brmson (YodaQA)

A DeepQA-style Question Answering Pipeline

Petr Baudiš ⟨pasky@ucw.cz⟩

FEE CTU Prague

January 2015

Outline

Petr Baudiš

Second year **PhD student** at FEE CTU Prague
(Petr Pošík + Jan Šedivý),
Masters degree in AI from Charles University in Prague

Strong software engineering background: The original Git team, GNU libc development, many open source projects, freelancing

Solid AI, RL background: Computer Go research
(MCTS software Pachi — top OSS program, ~4th worldwide)

Basic ML, optimization background: Algorithm portfolios, etc.

Relative newbie in Natural Language Processing!

brmson

A Question Answering system inspired by **IBM Watson** and its DeepQA pipeline architecture.

Primary goals:

- Practicality
- Extensible design
- Academic reusability

Current aims: Open-domain factoid questions (TREC QA), replicating the DeepQA scheme with 75% recall, 37.5% accuracy-at-1.

Multiple implementations:
BlanQA (legacy), YodaQA (current)

brmson: BlanQA (legacy)

- **BlanQA:** *Legacy* pipeline based on CMU's OAQA
- Java, UIMA without CAS branching, UIMA-ECD
- Architecture based on OAQA helloqa `prototype` GitHub branch, but rewritten almost from scratch
- *enwiki* in solr, Ephyra answer type system, Ephyra modules provide the actual algorithms and rules
- Complete setup documentation, fairly clean code
- Interfaces: Interactive and chatbot (IRC)
- **Functional OAQA end-to-end pipeline!**

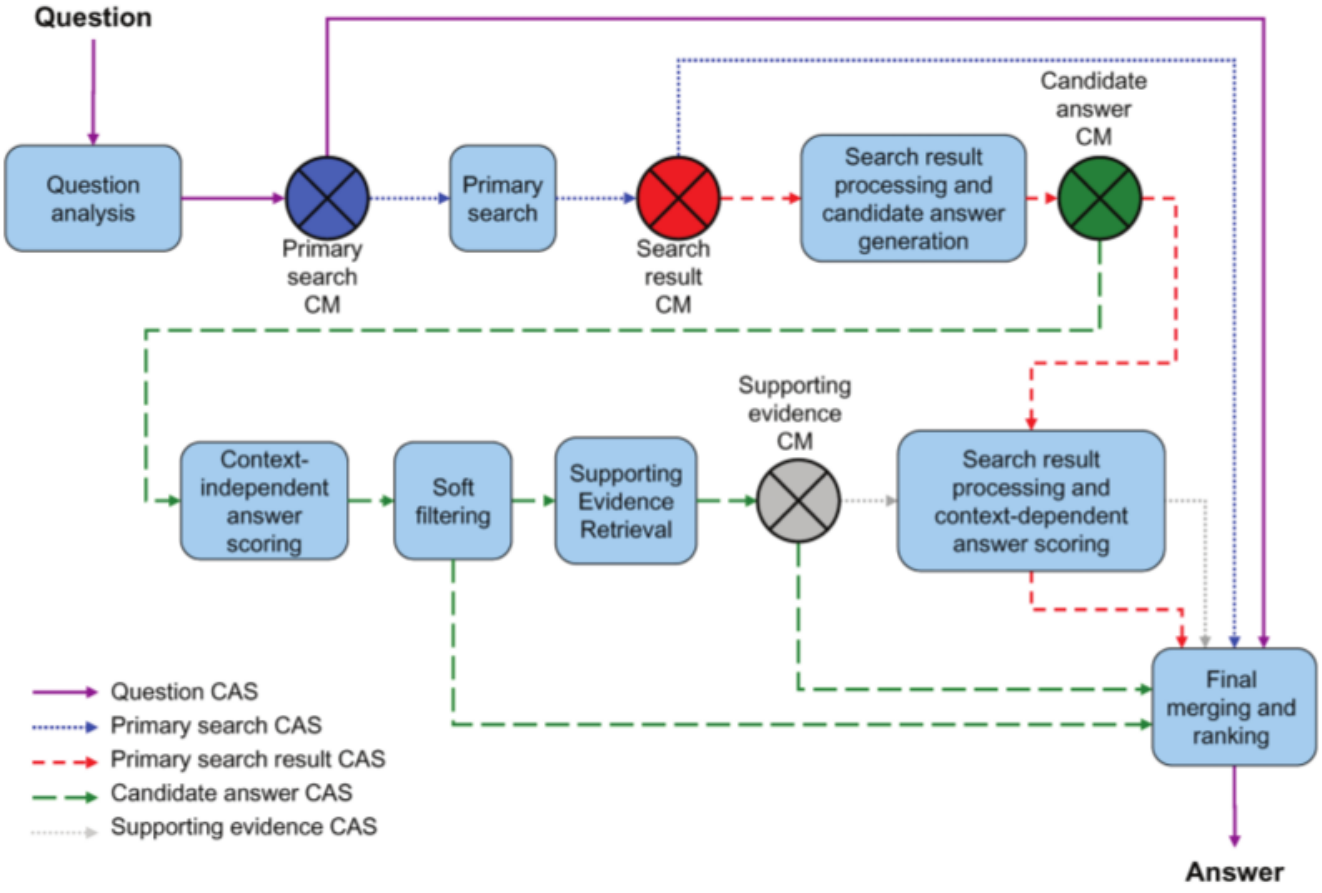
brmson: YodaQA (current)

- **YodaQA:** "Yet anOther Deep Answering pipeline"
- Designed and implemented from scratch — again
- Java, UIMA, DeepQA-style CAS branching, UIMAfit
- Architecture based on simplified DeepQA (as published)
- Every entity (question, retrieved document, answer) == CAS

- NLP analysis: Third-party UIMA annotators via DKpro
- Uses type coercion and general parse tree heuristics;
  no fixed regex patterns or pre-defined question types

Outline

YodaQA Pipeline



DeepQA architecture (Epstein et al., Making Watson fast).
Many modules in YodaQA are obviously much simpler.

Question Analysis

- Full dependency parse
- **Focus** generation (hand-crafted dependency, pos rules)
  - What was the first book written by Terry Pratchett?
  - The actor starring in Moon?
- **LAT** (Lexical Answer Type) generation (from focus)
  - ~~Where~~ is Mount Olympus? location
- **Clues** (search keywords, keyphrases) generation:
  - POS and constituent token whitelist
  - Named entities
  - Focus and the NSUBJ constituent
  - `enwiki` article titles

Outcome: Set of Clue and LAT annotations

Answer Production

Several answer production pipelines run independently in parallel.

- *SolrFull:* Passage-yielding search
  - *Fulltext:* Full-text + title search for clues, passages containing clues are considered
  - *Title-in-clue:* Title search for clues, initial passage is considered
  - Passages are parsed, NEs and NPs are answer candidates
- *SolrDoc:* Full-text search for clues, document titles are answer candidates
- *DBpedia:* Structured data, attributes of clue resources

Outcome: Set of candidate answers

Answer Analysis

- Each answer is POS-tagged and has dependency tree, Focus generated (dependency root)
- **LAT generation** — named entity type, DBpedia concept type, WordNet instance-of relation, rule for CD POS
- **Type coercion** of question + answer LAT: *Unspecificity* is path length in the WordNet (*hypernymy*, *hyponymy*) graph
- Answer features (help determine trustworthiness) for:
  - Phrase origin, clue overlaps
  - Generated LATs, type coercion
  - 81 features in total
- Logistic regression generates answer confidences

Outcome: Ordered set of Answers

Answer Merging and Evidencing

After initial scoring:

- Take just the top 100 answers
  (reduce noise for next classifiers)
- Measure overlaps with other answers and diffuse scores
- Re-score

After second scoring:

- Further evidence gathering — perform full-text search on the top five answers
- Re-score

Outcome: Ordered set of Answers

Outline

Testing Dataset

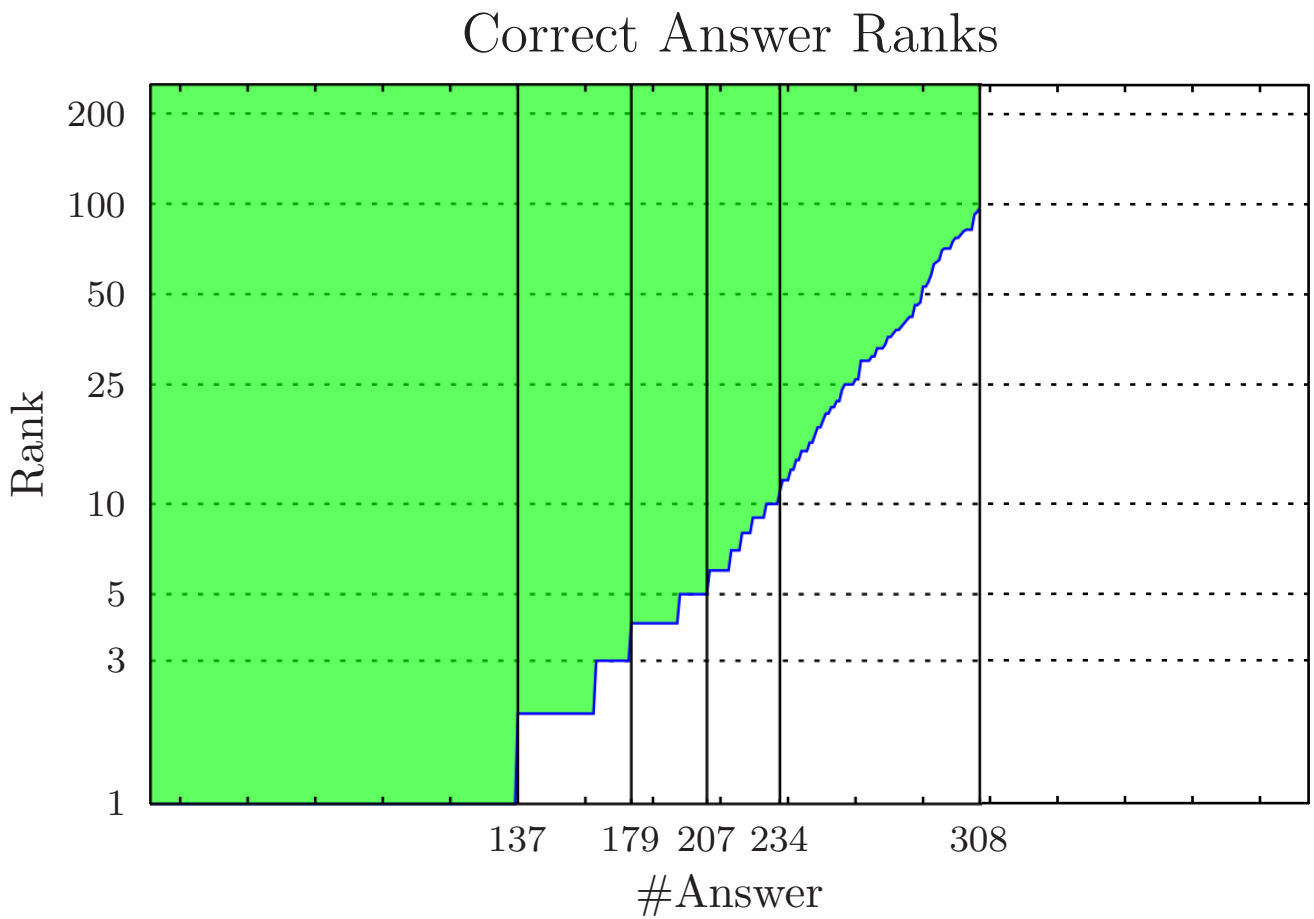
- TREC QA 2002 + 2003 XML datasets converted to plaintext
- Curated (pruned and with revised answer patterns), extended with an IRC BlanQA dataset
- 430 training questions (also used for development), 430 testing questions (held out)

- 2×430 is current practical limit for measurement turn-around (2-3 hour evaluation runs on my home computer)
- Matching correct answers with regexes has severe limits

Experimental Results (Test Set)

Candidate answer binary recall: 74.1%

Final answer accuracy-at-one: 31.6%



Analysis Tools

```
$ data/eval/trecnew-single200-measure.sh
...

$ data/eval/tsvout-stats.sh | head -n 5
3b46430 14-08-14 CluesMergeByText: Al... 29/134 14.5%/67.0% as 0.424
fdb239b 14-08-11 Revert "CluesToConce... 23/131 11.5%/65.5% as 0.395
4cd4a09 14-08-10 Clue: Add a label fe... 21/131 10.5%/65.5% as 0.390
e8ad387 14-08-10 SolrFullPrimarySearc... 24/131 12.0%/65.5% as 0.389
acf17eb 14-08-10 ClueConcept, CluesTo... 18/126  9.0%/63.0% as 0.372

$ data/eval/tsvout-compare.sh 01718ca 8fc9856
---------------- Gained answer to:
1424 Who wo... for best actor in 1970?    George C. Scott 0.00 1.00
---------------- Improved score for:
1417 Who wa... in less than four minutes? Roger Bannister 0.57 0.77
---------------- Worsened score for:
1408 Which ... Lionel Jospin a member of? Socialist       0.31 0.30
---------------- Lost answer to:
1427 What w... spaceship on the moon?     Eagle           0.04 0.00 $
```

AdaptWatson-style Analysis Snapshot

```
1407 When did the shootings at Columbine happen? | April 20\s?, 1999
     #stopwords/#synsets Columbine High School massacre does not
     appear; ignore "happen" or add synset that includes "occur"

1439 How deep is Crater Lake? | 1\s?,\s?932 feet
     #wikipieces "Crater Lake" yields "crater lake" matches and
     never the main article; also, it should be 1,943 feet

1666 What is the name of the US military base in Cuba? | Guantanamo
     #abbrev US -> U.S., then should work (answer Guantánamo!)

1606 What is the boiling point of water? | 212 degrees Fahrenheit
     100 °C

...

#wikipieces (7) - for all NPs/NEs/nouns in question, include same-
                  titled wikipedia articles in primary search;
                  furthermore, do not split such to sub-clues?
#synsets (6) - include synsets instead of words
#abbrev (4) - acronym generation / expansion; e.g. PC = P.C. =
              Personal Computer; in expansion, try using #redirects?
```

Outline

brmson: YodaQA vs. Primary goals

Practicality

- Detailed setup instructions (including data sources setup!)
- Detailed design documentation
- Interactive user interface
- Open source (ASL2 licence), clean code and build system

Extensible design

- UIMA + DeepQA structure: Easy pipeline branching and addition of new modules
- DKPro: Third-party UIMA annotators (tokenizers, parsers, etc.) are freely replaceable
- Internal UIMA components are as fine-grained as possible

Academic platform

- Gold Standard interface, TREC QA based dataset
- All datasets, evaluation tools and measurements published
- AdaptWatson methodology for performance analysis driving development

brmson: YodaQA vs. Primary goals

Practicality

- Detailed setup instructions (including data sources setup!)
- Detailed design documentation
- Interactive user interface
- Open source (ASL2 licence), clean code and build system

Extensible design

- UIMA + DeepQA structure: Easy pipeline branching and addition of new modules
- DKPro: Third-party UIMA annotators (tokenizers, parsers, etc.) are freely replaceable
- Internal UIMA components are as fine-grained as possible

Academic platform

- Gold Standard interface, TREC QA based dataset
- All datasets, evaluation tools and measurements published
- AdaptWatson methodology for performance analysis driving development

brmson: YodaQA vs. Primary goals

Practicality

- Detailed setup instructions (including data sources setup!)
- Detailed design documentation
- Interactive user interface
- Open source (ASL2 licence), clean code and build system

Extensible design

- UIMA + DeepQA structure: Easy pipeline branching and addition of new modules
- DKPro: Third-party UIMA annotators (tokenizers, parsers, etc.) are freely replaceable
- Internal UIMA components are as fine-grained as possible

Academic platform

- Gold Standard interface, TREC QA based dataset
- All datasets, evaluation tools and measurements published
- AdaptWatson methodology for performance analysis driving development

YodaQA: Future Work

TODO List for 1.0:

- Extra unstructured answer features (detailed clue-based context)
- Use Freebase as structured answer source (long-term: *solid* enwiki infobox based datasets)
- (?) Try different answer classifiers (random forests or MLP)
- Aiming to publish on CLEF2015

With more contributors:

- Cleaned up testing dataset
- UIMA component unit tests
- Verification dataset runs with human judges
- Insightful web interface
- Scale-out, parallelization and memory usage optimizations
- Apply to some real-world projects and domains

Long-term Plans and Goals

- Post-YodaQA architecture reformulation as IE problem:

  Latent knowledge graph paradigm

  (QA pipeline as on-demand population of semantic network; answer retrieved by path search, scored by edge coercion)

- brmson-based startup: Looking for good business cases
- Disembodied autonomous agent: QA with deduction + goal-setting + planning (maybe in 15 years)

- Personal: Internship at NII Tokyo in Q1 2015 (answering of Physics questions in university entry exams)

Conclusion

- Practical, open source QA system
  - Clean architecture and development methodology
  - Reasonably documented!
  - Clear path forward, towards reference experimental testbed

baudipet@fel.cvut.cz
petr.baudis@gmail.com
IRC: #brmson at freenode.net

Thank you for your attention!