Modeling of the Question Answering Task in the YodaQA System

Petr Baudiš and Jan Šedivý

Dept. of Cybernetics, Czech Technical University,
Technická 2, Praha, Czech Republic
baudipet@fel.cvut.cz

Abstract. We briefly survey the current state of art in the field of Question Answering and present the YodaQA system, an open source framework for this task and a baseline pipeline with reasonable performance. We take a holistic approach, reviewing and aiming to integrate many different question answering task definitions and approaches concerning classes of knowledge bases, question representation and answer generation. To ease performance comparisons of general-purpose QA systems, we also propose an effort in building a new reference QA testing corpus which is a curated and extended version of the TREC corpus.

Keywords: Question answering, information retrieval, information extraction, linked data, natural language processing.

1 Introduction

The Question Answering problem (converting an unstructured user query to a specific information snippet) is enjoying renewed research popularity, inspired in part by the high profile Jeopardy! matches of IBM Watson.

The problem is being applied both to open domain (general knowledge; e.g. the QALD challenge) and closed domain (specific knowledge; e.g. the BioASQ challenge). At the same time, the specific task can differ significantly based on the choice of a knowledge base — either a corpora of unstructured data (typically natural language text) or structured database (typically a linked data graph). Finally, when answering questions on top of unstructured data, some argue for yielding answer-bearing passages instead of specific answers.[1] These choices have repercussions on the very formulation of question answering problem, typically require vastly different systems, and lead in different research directions.

In Sec. 2, we review the competing approaches. In Sec. 3, we discuss a related issue of benchmarking question answering systems for comparison and propose a curated dataset initiative. In Sec. 4, we briefly present a system we have created for general question answering that aims to reconcile the competing paradigms. We conclude and outline future research in Sec. 5.

[1] See the **ACL Wiki** topic *Question Answering (State of the art).*

2 Question Answering Approaches

Likely the more popular area of research concerns structured databases, typically linked data, is often posed as a task of machine translation from naturally phrased question to a formal query based on processing the question parse tree [2] [8] or vector embeddings of the question and knowledge base subgraph [3].

Moving to querying an unstructured knowledge base, the problem becomes a mix of information retrieval, information and relation extraction, textual entailment and knowledge representation. In the era of the TREC QA track, systems with large amount of handcraft [10] or wrapping a web search engine [4] dominated (making results difficult to reproduce). The current high performance models for selection of answer-bearing passages relies on alignment of question and passage dependency trees [11] or vector embeddings [18]. The answer extraction can be regarded as a BIO sequence tagging problem alike named entity recognition. [17]

Another proposed task involves recognition of an entity described by a sentence using a TreeRNN-based vector embedding model. [12]

3 Benchmarking

Multiple datasets have been proposed for end-to-end Question Answering performance evaluation on open domain. Perhaps the most popular datasets are the TREC QA track, QALD [15] and WebQuestions [2].

There are many considerations that put a dataset on a scale from easy (single class of questions, clean, without required inference) to realistic (noisy with typos, requiring complex reasoning). Some datasets are highly biased for a particular knowledge base [2], or mix questions with typically entirely independent answering strategies (e.g. yes/no questions, which translate to a textual entailment, with factoid and paraphrasing questions).

To train and benchmark a system for answering factoid questions with answers that can be found in unstructured text corpora, we used the public QA benchmark from the main tasks of the TREC 2001 and 2002 QA tracks[2] with regular expression answer patterns,[3] extended by a set of questions asked to a YodaQA predecessor by internet users via an IRC interface. The dataset was further manually reviewed, questions deemed ambigous or outdated were removed, and the patterns were updated based on current data or Wikipedia phrasing.

The outcome is a dataset of 867 open domain factoid questions, randomly split to 430-question training (and development) and 430-question testing sets.[4]

[2] `http://trec.nist.gov/data/qa/2001_qadata/main_task.html`, or 2002.

[3] Similar datasets from TREC 1999 and TREC 2000 are also available, however are of lower quality and we lacked the resources required to clean them up — TREC 1999 contains large number of corpora-specific questions with many rephrasings, while TREC 2000 contains many paraphrasing questions, which are hard to match.

[4] The outstanding 7 questions are left unused for now.

We release this as a free-standing dataset `factoid-curated` (v1)[5] and invite researchers to use this system for performance measurements. To allow cross-system comparisons, the dataset also includes precise knowledge base versions to use; we offer their archived snapshots for download (and will run query endpoints for a time). We outline further plans for our common dataset initiative in Sec. 5.1.

4 YodaQA Question Answering System

To unite diverse approaches to Question Answering, we propose a new system **YodaQA**, which aims to provide an open source platform that can serve both as scientific research testbed and a practical system. It is composed from largely independent modules, allowing easy extension with better algorithms or novel approaches, while as a fundamental principle all modules share a common pipeline.

4.1 System Architecture

The YodaQA pipeline is implemented mainly in Java, using the Apache UIMA framework. YodaQA represents each artifact as a separate UIMA CAS, allowing easy parallelization and straightforward leverage of pre-existing NLP UIMA components (via the DKPro interface); as a corollary, we compartmentalize different tasks to interchangeable UIMA annotators. Extensive support tooling is included within the package. Detailed technical description of the pipeline is included in a technical report [1].

The system maps an input question to ordered list of answer candidates in a pipeline fashion, with the flow as in Fig. 1 (inspired by the DeepQA model of IBM Watson [6]), encompassing the following stages:[6]

- **Question Analysis** extracts natural language features from the input and produces in-system representations of the question.
- **Answer Production** generates a set of candidate answers based on the question, by performing a **Primary Search** in the knowledge bases according to the question clues and either directly using the results as candidate answers or selecting the relevant passages (the **Passage Extraction**) and generate candidate answers from these (the **Passage Analysis**).
- **Answer Analysis** generates answer features based on detailed analysis (most importantly, lexical type determination and coercion to question type).
- **Answer Merging and Scoring** consolidates the set of answers, removing duplicates and using a machine learned classifier to score answers by their features. Logistic regression is popular. [9]

[5] `https://github.com/brmson/dataset-factoid-curated`

[6] An extra **Successive refining** phase is available, but currently no-op in production.

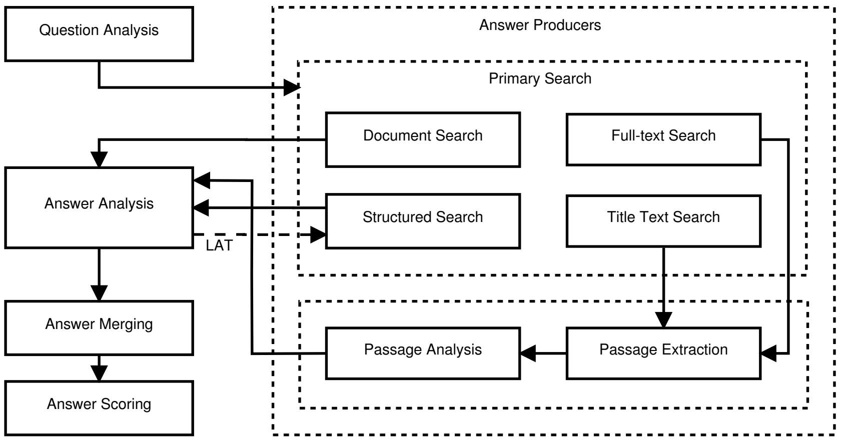

Fig. 1. The general architecture of the YodaQA pipeline. Present but unused final pipeline portions not shown.

4.2 Reference Baseline

The reference pipeline currently considers an English-language task of answering open domain factoid questions (in a style similar to the `factoid-curated` v1 dataset), producing a narrowly phrased answer. While [1] goes into technical details, here we just outline the aspects pertinent to our multi-approach paradigm.

Question Representation: Similar to DeepQA [13], we currently build just a naive representation of the question as bag-of-features. The most important characterization of the question is a set of clues (keywords, keyphrases, and concept clues crisply matching enwiki titles) and possible lexical answer types.

Knowledge Bases: So far, our system is optimized primarily to query unstructured corpora (English Wikipedia, *enwiki*). For informational retrieval, we use Apache Solr[7] and include the document and title-in-clue strategies described in DeepQA [5]. Full-text results are filtered for passages containing the most clues and answers are produced simply from all the named entities and noun phrases.

YodaQA can also query structured corpora (DBpedia, Freebase), so far by a simple baseline query generation approach that just generates an answer for each relation of a concept clue, using relation names as lexical answer types.

Our key design rule is avoidance of hand-crafted rules and heuristics, instead relying just on fully-learned universal mechanisms; we use just about 10 hard-coded rules at this point, mostly in question analysis.

4.3 System Performance

On the test set of the `factoid-curated` v1 dataset, the system achieved **accuracy-at-one of 32.6%**. When we consider all the generated answers, the **recall**

[7] `http://lucene.apache.org/solr/`

System	Accuracy-at-1	Recall	F1	MRR
LLCpass03 [10] (hand-crafted system)	68.5%			
AskMSR [4] (web-search system)	61.4%			0.507
OpenEphyra [14] (hand-crafted OSS)	“above 25%”			
JacanaIR [16] (modern fully-learned OSS)				0.299*
OQA [7] (modern fully-learned OSS)			29%**	
YodaQA v1.0	25.1%	62.2%	35.8%	0.323

Fig. 2. Benchmark results of some relevant systems on the unmodified TREC dataset. * answer-bearing sentence retrieval ** sub-sampled dataset with manual evaluation

is 79.3% (F1 46.2%), **accuracy-at-five is 52.7%** and the correct question **mean reciprocal rank is 0.420**.[8] In Fig. 2, we compare various performance measures with the most relevant previously published systems (on the TREC dataset, as reported in the respective papers), with ours benchmarked on non-curated version of the TREC 2002, 2003 dataset test split.

5 Conclusion and Future Work

We gave a bird eye's view of the Question Answering research landscape and presented the open source platform YodaQA that aims to bring together diverse approaches and allows to benchmark their contributions within a real-world portfolio of methods. We also discussed some common issues with QA datasets and proposed a new one.

While we invested large amount of software engineering effort to the YodaQA pipeline, it is algorithmically still fairly simple. Our work-in-progress efforts include an answer producer that uses a sequence tagging model [17] and extended question representations.

5.1 Benchmarking

By providing a free-standing dataset tracked on Github, we hope to kick-start an effort to build a larger, widely accepted benchmarking dataset. We also work on a web-based platform for crowd-sourcing both questions and correct answers.

One open problem is automatic answer verification, as a correct answer can typically have numerous paraphrases. The current approach of using regex patterns has many caveats (for example numerical quantities with varying formatting and units). While the problem seems ultimately QA-complete, we believe a satisfactory noise reduction could be achieved by specialized matching procedures for some question categories. Some datasets like WebQuestions side-step the issue by posing only questions asking for entity names, but could this bias mis-represent the scientific progress on QA?

[8] The system was configured to take 30s per answer on average without caching (most of it is spent in IR and dependency parsing of passages) on the author's machine; longer-time configurations can further improve the performance.

References

1. Baudiš, P. YodaQA: A Modular Question Answering System Pipeline. In *POSTER 2015 - 19th International Student Conference on Electrical Engineering.*
2. Berant, J., Chou, A., Frostig, R., and Liang, P. Semantic parsing on freebase from question-answer pairs. In *EMNLP* (2013), pp. 1533–1544.
3. Bordes, A., Chopra, S., and Weston, J. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676* (2014).
4. Brill, E., Dumais, S., and Banko, M. An analysis of the AskMSR question-answering system. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10* (2002), Association for Computational Linguistics, pp. 257–264.
5. Chu-Carroll, J., Fan, J., Boguraev, B., Carmel, D., Sheinwald, D., and Welty, C. Finding needles in the haystack: Search and candidate generation. *IBM Journal of Research and Development 56*, 3.4 (2012), 6–1.
6. Epstein, E. A., Schor, M. I., Iyer, B., Lally, A., et al. Making watson fast. *IBM Journal of Research and Development 56*, 3.4 (2012), 15–1.
7. Fader, A., Zettlemoyer, L., and Etzioni, O. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (2014), ACM, pp. 1156–1165.
8. Fader, A., Zettlemoyer, L. S., and Etzioni, O. Paraphrase-driven learning for open question answering. In *ACL (1)* (2013), pp. 1608–1618.
9. Gondek, D., Lally, A., Kalyanpur, A., Murdock, J. W., Duboué, P. A., Zhang, L., et al. A framework for merging and ranking of answers in deepqa. *IBM Journal of Research and Development 56*, 3.4 (2012), 14–1.
10. Harabagiu, S. M., Moldovan, D. I., Clark, C., Bowden, M., Williams, J., and Bensley, J. Answer mining by combining extraction techniques with abductive reasoning. In *TREC* (2003), pp. 375–382.
11. Heilman, M., and Smith, N. A. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL* (2010), Association for Computational Linguistics, pp. 1011–1019.
12. Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., and Daumé III, H. A neural network for factoid question answering over paragraphs. In *Empirical Methods in Natural Language Processing* (2014).
13. Lally, A., Prager, J. M., McCord, M. C., Boguraev, B., Patwardhan, S., Fan, J., Fodor, P., and Chu-Carroll, J. Question analysis: How watson reads a clue. *IBM Journal of Research and Development 56*, 3.4 (2012), 2–1.
14. Schlaefer, N., Gieselmann, P., Schaaf, T., and Waibel, A. A pattern learning approach to question answering within the ephyra framework. In *Text, speech and dialogue* (2006), Springer, pp. 687–694.
15. Unger, C. Multilingual question answering over linked data: Qald-4 dataset, 2014.
16. Yao, X., Van Durme, B., and Clark, P. Automatic coupling of answer extraction and information retrieval. In *ACL (2)* (2013), Citeseer, pp. 159–165.
17. Yao, X., Van Durme, B., et al. Answer extraction as sequence tagging with tree edit distance. In *HLT-NAACL* (2013), pp. 858–867.
18. Yu, L., Hermann, K. M., Blunsom, P., and Pulman, S. Deep Learning for Answer Sentence Selection. In *NIPS Deep Learning Workshop* (Dec. 2014).