brmson (YodaQA)

A DeepQA-style Question Answering Pipeline

Petr Baudiš ⟨`pasky@ucw.cz`⟩

FEE CTU Prague; brmlab hackerspace

Spring 2015

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)

NII: Physics questions (Tetsunari Inamura)

brmson

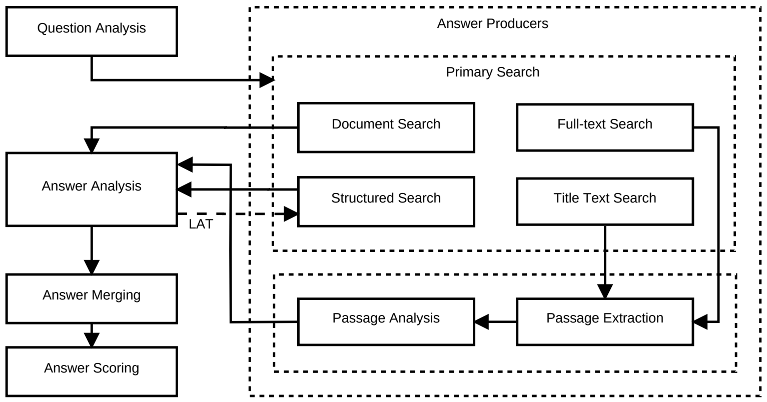
A Question Answering system inspired by **IBM Watson** and its DeepQA pipeline architecture.

Primary goals:

- Practicality
- Extensible design
- Academic reusability

Current status: Open-domain factoid questions (TREC QA), replicating the DeepQA scheme with 80% recall, 33% accuracy-at-1.

YodaQA Pipeline



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

Testing Dataset

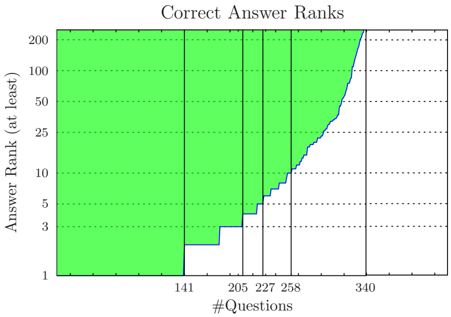
- TREC QA 2002 + 2003, curated and extended with an IRC BlanQA dataset
- 430 training questions (also used for development), 430 testing questions (held out)

- 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

Current State

Current performance:
32.6% accuracy-at-one
79.3% recall

Work in progress:
Better hypothesis generation,
smarter machine learning model.



brmson: YodaQA Implementation

- **YodaQA:** “Yet anOther Deep Answering pipeline”
- Designed and implemented from scratch
- Java, UIMA framework
- Architecture based on simplified IBM DeepQA (as published)
- NLP analysis: Third-party UIMA annotators via DKPro
- **Open Source!** Everything is on `github.com/brmson`, including documentation
- Looking for contributors, collaborators, commercial ideas...

YodaQA: Future Work

- Better and larger testing dataset
- Insightful web interface

- Scale-out, parallelization and memory usage optimizations
- Apply to some real-world projects and domains

- **Work in progress:**
  Better hypothesis generation,
  smarter machine learning model.
- Text understanding — distributed representations, deep learning approaches.

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)

Conclusion

- Practical, open source QA system
- Clean architecture, very modular system
- Reasonably documented!
- Long term:
  - Closed domain QA with powerful user interface
  - Bleeding edge NLP research (PhD)
  - Startup aims

`pasky@ucw.cz`
`petr.baudis@gmail.com`

Thank you for your attention!