

# Question Answering Systems

## Keeping efficiency in mind

**Rishiraj Saha Roy**

Max Planck Institute for Informatics, Germany

5

# Question of the day

How can we design efficient QA systems?

# You'll find this covered in

- ①
  - More accurate question answering on Freebase
    - Bast and Haussmann
    - CIKM 2015
    - <http://ad-publications.informatik.uni-freiburg.de/freebase-qa.pdf>
- ②
  - Towards a Question Answering System over the Semantic Web
    - Semantic Web Journal 2018
    - <https://arxiv.org/pdf/1803.00832.pdf>

*Diefenbach et al.*


# Research paper 1

## More accurate question answering on Freebase

### More accurate question answering on freebase

[H Bast](#), [E Haussmann](#) - Proceedings of the 24th ACM International on ..., 2015 - dl.acm.org

Real-world factoid or list questions often have a simple structure, yet are hard to match to facts in a given knowledge base due to high representational and linguistic variability. For example, to answer "who is the ceo of apple" on Freebase requires a match to an abstract" ...

☆  Cited by 133 Related articles All 6 versions

# The Aqqu System

- 1 ■ Template-based system *only gives templates* *More Accurate* → Aggu
- 2 ■ <sup>a</sup> Efficiency and <sup>b</sup> accuracy twin objectives
  - Thoroughly engineered → interactive response times 1 min X  
~10-20 secs/per D ✓  
→ efficient also research problem ~20 ms ✓
- 3 ■ Guides reader through possibilities → well-written
- 4 ■ No reliance on NERD, end-to-end → entity id as part of QA + handholding (LTR, metrics)
- 5 ■ Reproducibility of primary importance
- 6 ■ *Model paper* Related work tells a story
  - "extensibility" { demo (dom) result files KG subset/dump } + dependencies + code

# System overview

- 1 ■ Entity identification
- 2 ■ Template matching (Query candidate generation)
- 3 ■ Predicate Relation matching
- 4 ■ Answer type matching → pruning
- 5 ■ Candidate feature construction
- 6 ■ Query ranking
- 7 ■ Query Candidate pruning → answer

what char  
 does ellen play in finding nemo?  
 E Ellen1 Ellen2 R verb? FN1 2 2  
 (ent) mention, relations: surface for NL/question

entity, predicate: logical/K6 form

Here: relation  $\equiv$  predicate

① Ellen parent ?T  
 ② 

E	perform	?M
?M	film	?T

  
 ③ longer (another triple with M)  
 T  $\equiv$  answer not type

# Qualifiers + Reification in KGs

- Not all facts are truly binary

**Christopher Nolan** (Q25191)

nominated for



Academy Award for Best Writing, Original Screenplay

for work

Memento

Q1

statement is subject of

74th Academy Awards

Q2

qualifiers

→ Wikidata specific

Freebase

Mediator nodes M  
- Compound Value Types (CVTs)

CN

nominated AA  
- for what?  
- Which AA?

role

LD member of group  
- role?

marriage

TC ma MR  
- ST?  
- ET?

TC ma kth  
- ST?  
- ET?

# Qualifier models

Facts with qualifiers — Dump (RDF)  
 n-ary facts — KG (graph)

LD : starred in Inception  
 — role Cobb

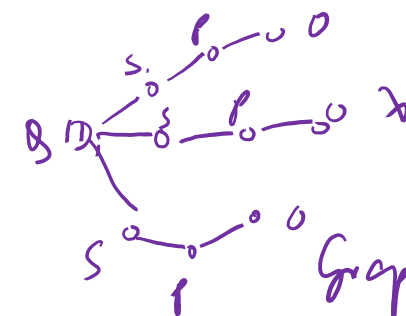
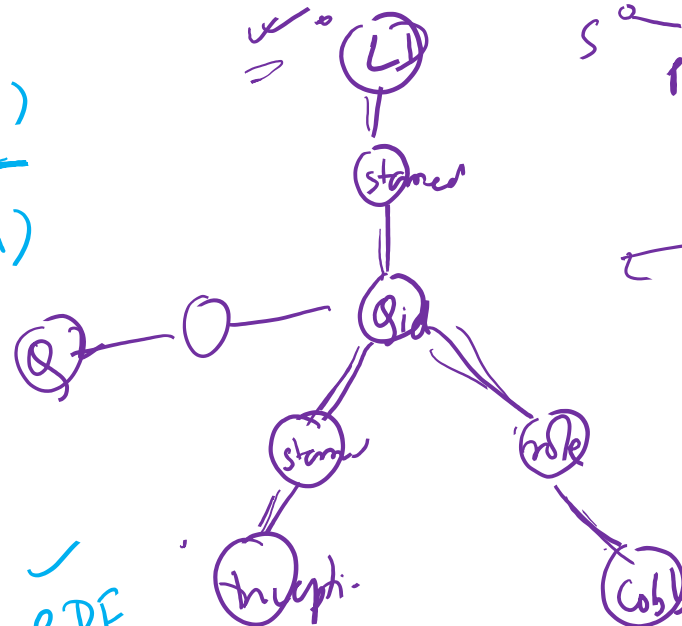
LD starred in Qid  
 Qid starred in Inc<sup>3253</sup>  
 Qid role Cobb

verification

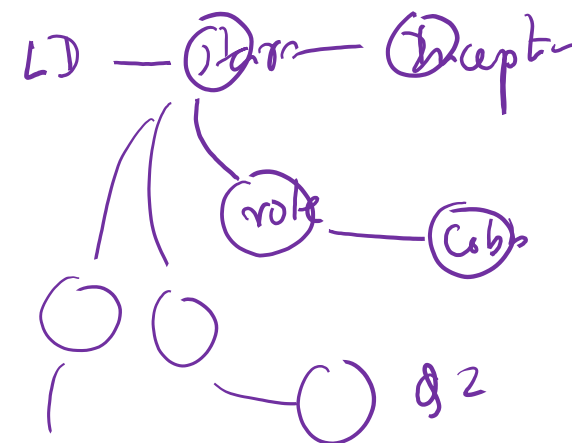
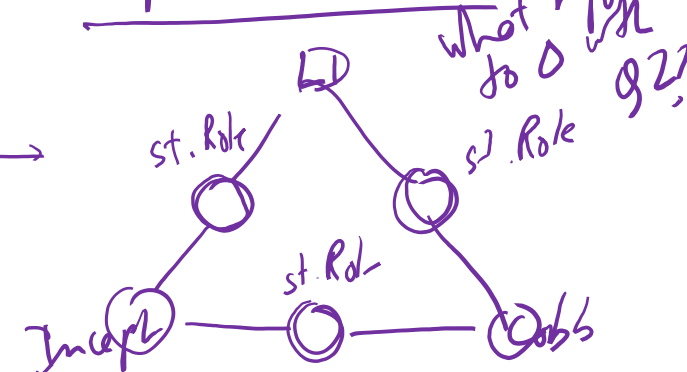
S	P	O

triple ✓  
 RDF

Dump (main) ↓  
 - preserve triple structure  
 - maintains equidistance



Graph alternative





# ① Entity matching

No NERD

due Web 12 corpus  
with ent annotation  
FACC1 → google



a ■ POS tagging (Nom, verb, adj, ...) NN, NN P

b ■ Subsequence generation multword entities FN, NN, NN P (single) → do not split

c ■ Find matching entries → Ellen, pinkie, pinkie name ↔ Entities lexicon

d ■ Scores for the entity matches → scored lexicon

e ■ Popularity scores for each entity

Cross Wikis

mention: s e

wiki anchor ↔ Freebase / Wikipedia

$p_{cross}(e|s)$

$$p_{pre}(e'|s) = p(e_{max}|s) \cdot \frac{pop(e')}{pop(e_{max})}$$

Freebase aliases  
Leo and d (ent) ↔ Leo (surface) ↔ CR7  
CR7

# Candidate generation

- Template 1
- Template 1
- Template 3

*query templates*

# Aqqu query templates

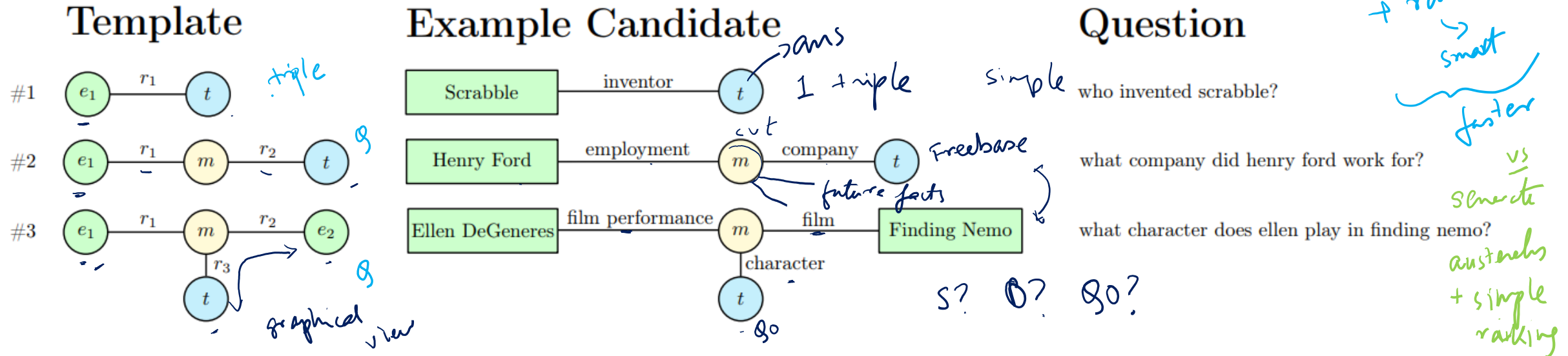


Figure 1: Query templates and example candidates with corresponding questions. A query template can consist of entity placeholders  $e$ , relation placeholders  $r$ , an intermediate object  $m$  and the answer node  $t$ .

how to search  $(e, r, \cdot)$   $e_1, r_1, m$   
 $m, r_2, e_2$

# Relation matching

You have many queries  $\sim 200-500$

1. Literal matches

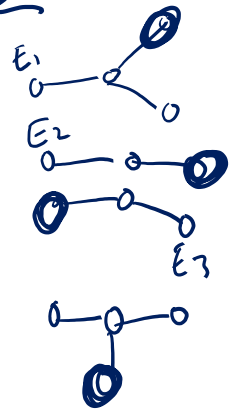
2. Derivation matches

3. Synonym matches

4. Context matches

$W = Q$   
 $\{w_1, w_2, w_3\}$   
 derivation  
 $q_1 \in QW$   
 $q_2$   
 $q_3$

$w_1 \in RW_c$   
 $w_2$   
 $w_3$



Maps: dictionaries: WordNet

{high, height, heights, higher, ...}

cat -> pussy - kitten

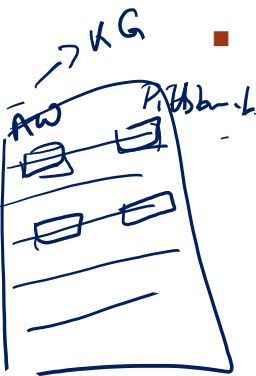
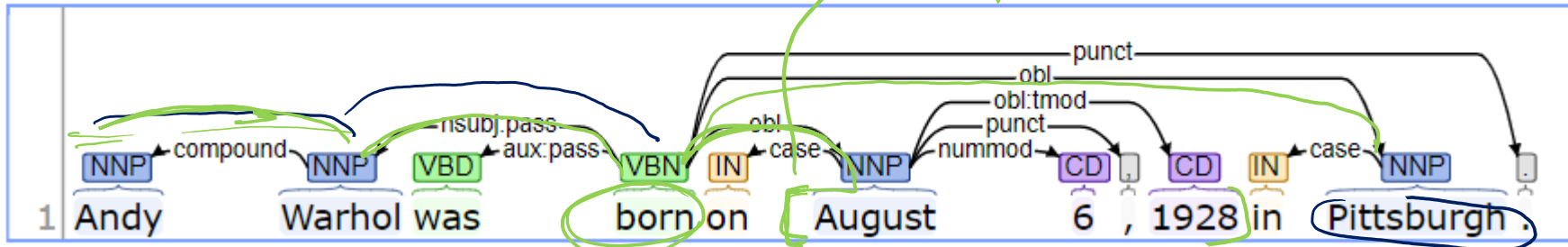
word2vec -> 0.4, spawn, wife, husband -> similar context

Andy Warhol was born on August 6, 1928 in Pittsburgh

he, she, Paraphrases, heuristic / rules, fast

Basic Dependencies:

Wiki Markup, coreference resolution, NLP



# Answer type matching

- ~~Coarse~~ ~~Course~~-grained types

Time?

> Pruning

who is the founder of apple?

- Who

- Where → location / event

- When  
    └─ sporting season  
    └─ date

Binary check  
    Y/N

target types  
    ↓  
    answer

Query candidate

answer?  $\underline{t}$

types →  $\begin{Bmatrix} \circ \\ \circ \\ \circ \end{Bmatrix}$  person

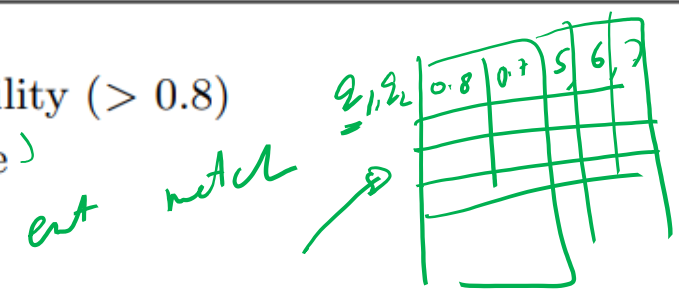
# Candidate features

- ① ■ Entity/Relation matching features
- ② ■ N-gram relation matching features
  - where is reggie bush from? (asking for the place of birth)
  - what to do downtown san francisco? (asking for tourist attractions)

→ *benchmarks*

*employment + work → <sup>KG</sup>employment · company*

ID	Description
1	<u>number</u> of <u>entities</u> in the query candidate
2	number of <u>entities</u> that matched exactly with their <u>name</u> , or with a high <u>probability</u> ( $> 0.8$ )
3	<u>number</u> of tokens of all entities that matched <u>literally</u> as per the <u>previous</u> <u>feature</u>
4-5	<u>average</u> (4) and <u>sum</u> (5) of entity match <u>probabilities</u>
6-7	<u>average</u> (6) and <u>sum</u> (7) of entity match <u>popularities</u>
8	number of relations in matched template
9	number of <u>relations</u> that were matched <u>literally</u> via their name
10-13	number of tokens that matched a relation of kind: <u>literal</u> (10), <u>derivation</u> (11), <u>synonym</u> (12), <u>context</u> (13)
14	sum of synonym match scores
15	<u>sum</u> of relation context match scores
16	number of times the answer relation ( $r_1, r_2, r_3$ for templates 1, 2 and 3 respectively) occurs in the KB
17	a value between 0 and 1 indicating how well the relation matches according to n-gram features (Section 4.5)
18	<u>sum</u> of features 3 and 10; that is, the number of tokens matching a relation or entity <u>literally</u>
19	number of tokens that match an entity or relation divided by the total number of tokens in question
20-22	whether the result size is 0 (feature 20), 1-20 (feature 21), or larger <u>than 20</u> (feature 22); all binary
23	binary result of the answer-type check (Section 4.4)



rel match cat dog → Vocab  
1 1 1.0 1.0

**Table 1: Features used by our ranking approaches. Top/middle/bottom: features for entity matches/features for relation matches/combined or other features.**

→ thoroughly engineered  
→ dense features



# Query ranking

- Pointwise ranking

- Pairwise ranking

- Classifier

- Linear

- Random forest



Learning-to-rank with ordinal regression  
 LTR  
 supervised ranking  
 ranks  
 → 1 ranked list

$Q, q_1, q_2$   
 $q_1 \geq q_2 ?$   
 $q_1 < q_2 ?$

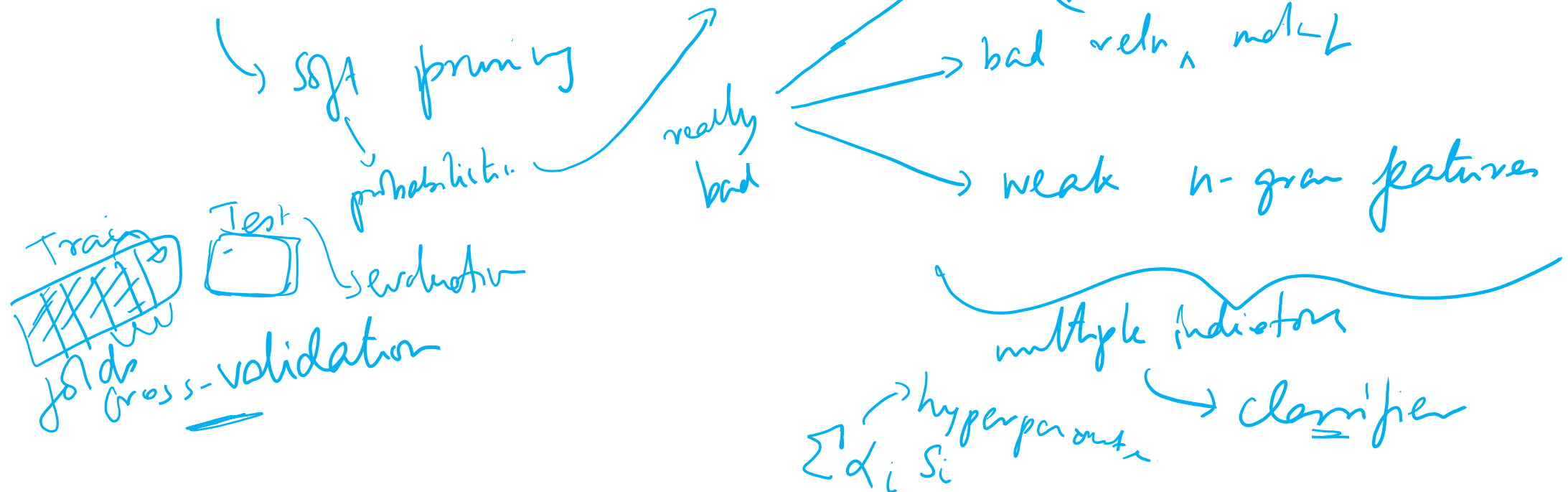
$$\phi_{\text{pair}}(a, b) = (\underbrace{\phi(a)}_{2 \times 1} - \underbrace{\phi(b)}_{2 \times 1}, \underbrace{\phi(a)}_{2 \times 1}, \underbrace{\phi(b)}_{2 \times 1})$$

23x3



# query Candidate pruning

- Without n-grams, with hard pruning
- With n-grams, with a pruning classifier



# Evaluation: Main

	Free917		WebQuestions
Method	Accuracy+	Accuracy	Average F1
Cai+Yates	59 %	—	—
Jacana	—	—	35.4 %
Sempre	62 %	52 %	35.7 %
Kwiat. et al	68 %	—	—
Bordes et al	—	—	39.2 %
ParaSempre	68.5 %	46 %	39.9 %
<b>Aqqu</b>	<b><u>76.4 %</u></b>	<b><u>65.9 %</u></b>	<b><u>49.4 %</u></b>

**Table 2: Results on the Free917 (267 questions) and WebQuestions (2032 questions) test set. For the results in the second column (Accuracy+) a manually crafted entity lexicon was used.**

*Yih et al*  
*STAGS*  
*ACL '15 - July*  
*CVFMMIS Oct*

# Evaluation: Top-k

important!!!

near-misses

	Top-2	Top-3	Top-5	Top-10
Free917	74.3 %	77.2 %	79.3 %	83.7 %
WebQuestions	67.1 %	72.7 %	77.5 %	82.3 %

→ Better ranking  
↓  
room for improvement  
↓  
inspiration

**Table 3: Top-k results on Free917 (top) and WebQuestions (bottom). Percentage of questions with the best answer in the top-k candidates.**

# Evaluation: Efficiency

- 644 ms for Free917
- 900 ms for WebQuestions
- Competitors slower – from source code
- Training on WebQuestions: 90 minutes in all!

test

Anxiety

~3k question

↓  
offline

# Why is Aqqu fast?

- ①
  - Minimal reliance on NLP *< understanding - slow*
  - No reliance on NERD *- know. coref. - sparse DP*
- ②
  - No question templates, directly create queries
  - Leverage benchmark insights
- ③
  - Smart pruning everywhere */ possibilities*
- ④
  - Index-based lookups *CrossWiki, map, ... dict*
- ⑤
  - Dense feature vectors *[ 0 0 -X ... ]*

# Research paper 2

## Towards a Question Answering System over the Semantic Web

D. Diefenbach

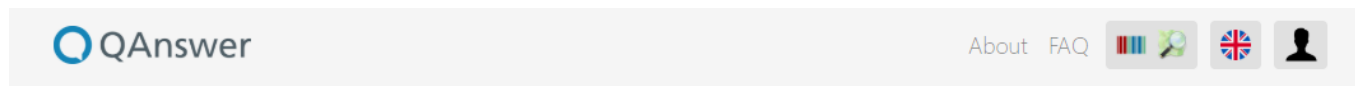
Now

Wdaqua (demo)

Wkshop

<b>Qanary—a methodology for vocabulary-driven open question answering systems</b> A Both, D Diefenbach, K Singh, S Shekarpour, D Cherix, C Lange European Semantic Web Conference, 625-641	37	2016
<b>Wdaqua-core0: A question answering component for the research community</b> D Diefenbach, K Singh, P Maret Semantic Web Evaluation Challenge, 84-89	31	2017
<b>Towards a message-driven vocabulary for promoting the interoperability of question answering systems</b> K Singh, A Both, D Diefenbach, S Shekarpour 2016 IEEE Tenth International Conference on Semantic Computing (ICSC), 386-389	24	2016
<b>Towards a question answering system over the semantic web</b> D Diefenbach, A Both, K Singh, P Maret Semantic Web, 1-19	19	2018
<b>The Qanary Ecosystem: getting new insights by composing Question Answering pipelines</b> D Diefenbach, K Singh, A Both, D Cherix, C Lange, S Auer International Conference on Web Engineering, 171-189	19	2017
<b>Question answering benchmarks for wikidata</b> D Diefenbach, T Tanon, K Singh, P Maret	14	2017
<b>Trill: A reusable front-end for qa systems</b> D Diefenbach, S Amjad, A Both, K Singh, P Maret European Semantic Web Conference, 48-53	14	2017
<b>Wdaqua-core1: a question answering service for rdf knowledge bases</b> D Diefenbach, K Singh, P Maret Companion Proceedings of the The Web Conference 2018, 1087-1091	13	2018
<b>Qanary—the fast track to creating a question answering system with linked data technology</b> K Singh, A Both, D Diefenbach, S Shekarpour, D Cherix, C Lange European Semantic Web Conference, 183-188	13	2016
<b>Pagerank and generic entity summarization for rdf knowledge bases</b> D Diefenbach, A Thalhammer European Semantic Web Conference, 145-160	9	2018
<b>Introducing feedback in qanary: How users can interact with qa systems</b> D Diefenbach, N Hormozi, S Amjad, A Both European Semantic Web Conference, 81-86	8	2017
<b>QAnswer: A Question Answering prototype bridging the gap between a considerable part of the LOD cloud and end-users</b> D Diefenbach, PH Migliatti, O Qawasmeh, V Lully, K Singh, P Maret The World Wide Web Conference, 3507-3510	6	2019

# Try it out!



Enter your question...

Go

Where is the inventor of dynamite born? What is the nationality of Jackson Pollock? Who is the author of Foundation?  
How many inhabitants has Southampton? communes in the province of biella Who is the director of A Clockwork Orange?  
List me temples in Athens. In which countries are the alps? museums in berlin Who is the author of A Game of Thrones?  
Who are the actors of Titanic? people who play chess Give me museums in Turin. formula water  
Who is the creator of Breaking Bad? brands of soft drinks Who is Tom Cruise?

*Samples →*

*have fun!!*

*link →*

<https://qanswer-frontend.univ-st-etienne.fr/>



# The QAnswer system

- Fast and effective *→ speed*
- Interpretable *→ accuracy* | *simple*
- Leverages KG structure *→ graph*
- Syntax agnostic for robustness *→ syntactical structure*
- Relies on efficient indexes *→ RDF* *→ HDT* *→ dense philosophy*

# The QAnswer system

- 1 ■ Expansion
- 2 ■ Query construction
- 3 ■ Query ranking
- 4 ■ Answer decision *innovation*
  - Also known as Wdaqua-core1  
*Wikidata*

# Expansion

- Search IRIs

→ Internationalize RI

: finding, news, n-grams

- Use n-grams

KG things

for search  
remove stopwords

stemming  
finding, find

dbr: entity    dbo, dbp: pred,    dbr:c: class

n	start	end	n-gram	resource	n	start	end	n-gram	resource
1	2	3	philosophers	dbr:Philosophes	52	5	6	saint	dbr:SAINT_(software)
2	2	3	philosophers	dbr:Philosophes	53	5	6	saint	dbr:Saint
3	2	3	philosophers	dbo:Philosopher	54	5	6	saint	dbr:Boxers_and_Saints
4	2	3	philosophers	dbr:Philosophers	55	5	6	saint	dbr:Utah_Saints
5	2	3	philosophers	dbr:Philosopher	56	5	6	saint	dbr:Saints,_Luton
6	2	3	philosophers	dbr:Philosophy	57	5	6	saint	dbr:Baba_Brooks
7	2	3	philosophers	dbo:philosophicalSchool	58	5	6	saint	dbr:Battle_of_the_Saintes
8	3	4	born	dbr:Born,_Netherlands	59	5	6	saint	dbr:New_York_Saints
9	3	4	born	dbr:Born_(crater)	:				
10	3	4	born	dbr:Born_auf_dem_Dar?	:				
11	3	4	born	dbr:Born,_Saxony-Anhalt	106	5	6	saint	dbp:saintPatron
:					107	5	6	saint	dbp:saintsDraft
42	3	4	born	dbp:bornAs	108	5	6	saint	dbp:saintsSince
43	3	4	born	dbo:birthDate	109	5	6	saint	dbo:patronSaint
44	3	4	born	dbo:birthName	110	5	6	saint	dbp:saintsCollege
45	3	4	born	dbp:bornDay	111	5	6	saint	dbp:patronSaintOf
46	3	4	born	dbp:bornYear	112	5	6	saint	dbp:patronSaint(s)
47	3	4	born	dbp:bornDate	113	5	6	saint	dbp:patronSaint'sDay
48	3	5	born in	dbp:bornIn	114	5	7	saint etienne	dbr:Saint_Etienne_(band)
49	3	5	born in	dbo:birthPlace	115	5	7	saint etienne	dbr:Saint_Etienne
50	3	5	born in	dbo:hometown	116	5	7	saint etienne	dbr:Saint-Étienne
					117	6	7	etienne	dbr:Étienne

Table 2

Expansion step for the question “Give me philosophers born in Saint Étienne”. The first column enumerates the candidates that were found. Here, 117 possible entities, properties and classes were found from the question. The second, third and fourth columns indicate the position of the n-gram in the question and the n-gram itself. The last column is for the associated IRI. Note that many possible meanings are considered: line 9 says that “born” may refer to a crater, line 52 that “saint” may refer to a software and line 114 that the string “Saint Étienne” may refer to a band.

# Query construction

①

SELECT / ASK var  
WHERE { s1 s2 s3 . }

0/1 → SPARQL

③

SELECT ?x  
WHERE { VALUES ?x {iri} . }

What is Apple?  
Apple - KG

②

SELECT / ASK var  
WHERE { s1 s2 s3 .  
          s4 s5 s6 . }

tuple pattern

from expansion

with

s1, ..., s6 ∈ R ∪ V

and

var ∈ {s1, ..., s6} ∩ V







answer

④

SELECT ?x  
WHERE { VALUES ?x {iri} .  
          iri ?p iri1 . }

Who is Steve Jobs from Apple?

# Ranking

- Cover words  

- Edit distances   

- Node degrees  

- Query properties
-  ■ #Variables
-  ■ #Triples (patterns)

# Ranking

→ avoid repetition

1 - SELECT DISTINCT ?y WHERE {  
dbr:Saint\_(song) ?p ?x .  
?x dbo:hometown ?y . }

2 - SELECT ?x {  
VALUES ?x { dbr:Saint\_Etienne\_(band) } }

3 - SELECT DISTINCT ?y WHERE {  
?x dbo:birthPlace dbr:Saint-Etienne .  
?x dbo:birthDate ?y . }

4 - SELECT DISTINCT ?y WHERE {  
?x ?p dbr:Philosophy .  
?x dbo:birthDate ?y . }

lots of queries!!  
~ 395 queries  
like AGG

none of them are perfect

# Answer decision

- Tackles unanswerability issue
- Using logistic regression classifier and threshold

→ whether to return an answer

$$P_{g_1} \geq \theta_2 \quad ?$$

↓  
threshold



# Handling implicit questions

- Give me German mathematicians

*user intent*

```
SELECT DISTINCT ?x WHERE {  
  ?x ?p1 dbr:Mathematician .  
  ?x ?p2 dbr:Germany .  
}
```

Here ?p1 is:

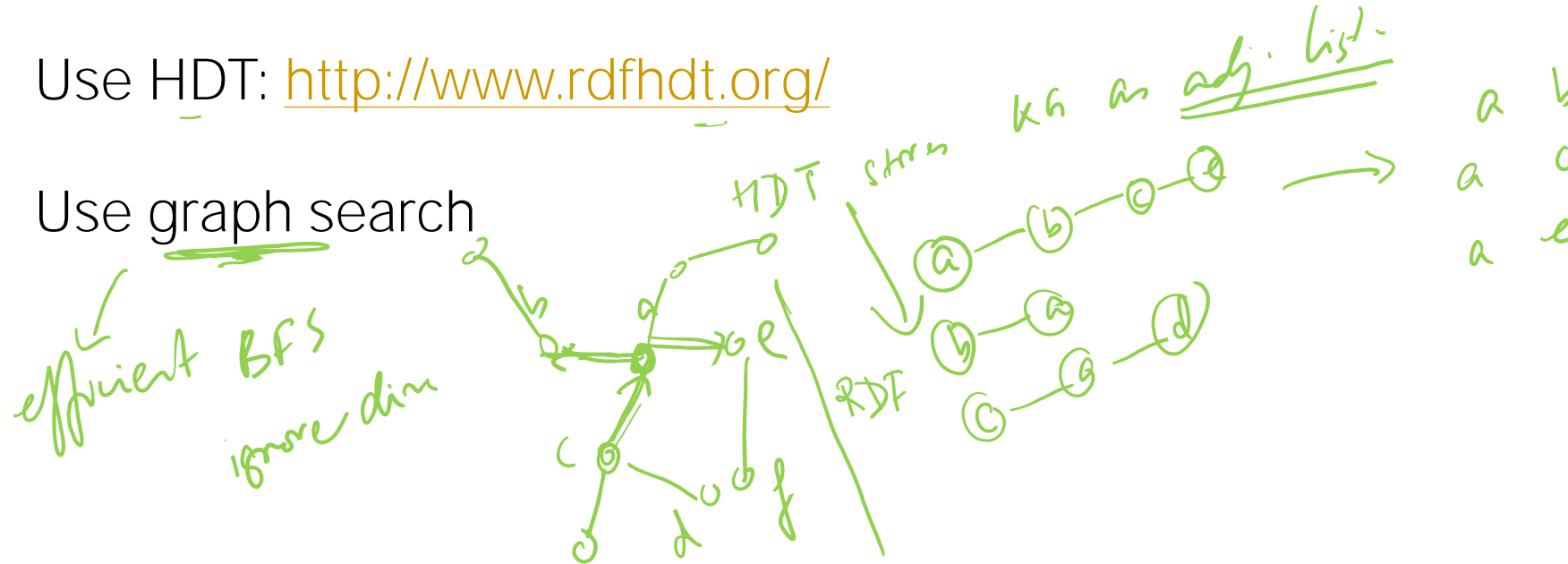
- dbo:field
- dbo:occupation,
- dbo:profession

and ?p2 is:

- dbo:nationality,
- dbo:birthPlace,
- dbo:deathPlace,
- dbo:residence.

# Fast candidate generation

- Use HDT: <http://www.rdfhdt.org/>
- Use graph search



## HDT – Your binary format for RDF

"HDT compresses big RDF datasets while maintaining search operations" ([Read more](#))

# What is HDT?

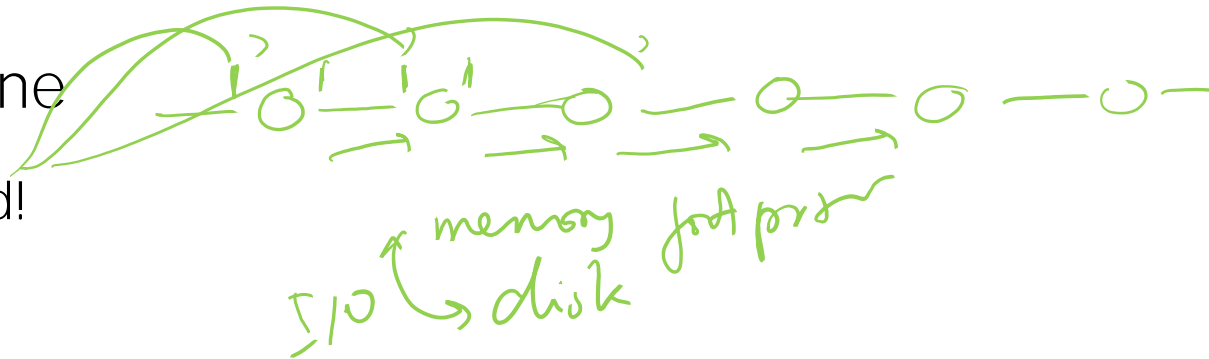
- Currently RDF data is stored and sent in very verbose textual serialization formats that waste a lot of bandwidth and are expensive to parse and index
- [HDT (Header, Dictionary, Triples) is a compact data structure and binary serialization format for RDF that keeps big datasets compressed to save space while maintaining search and browse operations without prior decompression
- Ideal format for storing and sharing RDF datasets on the Web

# More on HDT

- 1 ■ The size of the files is smaller
- 2 ■ The HDT file is already indexed
- 3 ■ High performance querying
- 4 ■ Highly concurrent
- 5 ■ The format is open
- 6 ■ The libraries are open source *adapt*

# Why is QAnswer fast?

- No reliance on syntax
- Simple query patterns
- Minimal pre- and post-processing
- No elongated pipeline
  - Think about overhead!
- Efficient use of HDT
- Relies on graph search



# Conclusions

- Efficiency should always be primary criterion in QA
- Think while making design choices!
- Use NLP techniques sparingly 😊 *NLP → addsh grammar*
- Efficiency versus accuracy are often trade-offs *template, entities, grammar...*
- Prune candidates wherever possible *→ nops, dict, entities, -ner...*
- Create efficient lookup indices whenever possible
- Use smart data structures *HDT*

Thank  
you