的测评

- 基于不同知识图谱、不同标注和不同语言的KBQA测评数据集

数据集	作者	数据类型	规模	URL
QALD 系列	ESWC workshop、ISWC workshop、CLEF 促进会的问答实验室	基于 DBpedia 的多语言问答任务、混合问答任务、基于 RDF 的大规模问答任务、以及基于维基百科的英语问答等多种任务。	50 至上百条样本	<a href="http://qald.aksw.org/">http://qald.aksw.org/</a>
WebQuestions	斯坦福大学	Google 用户搜索的问题，及其在 Freebase 上的答案	3778 个训练问答对样本和 2032 个测试问答对样本	<a href="https://github.com/brmson/dataset-factoid-webquestions">https://github.com/brmson/dataset-factoid-webquestions</a>
WebQSP	微软	对 WebQuestions 数据集加入了人工标注的语义解析	4737 个带有完整的语义解析查询语句的样本，1073 个带有部分标注的样本	<a href="https://www.microsoft.com/en-us/download/details.aspx?id=52763">https://www.microsoft.com/en-us/download/details.aspx?id=52763</a>
SimpleQuestion	Facebook	人工标注的问题及其在 Freebase 上对应的知识	108 442 个问答对样本	<a href="http://fb.ai/babi">http://fb.ai/babi</a>
NLPCC	中国计算机学会中文信息技术专业委员会	人工标注的中文问题，及其在中文百科知识图谱上的回答。作者同时放出了对应知识图谱的数据	24 479 个问答对样本	<a href="http://tcci.ccf.org.cn/conference/2017/taskdata.php">http://tcci.ccf.org.cn/conference/2017/taskdata.php</a>

# 基于模板的KBQA

---

# 模板方法分类

---

- 概念模板
  - `c1= *capital*$country*` （询问国家的首都）
  - `c2= *population*$location*` （询问地区的人口）
  - `c3= *length*$river*` （询问河流的长度）
  - `c4=“Which philosopher wrote $book?”` （询问哲学著作的作者）

# 模板方法分类

---

- 句法规则
  - 询问作者的问题
  - $S \rightarrow NP \parallel VP$
  - $NP \rightarrow Det \parallel N$
  - $VP \rightarrow V \parallel N$
  - $Det \rightarrow \text{which}|\text{who}$
  - $N \rightarrow \text{philosopher}|\text{scientist}$
  - $V \rightarrow \text{wrote}|\text{created}|\text{invented}$
  - $N \rightarrow \text{republic}|\text{Luceion}$
- 例:
  - Which philosopher wrote republic?
  - Which scientist created Luceion?

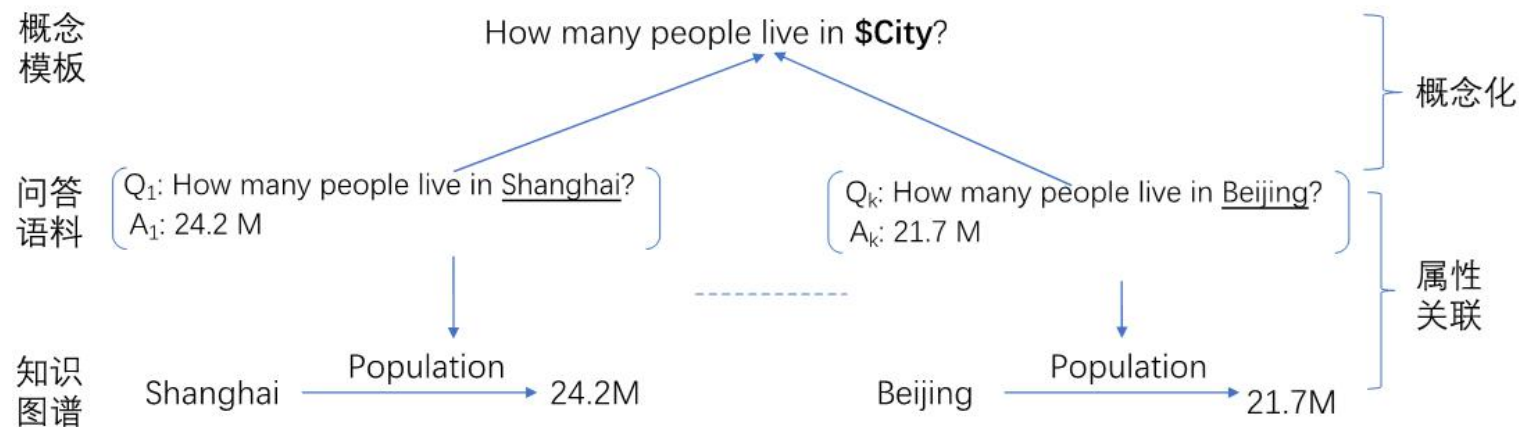
# 模板方法特点

---

- 可控性强、准确率高
  - 由专家人工构造
- 对问法多样性的覆盖差
- 成本高
  - 人工标注成本
- 适用于领域问答，或者高频问答

# 问题概念模板学习

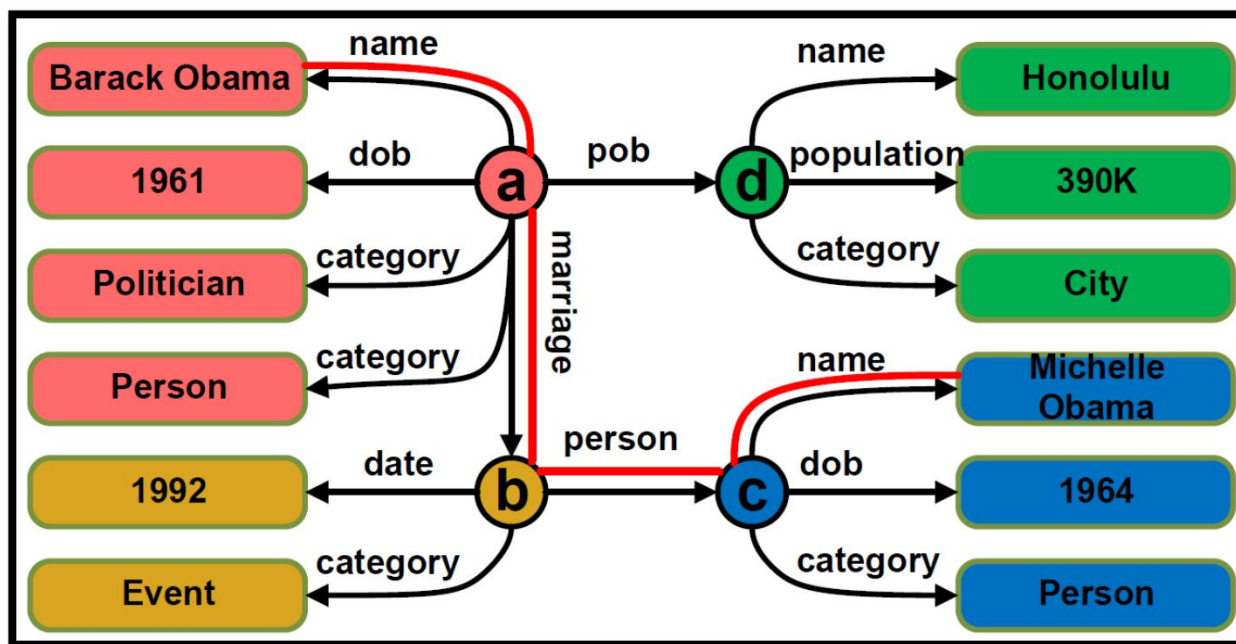
- 目标：突破人工构造模板的局限，降低成本，提升问法覆盖率
- 用模板（templates）表示自然语言问题
  - E.g.
    - How many people live in \$city?
  - 可解释性
  - 用户可控性



问题关键:收集大量模板并识别它们对应的属性

# Q2A: 生成过程

- A QA pair
  - Q: How many people live in Honolulu?
  - A: It's 390K.



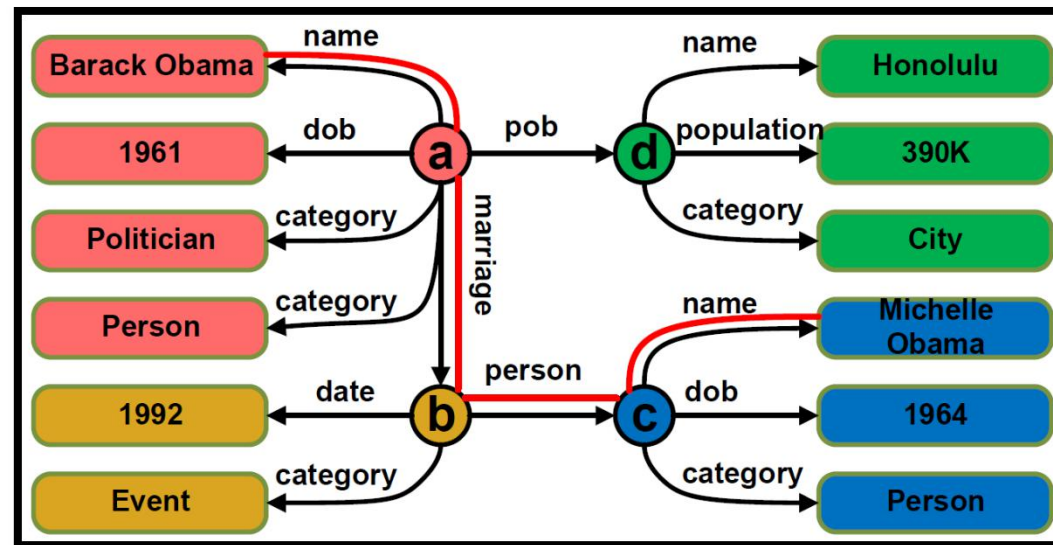


# Q2A: 实体链接 (entity linking)

How many people live in Honolulu?

d

$P(\text{entity} | \text{question})$



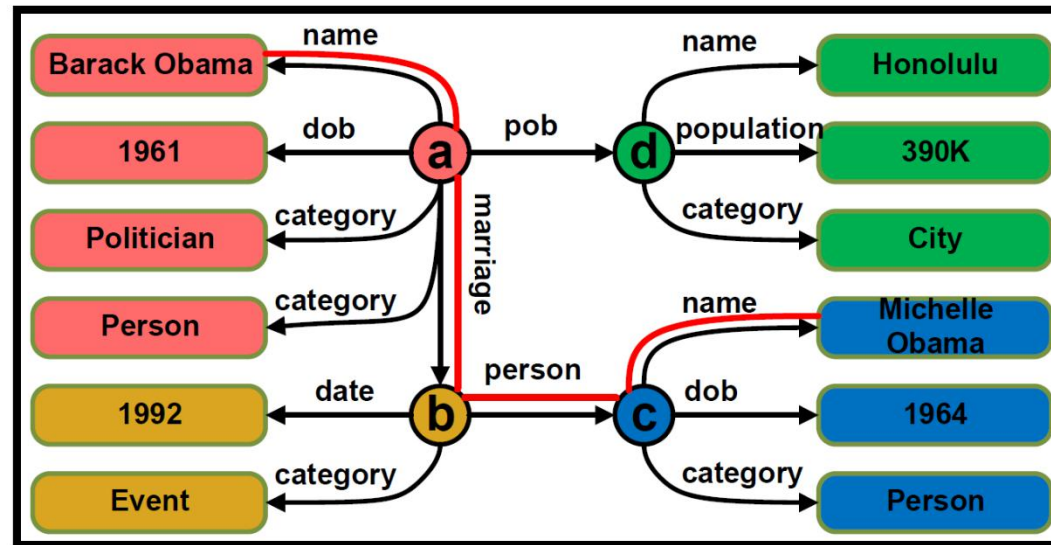
# Q2A: 概念化 (conceptualization)

How many people live in Honolulu?

d

How many people live in \$city?

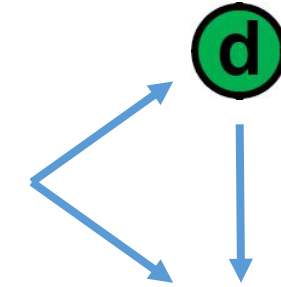
$$P(\text{type} \mid \text{question}, \text{entity}) \\ = P(\text{context} \mid \text{question}, \text{entity})$$



# Q2A: 属性推断 (predicate inference)

How many people live in Honolulu?

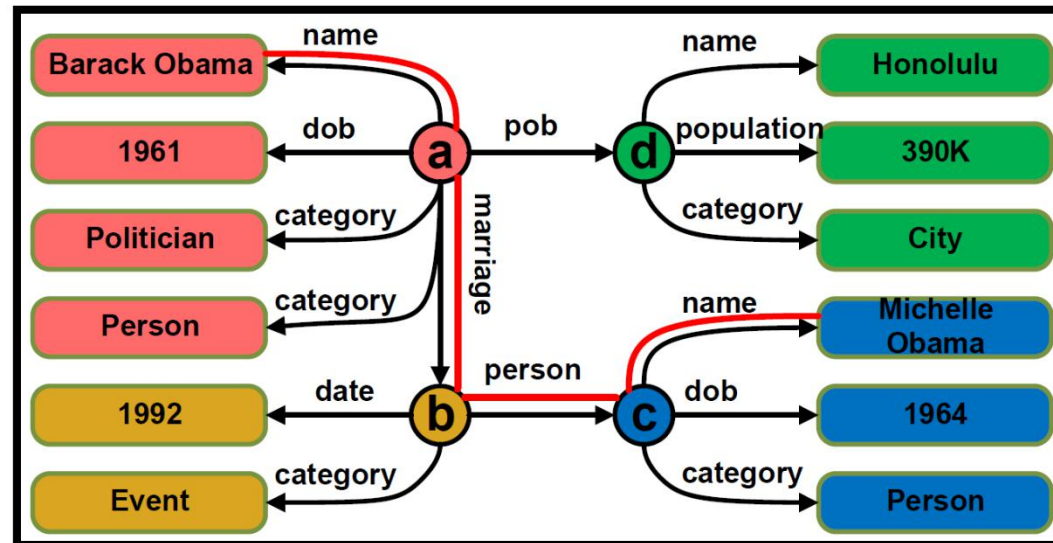
$P(\text{predicate} \mid \text{template})$



How many people live in \$city?



population



# Q2A: 值查找 (value lookup)

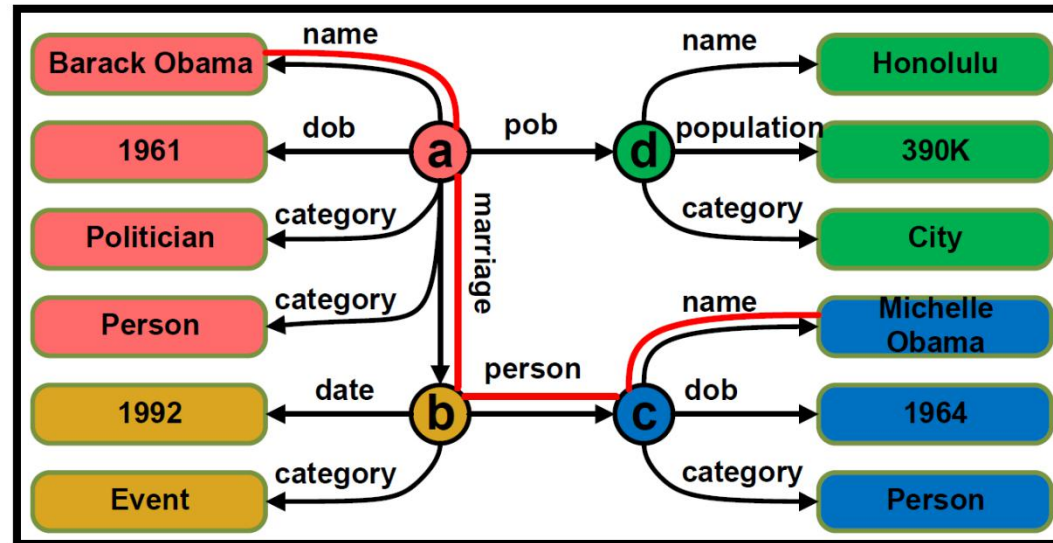
How many people live in Honolulu?

$P(\text{value} \mid \text{entity}, \text{predicate})$

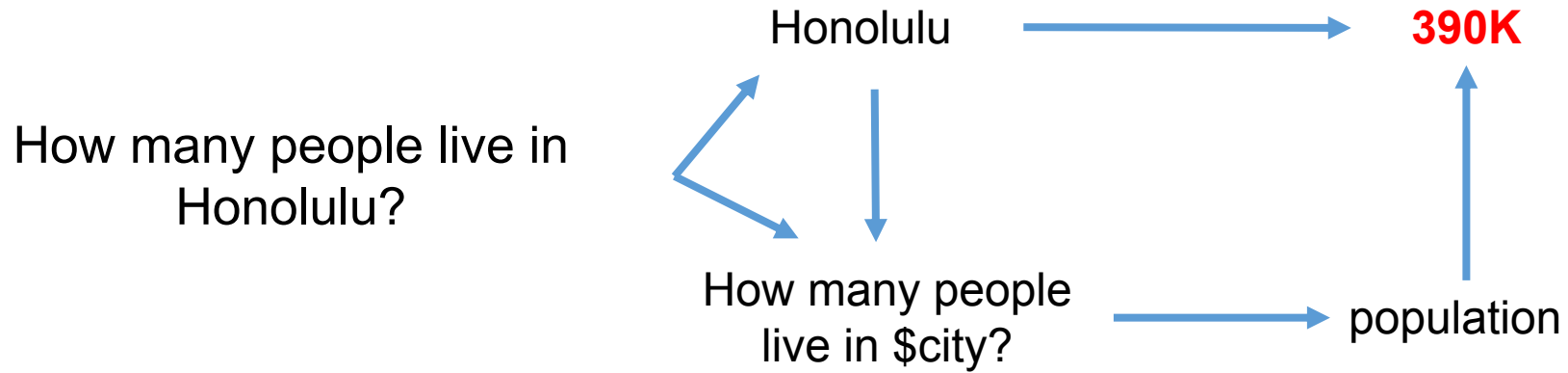
How many people  
live in \$city?

population

390K

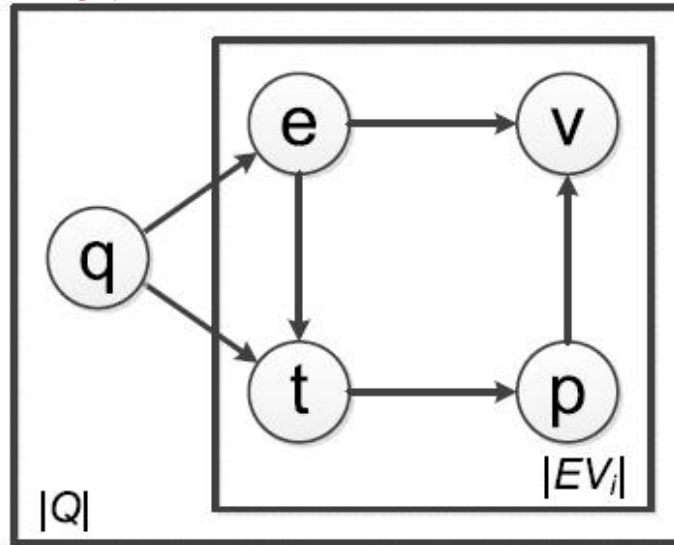


# 概率图模型 (Probabilistic graph model)



$$P(q, e, t, p, v)$$

$$= P(q)P(e|q)P(t|e, q)P(p|t)p(v|e, p)$$

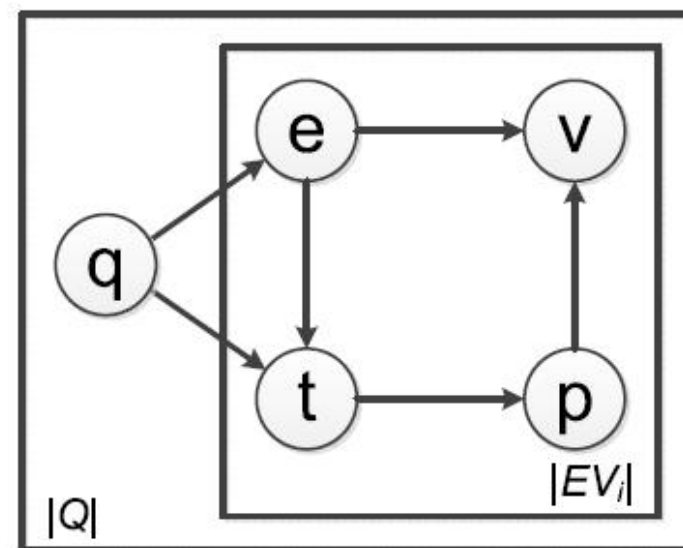


# 概率推断 (Probabilistic inferencing)

- 从QA语料库中学习参数 (42M Yahoo! Answers)

- 目标：最大化观测集如QA语料库的似然度

$$\begin{aligned} L_{\mathcal{QA}} &= \prod_{i=1}^n [P(q_i)^{1-|EV_i|} \prod_{(e,v) \in EV_i} P(e,v|q_i)P(q_i)] \\ &= \beta \prod_{i=1}^n [\prod_{(e,v) \in EV_i} P(e,v,q_i)] \end{aligned}$$



- 训练数据：4200万个问答对。远多于常用问答数据集 (WebQuestion, 5810对; QALD, 100对; SQUAD, 10万对)
- 学习结果：2700万个模板, 2782个意图

# 基于图模型的KBQA

---



# 基于子图匹配的问题理解

---

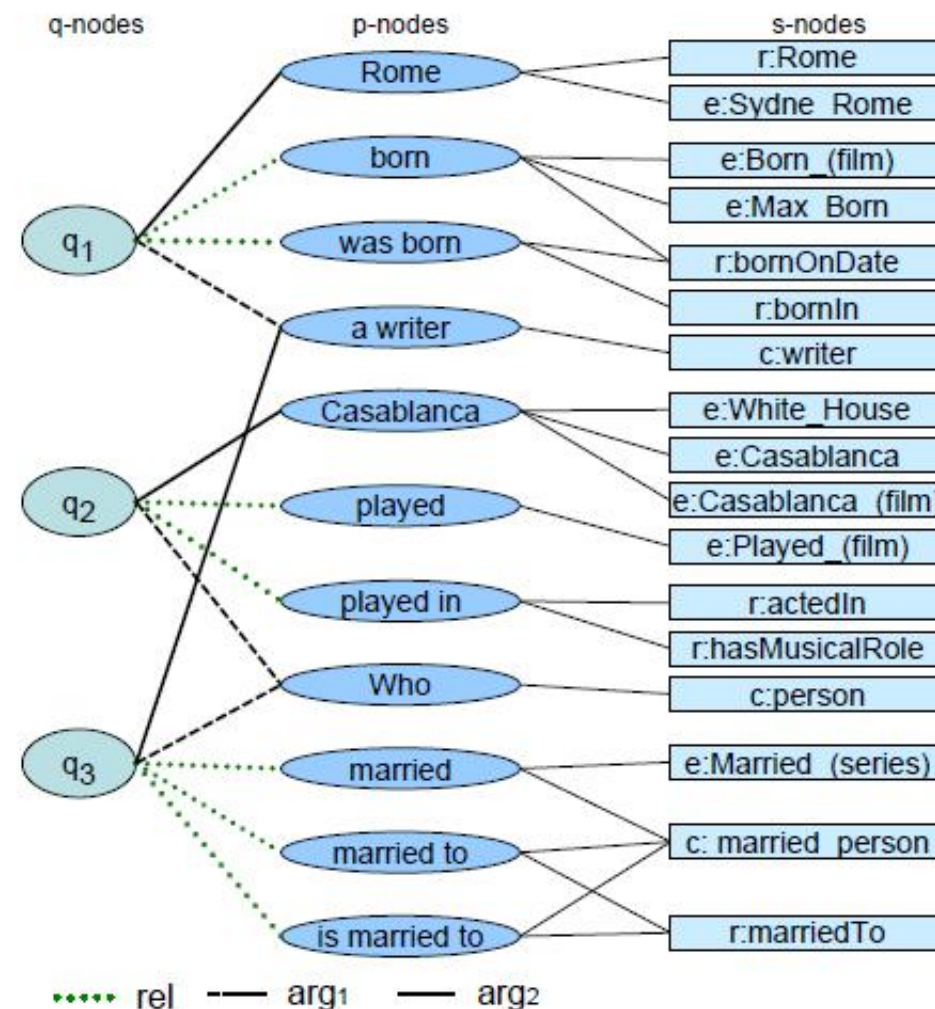
- 寻找可以回答问题 $q$ 的知识图谱子图 $d$

# 基于图算法的问题理解

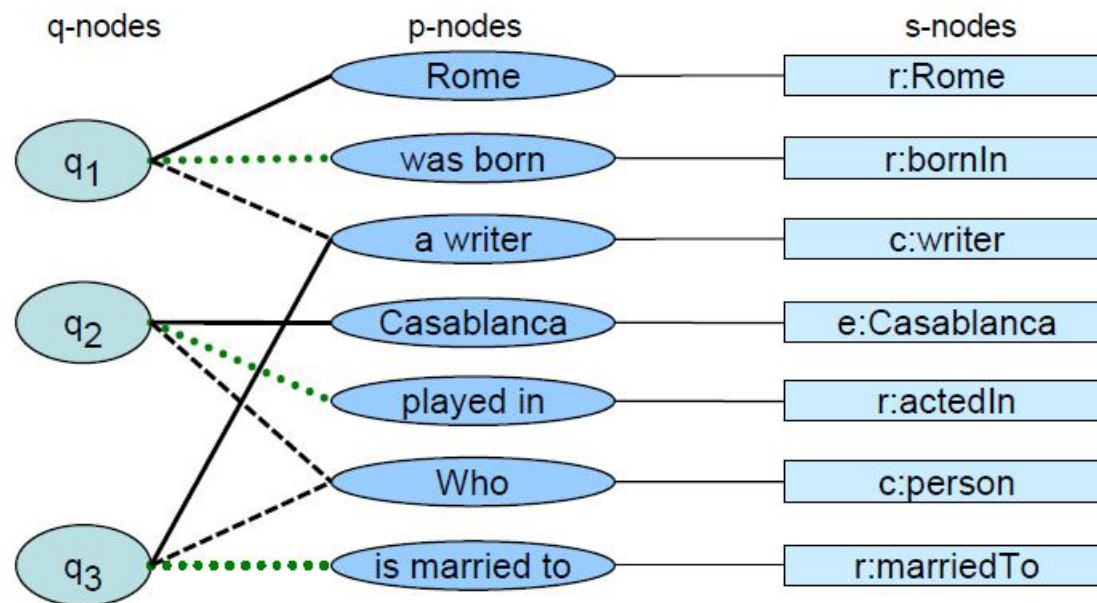
- 使用图表示问题语义与知识图谱关联
  - 问题的短语、候选实体、候选知识图谱属性表示为图中的结点
  - 问题短语、候选实体、候选知识图谱属性间关系表示为图中的边
  - 候选属性结点的权重，即为知识图谱属性与问题的语义相关度
- 将问题语义消岐，转化为图中候选点的筛选
- 使用图算法做语义消岐（候选点的筛选）
  - 通过子图匹配等图算法，选择高相似度子图所对应的语义

# 基于图算法的问题理解： DEANNA

- q-节点
  - 问题句法结构
- p-节点
  - 问题短语
- s-节点
  - 知识图谱的语义项 (semantic item, 包含实体及属性)
- p-s边权重
  - 短语与语义项的相似度
- s-s边权重
  - 两个语义项间的相似度
- q-p边
  - 关联问题句法结构与具体的SPO项



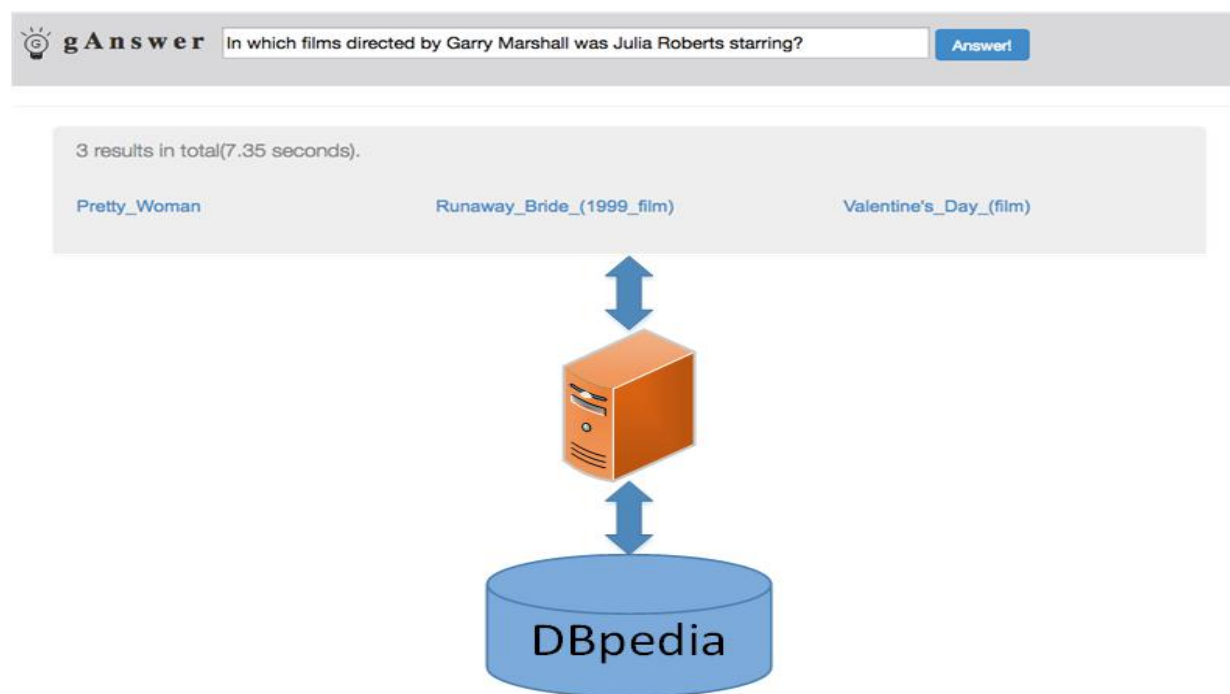
# 图表示方法：DEANNA



- 粘合度(导出子图权重)最高的子图即为消歧之后的句子语义

# gAnswer

- 提供方便的用户访问接口
- 数据库和自然语言处理的交叉研究，学术界和工业界共同关心的问题



# 示例

- Who was married to an actor that play in Philadelphia ?

Subject	Property	Object
Antonio_Banderas	type	actor
Antonio_Banderas	spouse	Melanie_Griffith
Antonio_Banderas	starring	Philadelphia_(film)
Philadelphia_(film)	type	film
Jonathan_Demme	director	film
Philadelphia	type	city
Aaron_McKie	bornIn	Philadelphia
James_Anderson	playForTeam	Philadelphia_76ers
Constantin_Stanislavski	create	An_Actor_Pre pares
Philadelphia_76ers	type	Basketball_team
An_Actor_Pre pares	type	Book

# 示例

- Who was married to an actor that play in Philadelphia ?



```
Select ?y where {  
  ?x starring Philadelphia_(film) .  
  ?x type actor .  
  ?x spouse ?y .  
}
```

Philadelphia

Philadelphia\_(film)

Philadelphia\_76ers



# 示例

- Who was married to an actor that play in Philadelphia ?



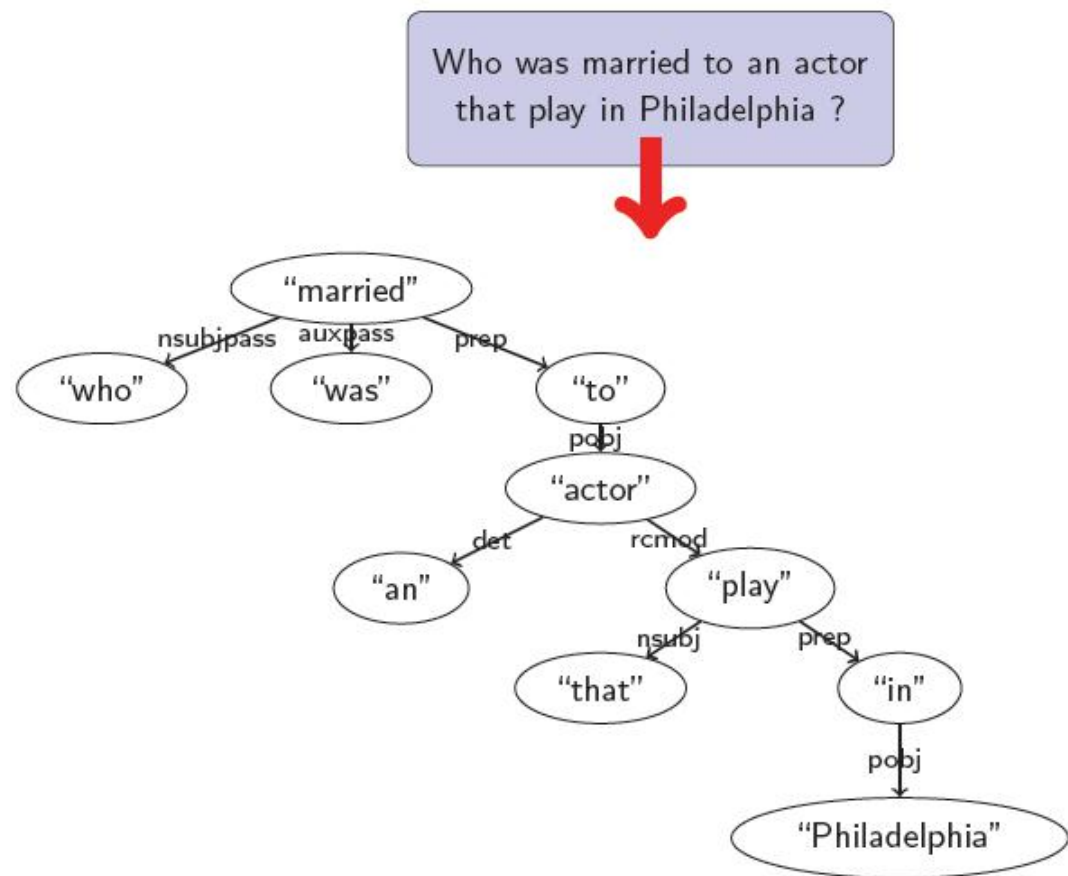
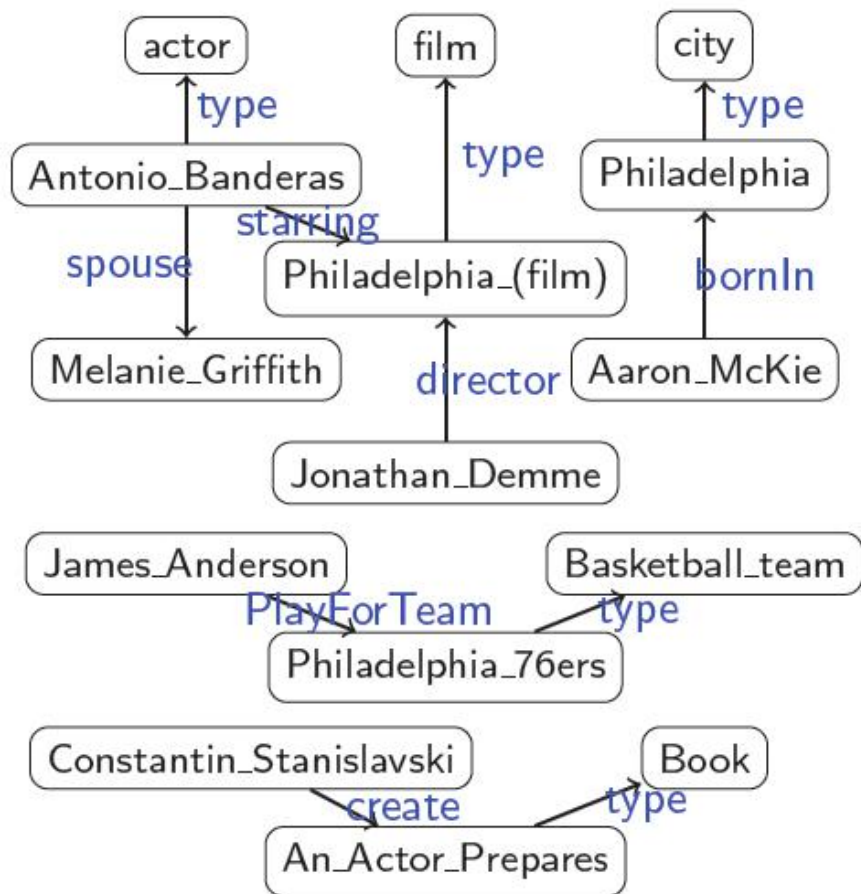
```
Select ?y where {  
  ?x starring Philadelphia_(film) .  
  ?x type actor .  
  ?x spouse ?y .  
}
```

playForTeam

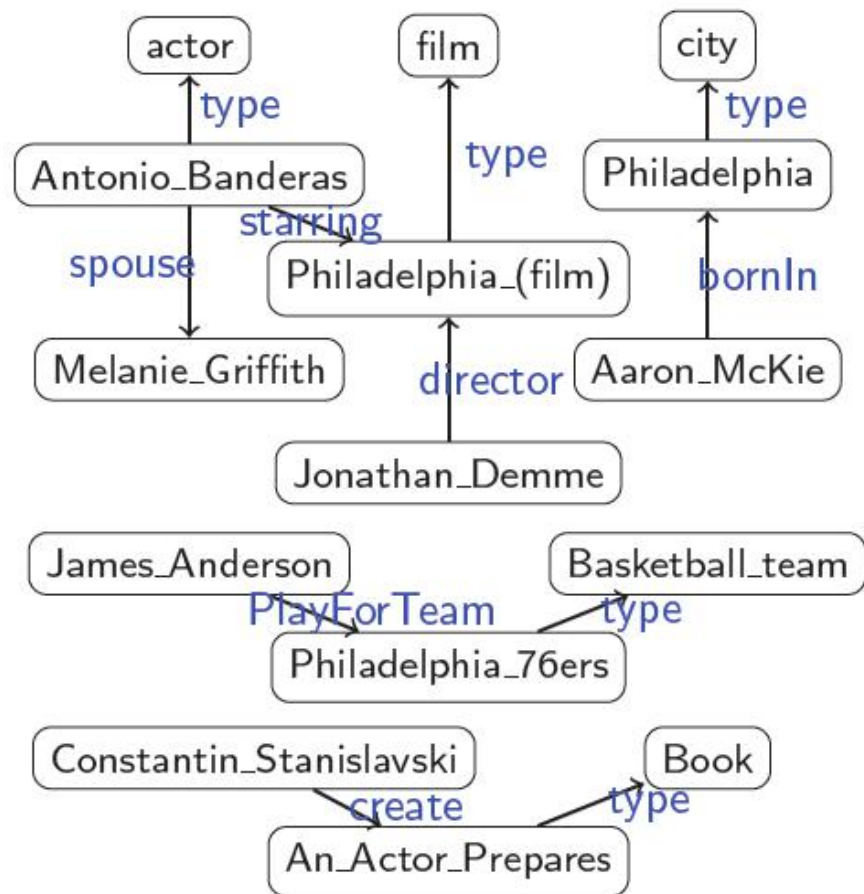
starring

director

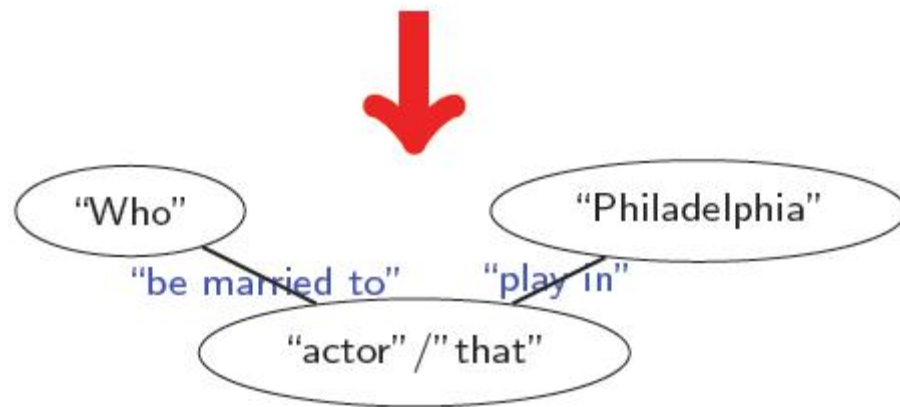
# gAnswer: 面向数据的问答方法



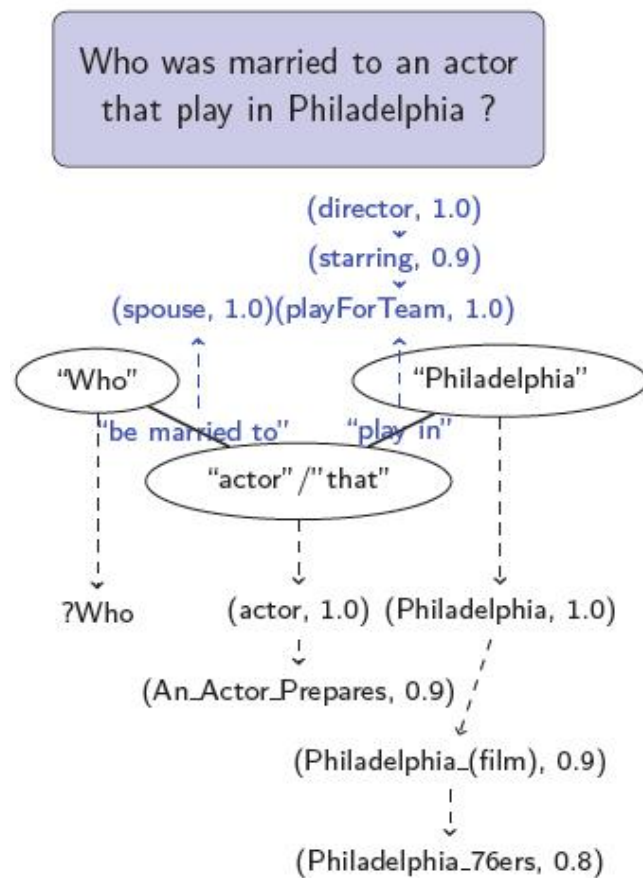
# gAnswer: 面向数据的问答方法



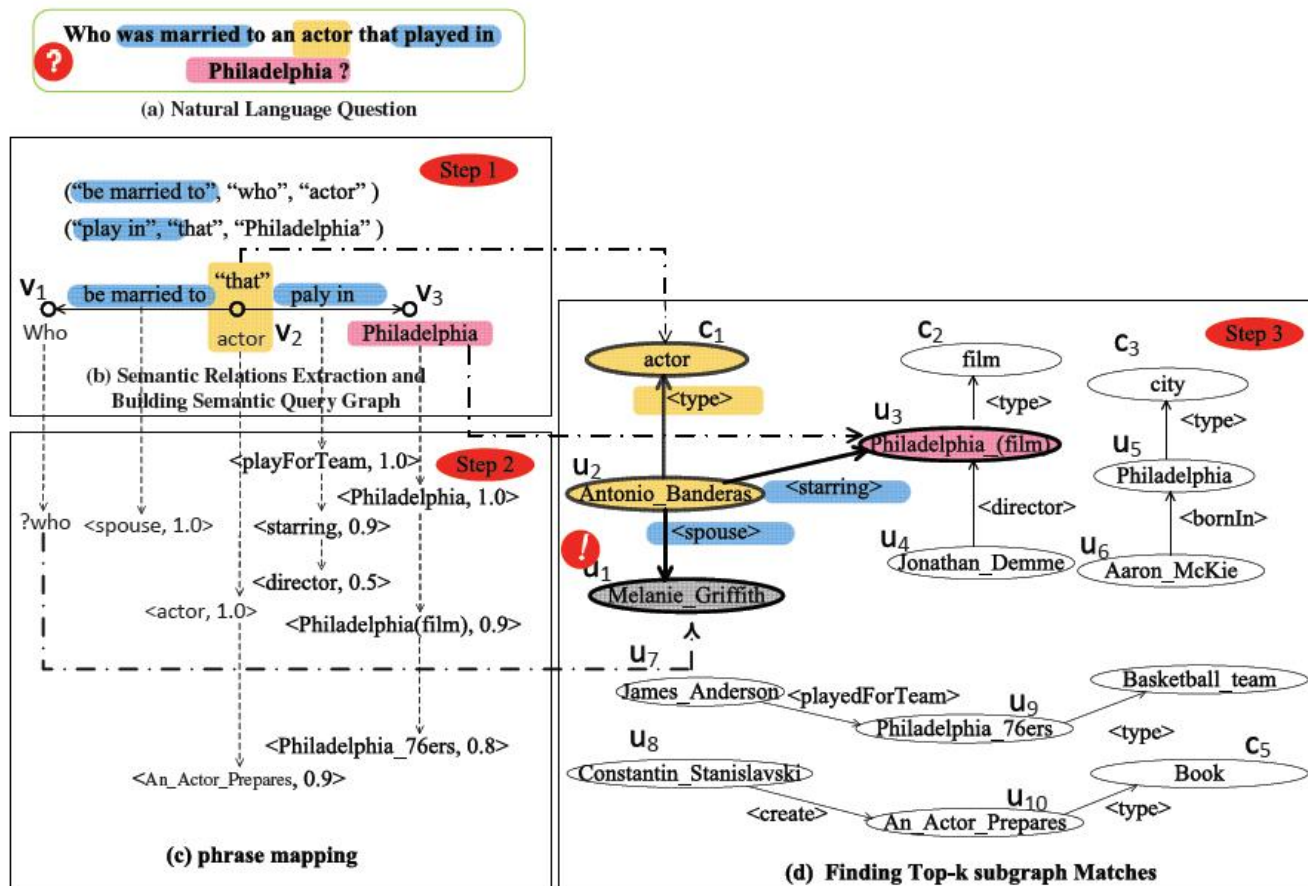
Who was married to an actor  
that play in Philadelphia ?



# gAnswer: 面向数据的问答方法

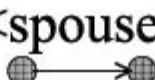
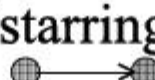
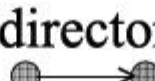



# 方法框架



# 核心贡献

- 构建一个从关系到知识图谱的映射

Relation Phrases	Predicates or Predicate Paths	Confidence Probability
“be married to”	 <spouse>	1.0
“play in”	 <starring>	0.9
“play in”	 <director>	0.5
“uncle of”	 <hasChild> <hasChild> <hasChild>	0.8
... ..	... ..	... ..

# 基于Patty的关系解析映射

- Patty是德国马克斯普朗克研究所做的一个语义关系对

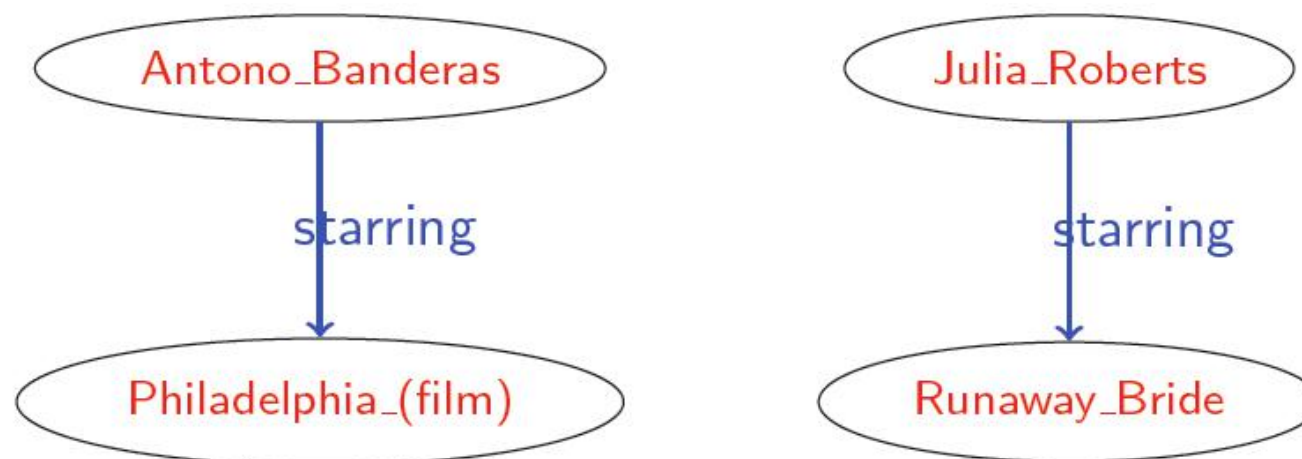
Relation Phrase	Supporting Entity Pairs
"play in"	(⟨ Antonio_Banderas ⟩, ⟨ Philadelphia(film) ⟩), (⟨ Julia_Roberts ⟩, ⟨ Runaway_Bride ⟩),.....
"uncle of"	(⟨ Ted_Kennedy ⟩, ⟨ John_F._Kennedy,_Jr. ⟩) (⟨ Peter_Corr ⟩, ⟨ Jim_Corr ⟩),.....



# 基于Patty的关系解析映射

- Patty是德国马克斯普朗克研究所做的一个语义关系对

Relation Phrase	Supporting Entity Pairs
"play in"	(⟨ Antonio_Banderas ⟩, ⟨ Philadelphia(film) ⟩), (⟨ Julia_Roberts ⟩, ⟨ Runaway_Bride ⟩),.....
"uncle of"	(⟨ Ted_Kennedy ⟩, ⟨ John_F._Kennedy,Jr. ⟩) (⟨ Peter_Corr ⟩, ⟨ Jim_Corr ⟩),.....

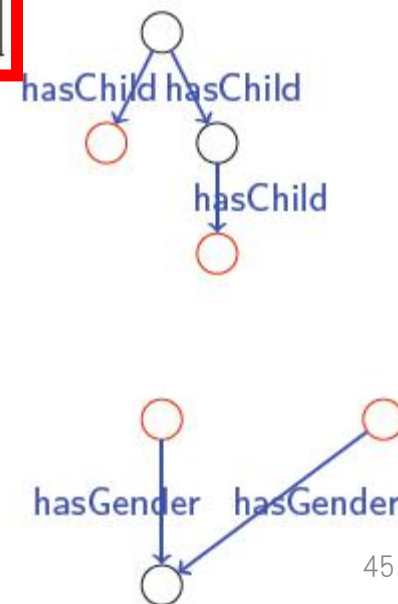
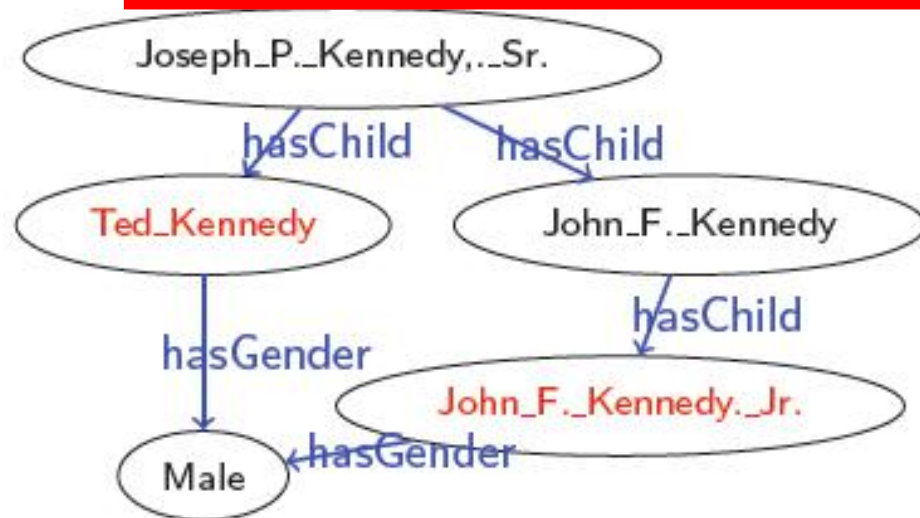





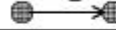

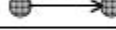
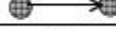


# 基于Patty的关系解析映射

- Patty是德国马克斯普朗克研究所做的一个语义关系对

Relation Phrase	Supporting Entity Pairs
"play in"	(⟨ Antonio_Banderas ⟩, ⟨ Philadelphia(film) ⟩), (⟨ Julia_Roberts ⟩, ⟨ Runaway_Bride ⟩)
"uncle of"	(⟨ Ted_Kennedy ⟩, ⟨ John_F._Kennedy,_Jr. ⟩) (⟨ Peter_Corr ⟩, ⟨ Jim_Corr ⟩),.....



# 基于Patty的关系解析映射

Relation Phrases	Predicates \\Predicate Paths	Confidence Probability
“was married to”	<spouse> 	1.00
“was born in”	<birthplace> 	1.00
“mother of”	<parent> 	0.95
“are located in”	<locatedInArea> 	0.98
“is fed by”	<inflow> 	1.00
“open in”	<locationCity> 	1.00
“is coauthor of”		1.00

# 基于图模型的方法小结

---

- 使用图表示问题语义
- 图算法做语义消岐
- 效果：
  - 召回上升（单词级别语义匹配，以单词与图结点的匹配表示问题）
  - 准确率受限（图的结点和边是离散的，无法有效表示文本语义）
- 可解释性：
  - 来自图的合理性
  - 图上算法可解释性一般

# 基于深度学习的KBQA

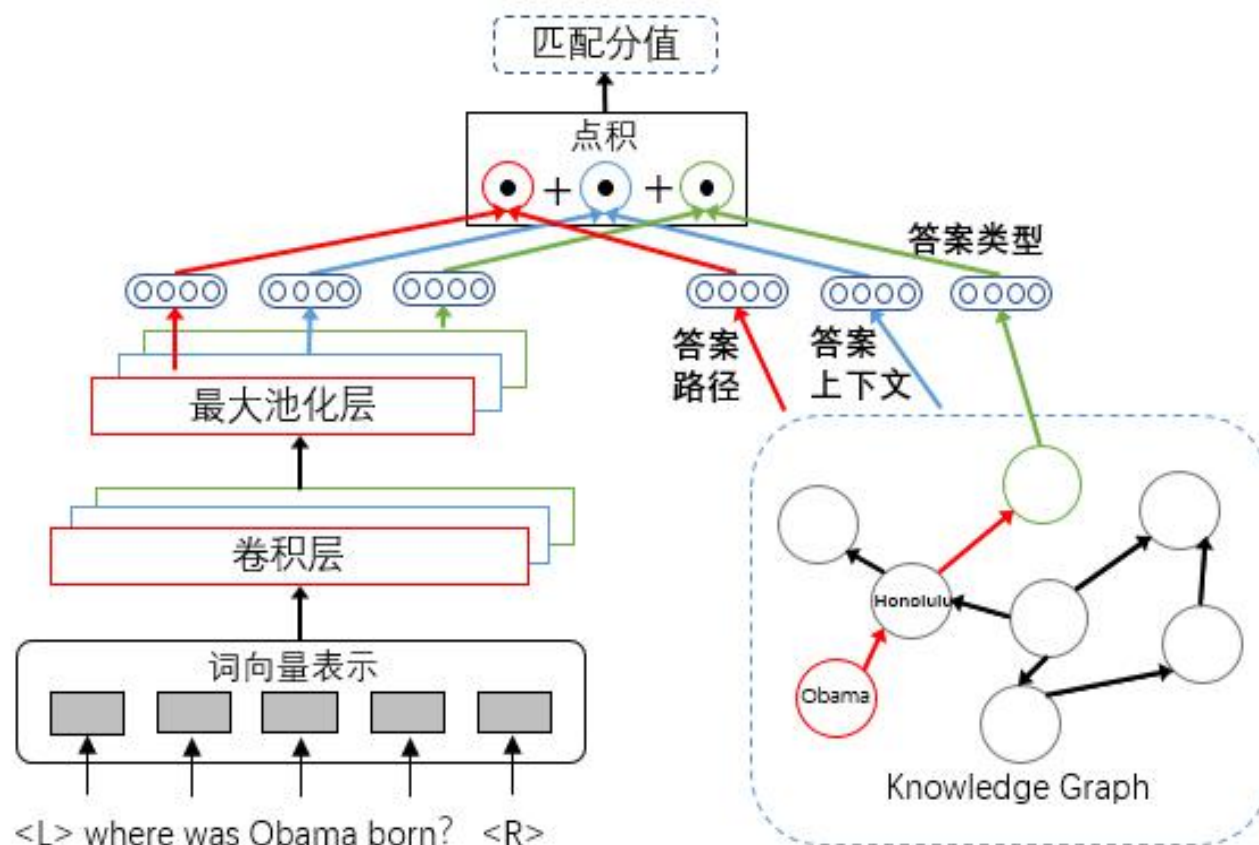
---

# 深度学习分类模型

- 将问题理解转化为知识图谱属性识别的多分类问题（multi-class）。
- 输入：问题
- 输出：意图/属性分布（predicate distribution）
  - 多少人居住在美国？
    - 地点： 0.04
    - 人口： 0.85
    - 总统： 0.02
    - ...

# 深度学习分类模型

- 问题表示
  - 词向量表示
  - 卷积层
    - 序列信息表示
  - 池化层
    - 句子信息融合
- 知识图谱表示
  - 答案路径嵌入表示
  - 答案上下文嵌入表示
  - 答案类型嵌入表示
- 匹配
  - 问题和知识图谱表示的相似度计算（点积）



# 基于深度学习的问题表示

---

- 深度学习使用表示学习方法
  - 将离散的问题表示为连续向量
  - 使用端到端的方法，建模学习问题表示及属性关联
- 避免了人工设计的特征

# 基于深度学习的答案表示

- 答案特征
  - 答案路径
  - 答案上下文
  - 答案类型
  - ...
- 答案路径的嵌入表示
  - 将离散的答案路径转化为连续低维向量

$$f(p) = W_p u_p$$

- $u_p$ : 答案路径中属性的二元向量表示
  - $W_p$ : 嵌入矩阵
- 其它简单知识图谱特征的表示类似



# 答案上下文表示

- 一般表示为该复杂图结构的所有实体（点）和关系（边）的表示的和

$$\begin{aligned} & Ebn (< \text{柏拉图}, \text{代表作品} - i \mathbb{A}, \text{书籍} >) \\ &= Ebn (\text{柏拉图}) + Ebn (\text{代表作品}) + Ebn (i \mathbb{A}) + Ebn (\text{书籍}) \end{aligned}$$

- 图神经网络算法

# 深度学习生成模型

---

- 问题涉及的候选属性较少时，适用分类模型
- 问题涉及多个属性或条件时，分类模型的结果空间会呈现指数级增长，分类模型不再适用
- 生成模型
  - 将自然语言问题到知识图谱结构化查询的映射问题转化为结构化查询语句的生成问题

# 生成模型方法—与分类模型对比

- 分类模型

- 多少人住在美国?

- 地点: 0.04

- 人口: 0.85

- 总统: 0.02

- ...

- 分类模型适用于问题涉及的候选属性较少时

- 生成模型 (generative model)

- 纽约大学有多少个CFL团队?

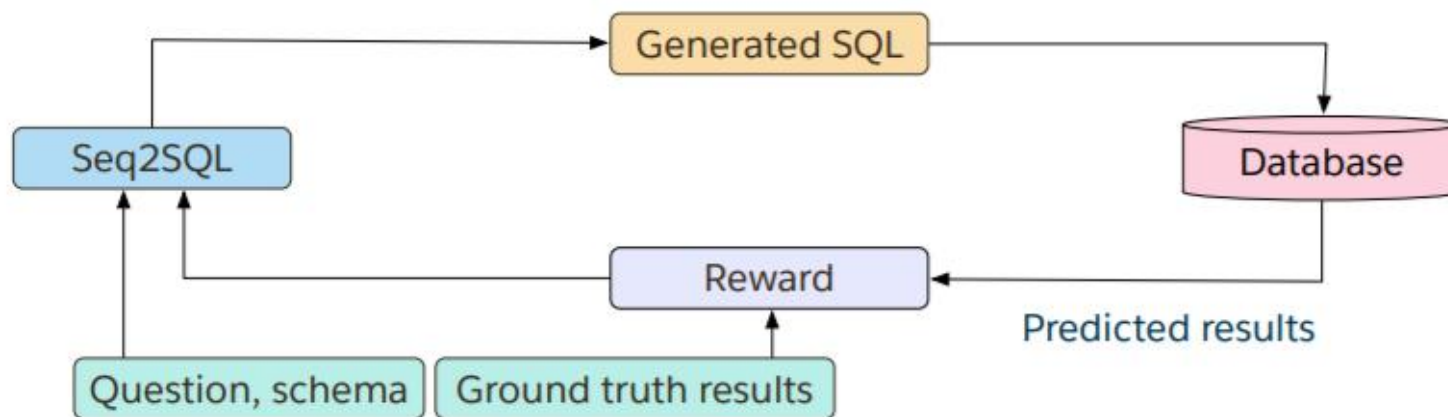
- `[SELECT] [COUNT CFL Team]`  
`[FROM CFLDraft] [WHERE`  
`College='York' ]`

- SEQ2SEQ → SEQ2SQL

- 问题涉及多个属性或条件时, 分类模型的结果空间会呈现指数级增长

- 此时适用生成模型, 分类模型不再适用

# SEQ2SQL：基于强化学习的生成模型



- Seq2SQL
  - 输入：问题和表格的列
  - 输出：生成相应的SQL
- 目标函数（Reward）：SQL对数据库查询的结果和标准答案的匹配度
- 结构化查询有效性：
  - Sql需要符合固定语法规则
  - 使用RL做自监督

# 深度学习方法

---

- 使用端到端的学习知识图谱表示和问题理解
- 效果：较高的准确率和召回率
  - 基于对问题和知识图谱的向量化理解
- 可解释性：差，人类无法解释和预测神经网络
- 趋势性方法

# 未来的工作

---

- 如何构建一个更好的KBQA系统?
  - 精确率/召回率 (Precision/recall)
    - 更好的模型 (自监督)
    - 更多数据(文本/图像)
    - 常识理解
  - 可控性
    - 神经网络的可控性和可解释性
  - 可迁移性
    - 将一个领域的KBQA模型迁移到新的领域
  - 鲁棒性
    - 如何理解对抗问题样本