YodaQA

Technology as of 2016-01-22

Outline

- Factoid QA & YodaQA
- YodaQA Architecture:
  - Question Analysis
  - Database Search
  - Answer Scoring

These slides are quick adaptation of another presentation, and miss academic work citations. Sorry!

Factoid QA & YodaQA

Factoid QA

Extension of **search**, but:

- naturally phrased question instead of keywords
- output is not a whole document, but just the snippet of information

Ideal for **voice**:

- people ask in whole sentences
- we reply with just what the user needs to know

Our work in QA

Ailao / **Medialab** / **eClub** / **Cyber.FEE.CTU**
Petr Baudiš + several undergrad students (→ ~24 man-months)

YodaQA:

- universal end-to-end QA pipeline with basic functionality
- research vehicle for more advanced strategies
- machine learning instead of hardcoded rules
- Java, Apache UIMA, Apache Solr, RDF/SPARQL
- proof-of-concept web+mobile interface, public live demo

Factoid QA Datasets

Text data (unstructured): initial focus

- Wikipedia, websites
- NLP, information extraction problem

Databases (RDF graph): **current work, today's topic**

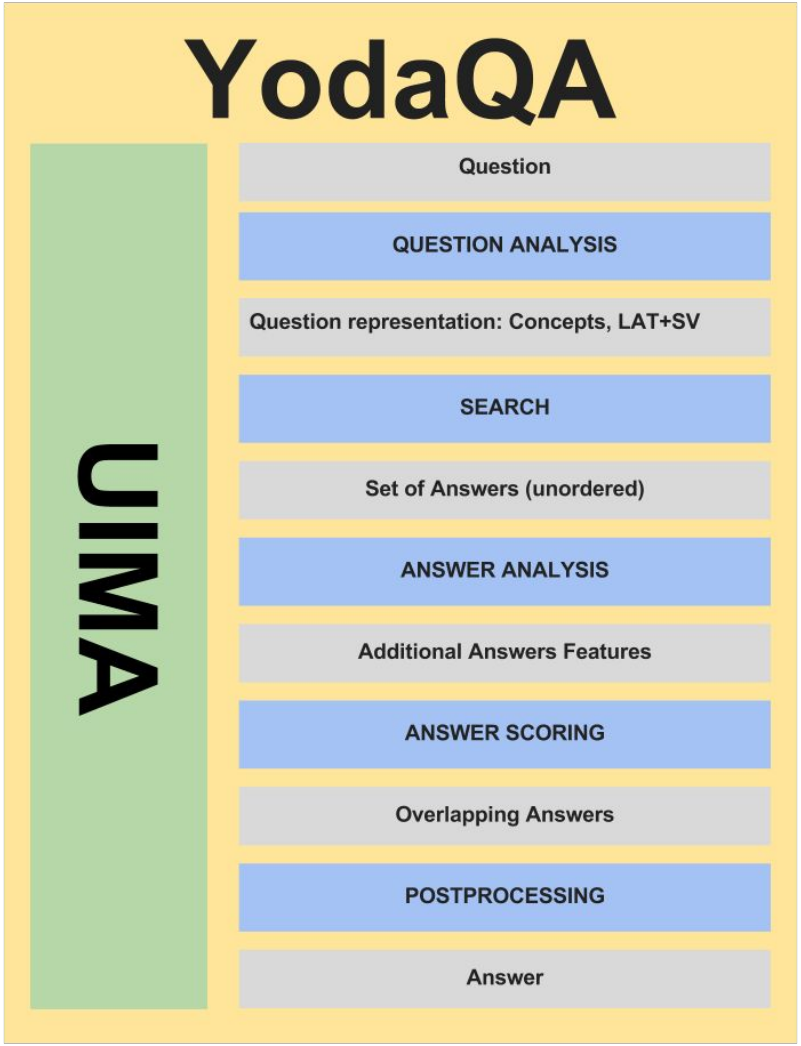
- DBpedia (Wikipedia infoboxes)
- Freebase (Google Knowledge Graph) → movie questions

Synthesis is possible (and done!)

YodaQA Operation Principles

- Question Analysis
- Database Search
- Generate many candidate answers
- Answer Typing & Scoring

Reminder: Full-text QA portion of pipeline omitted in these slides! Only database search!



Question Analysis

Example

Who played Marge in The Simpsons?

___

NLP Analysis

Output: linguistic representation of the question

- OpenNLP segmenter
- StanfordParser (pos + dependencies + constituents)
- LanguageTool lemmatizer
- OpenNLP NER off-the-shelf models: person, place, date, …

NLP Analysis

Who played Marge in The Simpsons?

```
c ROOT null [Who played Marge in The Simpsons?]
 c SBARQ null [Who played Marge in The Simpsons?]
  c WHNP null [Who]
   t WP who [Who]
  c SQ null [played Marge in The Simpsons]
   c VP null [played Marge in The Simpsons]
    t VBD play [played]
    c NP null [Marge]
     t NNP Marge [Marge]
    c PP null [in The Simpsons]
     t IN in [in]
     c NP null [The Simpsons]
      t DT the [The]
      t NNP Simpsons [Simpsons]
  t . ? [?]
```

Lexical Answer Type (LAT)

- **Focus** identification
- Focus name proxy
- **LAT** by focus
- LAT by Wordnet
- **Selection verb** (and LAT by SV)

Lexical Answer Type (LAT) - Focus

Question focus identification - This is the focus point of the sentence where you should be able to place the answer.

> What was the first **book** written by Terry Pratchett?
> The **actor** starring in Moon?
> **Where** is Mt. Olympus
> **When** was the U.S. capitol built
> **Who** played Marge in The Simpsons?

Lexical Answer Type (LAT) - Focus

Blackboard rules based on specific dependencies and pos tags detected in the sentence (about 5 rules), e.g.:

- If there's ADVMOD (adverb modifier) dependency pointing at lemma "how", the other end is the focus (e.g. "how"-"**old**")
- As a fallback, the NSUBJ (noun subject) dependency endpoint is the focus (e.g. "**who**"-"did"-"X", "**spouse**"-"of"-"Madonna")

Lexical Answer Type (LAT)

LAT By Focus - transform focus to LAT; mostly just copy it over, plus blackboard “where” -> “place” etc.

> Who → **person**

LAT By Wordnet - expands existing LATs, adding more LATs, by hypernymy relations in Wordnet

> (“biologist” -> “scientist” -> “person” -> “living being”)

Lexical Answer Type (LAT) - Selection verb

SV (selection verb) identification - semantically meaningful verb in the sentence

> where is Mt. Olympus *(none)*
> When was the U.S. capitol **built**
> Who **invented** telephone?
> Who **played** Marge in The Simpsons?

Simple heuristic rule based on the parse tree root

Question Clues

Search keywords and keyphrases

- **Noun phrases** (constituents, parser output)
  Marge, Simpsons
- **Nouns**
  Marge, Simpsons
- **Named entities** (e.g. Paris [location], Abraham Lincoln [person], 1984 [date], etc.)
- **LAT and SV** from the above
  played

Entity Linking

Input: clues
Output: set of concepts (links to DBpedia - enwiki articles)

- Candidate generation
- Entity lookup
- Disambiguation
- Deduplication

Entity Linking - Candidate Generation & Lookup

- Candidates from **clues**
- Candidates from 2,3,4-grams
- Lookup by **fuzzy search** in entity names+aliases
  (sorted by *popularity* - number of cross-links)
- Lookup in **crosswikis** (labels used as enwiki link texts)
  (sorted by *probability* of linking the particular entity)

Entity Linking - Candidate Generation & Lookup

- Candidates from **clues**

Marge:
crosswiki: **Marge Simpson** (p=0.858966)
fuzzy lookup: **Marge** (d=0), Margay (d=2), Margaz (d=2)
    List of supporting Harry Potter characters
    Marge (cartoonist)
    Marge Burns

The Simpsons: fuzzy **Simpsons**, matches both series and movie!

Entity Linking - Disambiguation

Classifier **scoring** candidate entities to be linked to the keyword/keyphrase (top 5 kept; **logistic regression**)

Features: p_crosswiki, log-popularity, edit distance, origin

```
<<The Simpsons Movie>> → 0.784, d=0, lprob=0.872, logpop=4.
997
<<The Simpsons>> → 0.667, d=0.00, lprob=0.872, logpop=5.710
<<Marge Simpson>> → 0.376, d=0.00, lprob=0.859, logpop=3.829
<<Marge Champion>> → 0.112, d=0.00, lprob=0.000, logpop=4.
522
<<List of supporting Harry Potter characters>> → 0.079,
```

Question Representation

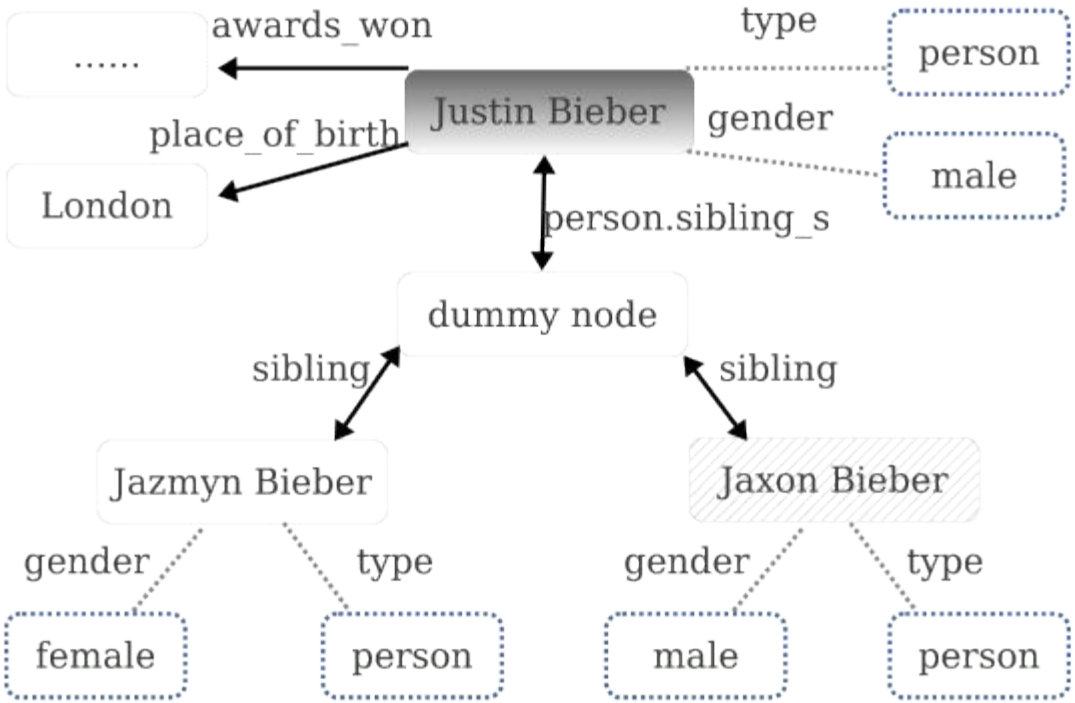
- **LAT**, **SV** (“semantics”)
- **Concepts** (entity linking)
- **Clues** (search keywords;
  in database-only task, mostly superseded by concepts)

Database Search

Knowledge Bases

RDF databases

- **Freebase** (robust)
- **DBpedia** (noisy, gappy!)
- Wikidata (work in progress)



Freebase Property-Paths

Multi-label classifier suggests **specific property paths** in the Freebase knowledge graph based on **specific words in the question** representation

LAT = director -> try property "/film/director/film"

Uses logistic regression (one-vs-rest, L1-regularized)

Freebase Property-Paths - Example

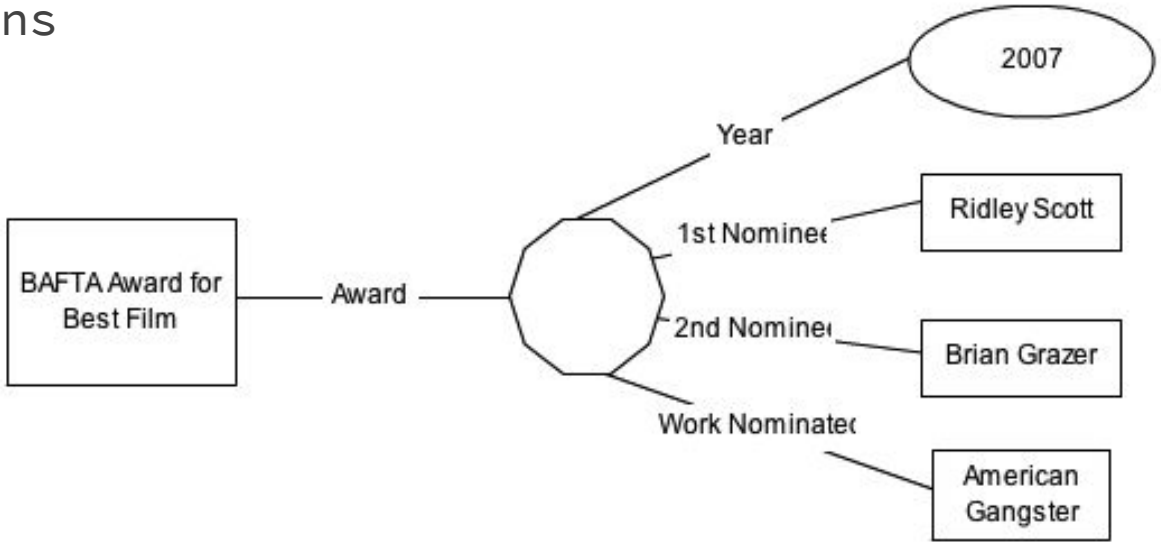
Who played Marge in The Simpsons?
sv=played, lat=person, …

- sports.sports_team.arena_stadium
- film.film.starring → film.performance.actor
- award.award_winning_work.awards_won → award.award_honor.award_winner
- film.film_character.portrayed_in_films → film.performance.actor
- film.film.starring → film.performance.actor (| film.performance.character)
- tv.tv_program.regular_cast → tv.regular_tv_appearance.actor (| tv.regular_tv_appearance.character)

Freebase Property-Paths - CVT

We need **combination**
of Marge & The Simpsons

Witness nodes
(secondary
question concept
matches)



Fallback Strategy: Any-property search

- Gather all immediate properties
  and generate candidate answers

  The Simpsons Movie
  Title ID/imdb.topic.title_id -> tt0462538
  Initial release date/film.film.initial_release_date -> 2007-07-21
  Genres/film.film.genre -> Comedy Animation Adventure Film Satire Parody
  Directed by/film.film.directed_by -> David Silverman
  Country of origin/film.film.country -> United States of America

- Not suitable for QA integration (introduces a lot of noise)

Freebase Graph Exploration

Meld property-path and any-property search

- Score relevancy of each property
- Explore some of the links deeper

Work in progress! (almost done)

Answer Scoring

Embedding-Based Selection

Goal: Match property label of the answer to question repr.

- Vector embedding (word2vec, **GloVe**) of question LAT+SV and the property label (bag of words - average)
- Project question embedding to label space
- Use dot-product to measure similarity (cos-dist. w/o norm)

$$s(q,a) = \sigma(q \cdot M \cdot a^T)$$

Train matrix M - machine learning, logistic regression

Embedding-Based Selection

Marge Simpson

Programs in which this was a regular character: Actor
(| The Simpsons)

Julie Kavner
prop-path score=0.132, GloVe score=0.011

Name
Marge Simpson
prop-path score=0.048, GloVe score=0.004

Lexical Answer Type for Answer Candidates

Many strategies for assigning an LAT to the answer:

- **Originating property** becomes LAT (+one-word normalization)
- Check if the answer triggers an OpenNLP NER stock model
- Look up the answer as article name in enwiki (DBpedia), use **DBpedia type** (category-based)
- “Quantity” LAT for numbers
- *instance-of* Wordnet attribute
- JoBimText (work-in-progress)

Lexical Answer Type for Answer Candidates

Julie Kavner
    actor (dbpedia, by category)
    person (by NER)

Marge Simpson
    name (by property)
    doctor, **divorcee** (by Wordnet)
    person (by NER)

Type Coercion

Goal: Match question LAT (e.g. "person") to the answer LAT (e.g. "director" or "date").

Uses Wordnet links between more general and less general words (hypernymy / hyponymy); number of steps ~ **TyCor score**.

> Fun fact: “**date**” is (i) time point identification; (ii) fruit; (iii) a *person* that you go on a date with
> → we use *(token, synset)* tuples whenever we can

Julie Kavner → person-person, perfect tycor
Marge Simpson → person-person, prefect tycor

Clue Overlaps

Checking if answer overlaps some of the question keywords (many false positives are like that).

```
Marge Simpson !
```

From Features to Scores

Extra features:

- Question Class (6-fold classification)
- Entity Linking Score
- Each feature: also normalized version and unset-indicator

Gradient-boosted Decision Forest (200 trees)
Who played Marge in The Simpsons? 1199 candidates, 308 unique
1. Julie Kavner (conf. 0.991)
2. Dan Castellaneta (conf. 0.092)
3. Hank Azaria (conf. 0.092)

YodaQA End-to-end

YodaQA Performance

KB: 62.9% on a collection of movie questions (Google: 59.9%)
Fulltext: 35% on general Wikipedia-based questions

http://ailao.eu/yodaqa
http://movies.ailao.eu/
http://live.ailao.eu/

Voice-activated prototype mobile app - download

Current Limitations

- **List queries** *(work in progress)*
  (+ “first”, “how many”)
- **Transformations** *(work in progress)*
  (“how old” → “birthdate” and back)
- **Compound queries** not supported yet
  (“When was the third wife of Tom Cruise born?”)
- **Performance:** Fast SPARQL is tricky
  (but maybe general SPARQL is overkill)

Questions? Shoot them at ailao@ailao.eu!