YodaQA

A DeepQA-style Question Answering Pipeline

Petr Baudiš ⟨pasky@ucw.cz⟩

3C Group, FEE CTU Prague; Ailao

Spring 2015

Petr Baudiš

Second year **PhD student** at FEE CTU Prague
(Jan Šedivý + Petr Pošík),
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, ML background: Computer Go research
(MCTS software Pachi — top OSS program, ~4th worldwide)

YodaQA

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.

What questions do we look at?

Hi!

What's the time?

Do *you* dream of electric sheep?

Can you make me a program that prints all primes?

Can entropy ever be reversed?

How do you work?

誰があなたを作成しましたか?

What's the highest mountain in the world?

Only knowledge ("trivia", "factoid") questions.

Where to get the answer?

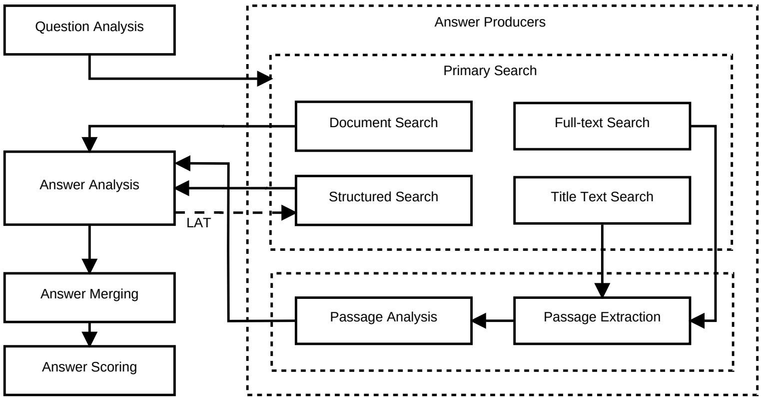
Unstructured knowledge bases (Wikipedia):

- Information Retrieval problem
- Information Extraction problem
- Type checking
- Related: Textual entailment

Structured knowledge bases (linked data):

- Machine Translation problem
  Convert freetext query to formal representation
  (e.g. SPARQL or lambda expression)

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
    - **Concepts:** `enwiki` article titles

Outcome: Set of Clue and LAT annotations

Recap: Question Representation

Bag of features question representation:

- Subject concept(s)
- Bag of clues (keywords, keyphrases)
- Answer type (LAT)

Future ideas:

- `population(?, new-york)`, `president(obama, ?)`
- Distributed representations

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

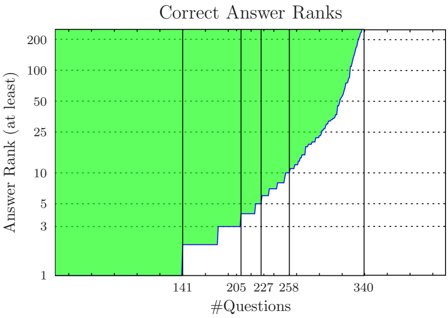
- TREC QA 2002 + 2003, curated and extended with an IRC BlanQA dataset
- 430 training questions (also used for development), 430 testing questions (held out)

- 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
30s per question

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



Baudiš, 2015: YodaQA: A Modular Question Answering System Pipeline

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, applications...
- **NEW:** Paid internship position for this summer!

YodaQA: Future Work

- Better and larger testing dataset
- Insightful web interface

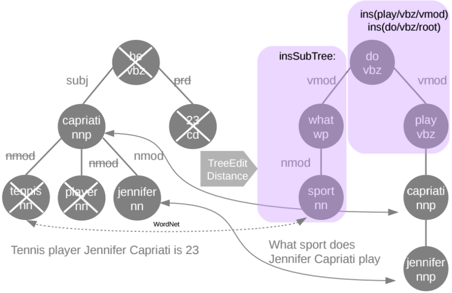
- Scale-out, parallelization and memory usage optimizations
- Apply to some real-world projects and domains

- **Work in progress:**
  Better hypothesis generation, smarter machine learning model.
- Better question representation
- IE, entailment — distributed representations,
  deep learning approaches.

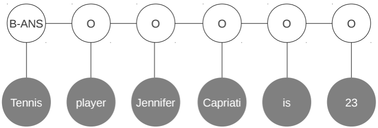
Sneak-peek: Jacana

Yao and van Durme, 2013: Answer Extraction as Sequence Tagging with Tree Edit Distance

- Sequence tagging of answer bearing passages.
- Linear-chain CRF tagging of tokens as B, I, O.
- Contextual features: POS, dependencies, NE
- Q-A alignment features: Tree edit model (distance, edit types, alignment)



Followup: Yao et al., 2013: Automatic Coupling of Answer Extraction and Information Retrieval

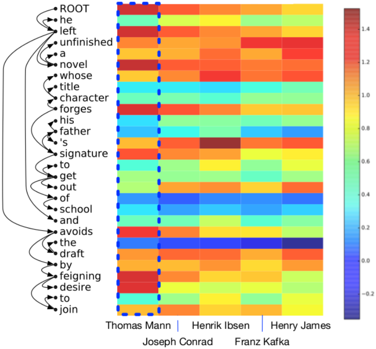


Sneak-peek: QANTA

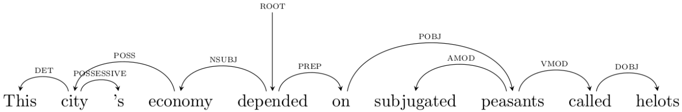
Iyyer et al., 2014: A Neural Network for Factoid Question Answering over Paragraphs

A question answering neural network with trans-sentential averaging.

Later in its existence, this polity's leader was chosen by a group that included three bishops and six laymen, up from the seven who traditionally made the decision. Free imperial cities in this polity included Basel and Speyer. Dissolved in 1806, its key events included the Investiture Controversy and the Golden Bull of 1356. Led by Charles V, Frederick Barbarossa, and Otto I, for 10 points, name this polity, which ruled most of what is now Germany through the Middle Ages and rarely ruled its titular city.



DT-RNN, sentence level distributed representations.



Conclusion

- Practical, open source QA system
- Reasonably documented!
- Paid internship position for this summer!
- Long term:
  - Bleeding edge NLP, IE research
  - Closed domain QA with powerful user interface
  - Commercial application aims

http://ailao.eu/yodaqa

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

Thank you for your attention!

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)

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