
Question Answering

Jason Baldridge

The University of Texas at Austin

Language and Computers, Spring 2008

Questions for question answering

- What is a question?
- What constitutes a valid answer?
- How does one find an answer to a question?
- How do you know when to trust knowledge sources?
- How can we do this automatically?
- Is a web search query a question? Are the pages returned answers?

Question answering via students

- Question for the class: Who directed the Empire Strikes Back?
 - First: open response
 - Second: choose from list

Mark Hamill
George Lucas
Woody Allen
Irvin Kershner
Steven Spielberg

How did the class pick an answer?

- Each student analyzes the question
- Each student matched the query against their own database of knowledge
- Answers were aggregated and filtered
- Was a single answer chosen?

Understanding a question, part 1

- Identify the main predicate (a.k.a. the frame)
- Identify the main participants and their roles
 - ESB is the Patient of the predicate
 - the question is about the Actor
- ESB is a movie (how do we know?)
- The question is about a person
- There are much deeper issues we'll return to

A more formal representation

Find the *person* X , such that
there exists an *event* e , such that

$\text{direct_movie}(e) \ \& \ \text{Actor}(e, X) \ \& \ \text{Patient}(e, \text{ESB}) \ \& \ \text{in_past}(e)$

Automating QA

- People are expensive, and often have better things to do, so we'd like to automate QA
- What resources do we need to do this?
- What kind of information processing do we need to work with such resources?
- What kind of responses will our application return?

Early QA

- Early systems were interfaces to domain-specific databases, such as BASEBALL and LUNAR in the 1960s.
- These are **closed domain** systems: they deal with a specific domain, such as baseball, moon rocks, or medicine.
- This allows them to exploit rich knowledge representations, such as ontologies
- It limits the amount of ambiguity that the system has to deal with.
 - For example, BASEBALL doesn't need to consider whether a bat is a flying rodent: a bat is always a baseball bat for its purposes.

Early QA (cont'd)

- The range of queries closed systems can handle is also limited: this means closed systems are geared toward precision over coverage
- The database for such systems is likely to be fixed: there is no learning.
- We've seen a learning, walking, talking QA system before...

Ask a robot



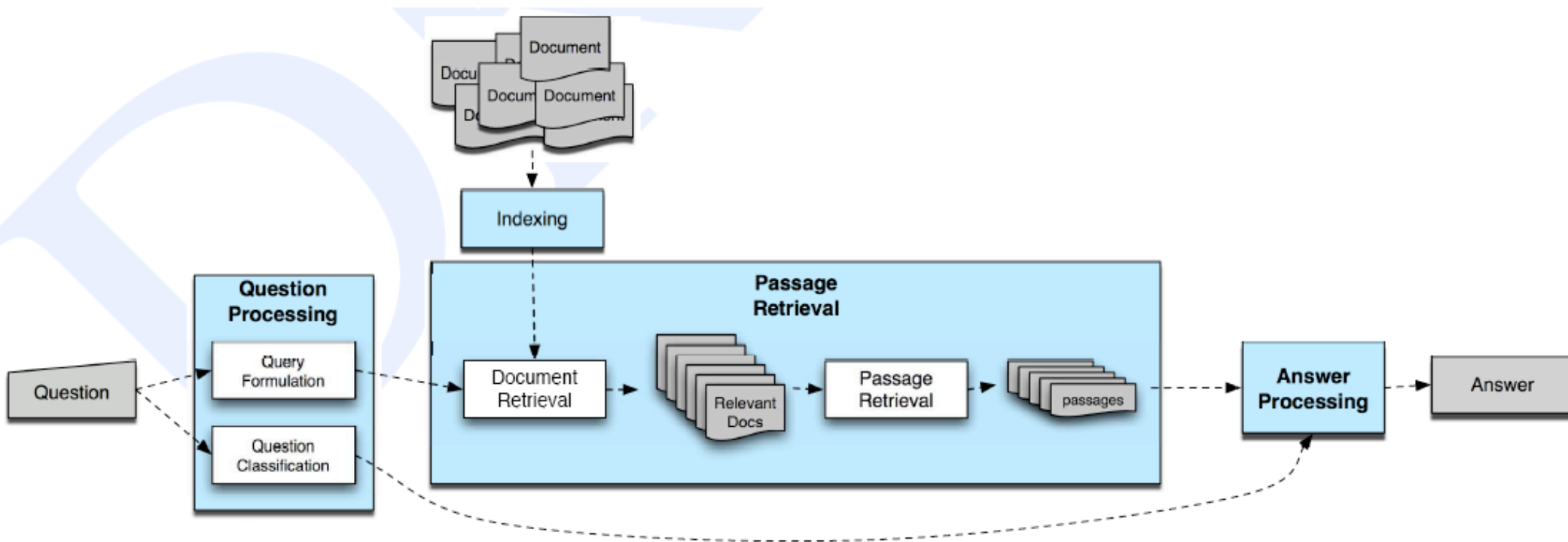
Open-domain systems

- Attempt to answer questions about virtually anything
- Use a much wider range of general knowledge sources
 - often, these sources are unstructured texts which require language processing
- Risk returning irrelevant responses unless they understand the user's query well enough. E.g. "How many records does DJ Shadow have?" should probably be about vinyl LPs, not Olympic swimming records or the number of albums DJ Shadow has created.
- Is web search an example of a QA system? Is a web page an answer?

QA System Architecture for factoids

- Answering a **factoid** question is simpler than full QA: find factual tidbits of information such as names, dates, locations, and quantities.
- Three stages:
 - question processing
 - passage retrieval
 - answer processing
- In many ways similar to standard information retrieval, but with more refined responses returning an exact answer rather than a set of documents.

QA System Architecture for factoids



From Jurafsky and Martin,
Speech and Language Processing, 2nd Edition
<http://www.cs.colorado.edu/~martin/slp2.html>

Query reformulation

- Text often contains answers to queries very directly. Recall can be improved by using **query reformulation** using simple rephrasing and pattern matching. E.g. Given “What is a X?”, look for patterns “X is a Y” in texts and return text matching Y.
 - Regular expression: $\wedge b(.*)\backslash bis\ a\ \backslash b(.*)\backslash b/$ ($\backslash b$ means word boundary)
 - “*wh-word* did *A verb* B” -> “*A verb+ed* B”
 - “Where is X?” -> “X is located in”
- Look at “Who directed the Empire Strikes Back?” on QuALiM, which uses Wikipedia as its knowledge source.

Data redundancy

- The answer might be found in many documents (web-pages) matching different patterns.
 - This eases the burden on the sophistication required for language analysis.
- Frequency: some answers might be wrong (false positives), but the correct one will hopefully be more common.
 - Related to trust in one's knowledge sources: look at QuALiM answers for *Who killed Kennedy?*
 - Show result of *What is the capital of Texas?* for askEd!.

Question classification

- Identify what type of answer is desired.
 - E.g, DEFINITION, PERSON, CITY, ORGANIZATION, BIOGRAPHY
 - Machine learning is used here, similar to what you saw for spam filtering.
- Can include subtypes:
 - HUMAN:DESCRIPTION
 - HUMAN:GROUP
 - HUMAN:INDIVIDUAL

Passage retrieval

- Identify documents that could contain an answer.
- Pull out sentences, paragraphs or sections from each document that contain potential answers.
- Rank these passages using rules or machine learning, e.g.:
 - number of question keywords in the passage
 - N-gram overlap between passage and question
 - number of named entities of the right type (based on question classification)

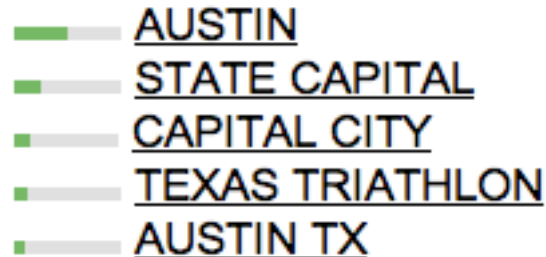
Answer processing

- From the passages, the concise answer must be extracted.
 - *How tall is Mt. Everest? -> The official height of Mount Everest is 29035 feet.*
-> Response: *29035 feet.*
- Use rules or machine learning, again:
 - does the answer match the type sought by the question (e.g., QUANTITY)?
 - novelty: is at least one word in the answer not in the question?
 - use predefined patterns: for YEAR-OF-BIRTH answer, look at patterns like <NAME> (<BD>-<DD>) and <NAME> was born on <BD>

Answer aggregation

- The system is likely to get back many answers. It can use these to try to pick the best answer, or return a ranked set of answers back to the user.
- For example, for the question *What is the capital of Texas?*, askEd! returns the following, where the bars indicate its confidence in each answer:

Answers 1-5



- For description questions, like *Who is Edgar Allen Poe?*, aggregation might be more complex, combining many different pieces of information.



Practical challenges with open-domain QA

- Attempts to manually encode all knowledge in structured format (database style) face a very long tail
- The corpus of documents might not contain the answer.
- The corpus might have a relevant document -- but how do we identify the answer?
- The answer might be in a document in a language other than that of the question.
- Basically, the linguistic processing needs to range from simple **shallow** methods to **deep** language analysis

Linguistic challenges with open-domain QA

- There are different words for specifying a predicate: simple patterns often won't suffice.
 - E.g. "Who is the author of the Star-Spangled Banner?, where the corpus has:
 - "Francis Scott Key wrote the Star-Spangled Banner."
 - "Roseanne Barr sang the Star-Spangled Banner
- Word sense disambiguation. E.g. is "canine" a dog or a tooth? (for a given question)

Linguistic challenges with open-domain QA (cont'd)

- There are multiple ways of referring to an entity, such as with pronouns. This is called **coreference**.
 - E.g. “Who is the author of the Star-Spangled Banner?, where the corpus has: “Francis Scott Key (August 1, 1779 – January 11, 1843) was an American lawyer, author, and amateur poet, from Georgetown. He is the author of the Star-Spangled Banner. The author served as the attorney for Sam Houston... ”
- There many other types, including temporal and bridging references:
 - We WENT to a great restaurant last night. The waiter was very friendly and the food we ATE was delicious.

Linguistic challenges with open-domain QA (cont'd)

- It is hard to recognize **presupposition failure**.
 - *Who is the president of the United Kingdom?* Nobody: the UK has a prime minister, not a president. This requires domain knowledge.
 - *Who was the coach of the Houston Oilers in 2005?* Nobody: the Oilers only existed from 1960-1996. This requires temporal understanding and domain knowledge.
- We may need to reason about the information in the database.
 - *Was Tony Blair in England during the year 2000?* Given: While *Tony Blair* was prime minister of the United Kingdom from 1997 to 2007, he resided at 10 Downing Street.

Shallow and deep analysis

- Shallow: simple, cheap processing
 - uses keywords
 - simple patterns (good for factoids)
- Deep: more sophisticated and intensive processing
 - identify the arguments in a question (and the database) and match on things like Actors and Patients rather than straight text (parsing)
 - handle coreference, word-sense disambiguation, reasoning

Example of deep syntactic analysis

- ISI TextMap syntactic analysis of *Who directed the Empire Strikes Back?*

Who directed the Empire Strikes Back?

Qtargets: **I-EN-PROPER-PERSON&S-PROPER-NAME, I-EN-PROPER-ORGANIZATION (0.5)**

```
[1] Who directed the Empire Strikes Back? [S-SNT]
  (SUBJ) [2] Who [S-INTERR-NP]
    (PRED) [3] Who [S-INTERR-PRON]
  (PRED) [4] directed [S-TR-VERB]
  (OBJ) [5] the Empire Strikes Back [S-NP]
    (DET) [6] the [S-DEF-ART]
    (PRED) [7] Empire Strikes Back [S-PROPER-NAME]
      (PRED) [8] Empire Strikes [S-NP]
        (PRED) [9] Empire Strikes [S-COUNT-NOUN]
          (MOD) [10] Empire [S-NOUN]
            (PRED) [11] Strikes [S-COUNT-NOUN]
              (MOD) [12] Back [S-ADV]
            (DUMMY) [13] ? [D-QUESTION-MARK]
```

Interactive dialog in QA

- It would be useful at times to use dialog to ask and elaborate on questions, e.g.:
 - *What is the drinking age in Scotland?* [Response: 18]
 - *What about other countries?* [Response: USA: 21, France: 16, ...]
- Refining questions
 - When did Bush become President? [Response: Which Bush?]
- This is hard, and we'll get back to it with dialog systems.

Precision and Recall in QA

- Very relevant for QA:
 - there are right and wrong answers and there are questions the system knows nothing about
 - a QA system can abstain from providing an answer; e.g., it says “DUNNO” and passes the question to a (costly) human
- Consider a call center that has a QA system that tries to answer customer’s questions before passing them to a human employee: the goal is to minimize the number of questions answered by humans and provide accurate answers to the ones which the system answers itself.

P/R for detecting answerability

- Let's first consider the QA system's performance for detecting whether or not it can answer a question (i.e., the information is available in its database (DB))
- Let's say that in one hour, the call center receives 154 calls. The QA system thinks that 112 can be answered automatically and passes the 42 (=154-112) remaining ones to humans. Of the 112, the DB only has answers for 91 of them. Of the 42, the DB actually did contain 12 answers.

| | QAS:Yes | QAS:No | |
|--------|---------|--------|-----|
| DB:Yes | 91 | 12 | 103 |
| DB:No | 21 | 30 | 51 |
| | 112 | 42 | 154 |

True Positives: 91

False Negatives: 12

False Positives: 21

True Negatives: 30

Definition of precision and recall

- Precision: of how many you guessed, how many were correct?

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

- Recall: of how many you should have found, how many did you identify?

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Back to our call center

| | QAS:Yes | QAS:No | |
|--------|---------|--------|-----|
| DB:Yes | 91 | 12 | 103 |
| DB:No | 21 | 30 | 51 |
| | 112 | 42 | 154 |

- Precision = $TP/(TP+FP) = 91/(91+21) = 91/112 = .8125 = 81.25\%$
- Recall = $TP/(TP+FN) = 91/(91+12) = 91/103 = .8835 = 88.35\%$

Precision and recall are relative

- What if we were interested in the system's performance as a detector of information which is missing in the database?
- Our positives and negatives are now defined differently.

| | QAS:Yes | QAS:No | |
|--------|---------|--------|-----|
| DB:Yes | 91 | 12 | 103 |
| DB:No | 21 | 30 | 51 |
| | 112 | 42 | 154 |

True Negatives (91)

False Positives (103)

False Negatives (21)

True Positives (51)

P/R as detector of missing information

| | QAS:Yes | QAS:No | |
|--------|---------|--------|-----|
| DB:Yes | 91 | 12 | 103 |
| DB:No | 21 | 30 | 51 |
| | 112 | 42 | 154 |

- Now, TP=30, FP=12, FN=21, and TN=91
- Precision = $TP/(TP+FP) = 30/(30+12) = 30/42 = .7143 = 71.43\%$
- Recall = $TP/(TP+FN) = 30/(30+21) = 30/51 = .5882 = 58.82\%$