YodaQA: A Modular Question Answering System Pipeline

Petr Baudůš
baudipet@fel.cvut.cz

Department of Cybernetics, Czech Technical University, Prague

---

**Goal:** Answer naturally phrased factoid questions, using both structured (e.g. Freebase) and unstructured (e.g. Wikipedia) knowledge bases.

**Question Analysis**
Unstructured user query → narrow text snippet answering the query.

... vs. linked data graph search: requires a precisely structured user query.

... vs. a search engine: returns a whole document or passage.

The Question Answering task is already part of the Google Search interface or personal assistants like Apple Siri, and with the high profile IBM Watson Jeopardy! it has become a benchmark of progress in AI research.

As we are interested in a general purpose QA system, we will consider an "open domain" factoid question answering, rather than domain-specific applications (though we have domain flexibility as one of our goals).

---

**Previous Work**

The most popular approach in QA research has been restricting the task to querying structured knowledge bases, typically using the RDF paradigm and accessible via SPARQL. The problem can then be rephrased as machine translation from free-text user query to a structured query (SPARQL, \(\text{\lambda}e\)-expr).

When relying on unstructured knowledge bases, a common strategy is to offload the information retrieval on an external high-quality web search engine like Google. However, we aim for the sake of domain flexibility and reproducibility of results.

Notable open source systems:

- OpenEpyhta, QAQA, WatsonSim, Jacana, OpenQA.

**Contribution:** A universal framework that allows integration of diverse state-of-art approaches within a common pipeline.

**Keywords:** Question answering, information retrieval, information extraction, linked data, natural language processing, Apache UIMA, software engineering.

---

**The YodaQA Framework**

Paradigm: We are interested in combining different approaches, using different question representations, answer sources and scoring features. Our baseline system is domain flexible and we strongly prefer machine learning-based hand-crafted heuristics.

Platform: Mainly Java, using the Apache UIMA framework and DKpro family of adapters to various NLP tools.

Availability: Publicly available free software under the Apache license at https://github.com/brmson/yodaqa.

---

**The Baseline QA Pipeline**

The basic pipeline flow is inspired by the DeepQA model of IBM Watson. Throughout the flow, answer features are gradually accumulated.

**Question Analysis**

- **Focus:** What was the first book written by Terry Pratchett?
- **LAT:** Lexical Answer Type
- **Where** is Mount Olympus? location

**Clues** (search keywords/phrases):
- POS and constituent token whitelist
- Named entities
- Focus and the NSUBJ constituent
- Concepts: enwiki article titles

**Outcome:** Question representation

---

**Answer Analysis**

- **LAT:** NE type, DBpedia concept type, WordNet relations, numerical
- **Type coercion** of question and answer LATS: Unspecificity is WordNet hypernymy distance

**Answer Production**

- Phrase origin, clue overlaps, LAT kinds, type coercion (\(\Rightarrow\) features)
- Logistic regression scores answers

**Outcome:** Ordered set of Answers

---

**Performance Analysis**

Dataset: 430+430 trivia factoid questions (TREC 2001, 2002 + IRC)

Recall: Whether a correct answer has been generated (with any score)

Accuracy-at-one: Whether the correct answer has been returned as the top answer by the system.

**Future Work**

- Better, larger dataset
- Insightful web interface
- B-I-O answer extraction
- Tree alignment features
- Real-world domains
- Smarter scoring model

---

**Acknowledgements**

This research is supervised by Dr. Jan Šedivý and Dr. Petr Pošik, and supported by the CTU grant GSG14/194/OHK3/3T/13.