Semantic Search

Adding context to basic query term matching, by better understanding/representing features of the query and docs.

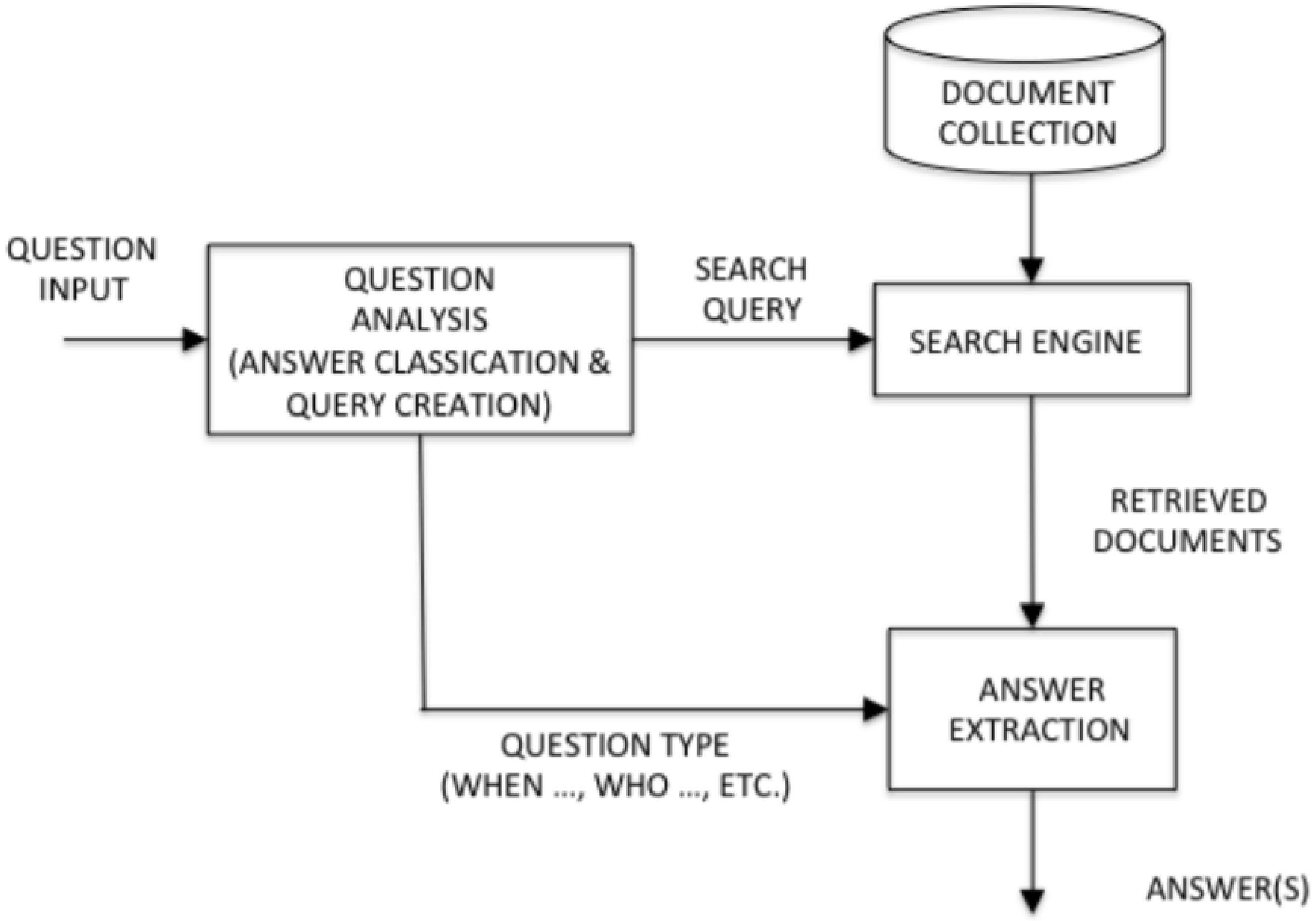
(basic matching relies on the principle of redundancy of info in the doc collection - relevant info needs to be in more than one doc, and at least one has the same vocab as the user query)

Question Answering (QA) systems are a step further than summarisation in terms of reducing user effort. Instead of retrieving potentially relevant docs for the user to read/access, a QA system seeks to provide the answer(s) to the user's question

Differing levels of difficulty..

- simple answers to simple questions eg easily identified name of place, person, company
- simple answers to difficult questions eg where an amount of money must be totalled from different sources
- difficult answers to difficult questions eg composing a list from multiple docs, contrasting info in multiple docs

QA system needs to access a large set of info/knowledge. Use IR system to identify a small number of docs from within a large doc collection which are potentially relevant to a request, and attempt to extract the answer from these docs. Finding the answer will require application of an *answer extraction* component to the retrieved docs



Typical workflow of QA system:

- analyse question and determine type of question
- form search query from the question. Note: query type eg "when" often left out since its not related to identity of potentially useful docs
- enter query into IR system to retrieve a number of docs which are potentially relevant and potentially contain the answer
- enter output of question of analysis module and top ranked retrieved docs int **answer extraction module**
- answer extraction module analyses retrieved docs to try identify the answer(s) to the q

IR component is generally robust and cheap. Answe extraction module typically more expensive to apply. Retrieved list truncated either after a fixed number of docs or based on rate of decrease in matching scores

Tradeoff in precision and recall in deciding length of list

- too short: answers to the question may not be included (recall of answer too low)
- too long: too much irrelevant info may be included increasing computation time and chance of wrong answer being selected

Contrasting approaches to QA

- Knowledge-based: apply formal Natural Language Processing (NLP) and extensive linguistic resources
- Data-based: use of large doc collections with shallow informal language processing and exploitation of info redundancy.

Knowledge Based QA

- Locate examples of *Expected Answer Type* (EAT) in the retrieved docs. Often some kind of named entity.
  - Identify the words in the question that determine the EAT eg *who* (easy) or *what* (more vague).
- examine relationship between words from the question and the possible answers to try identify the answer to the question

NLP methods used here fall into Information Extraction (IE) methods designed to extract *specific pieces of info* from a text. Can fail if NLP methods used or dictionaries are not sufficient to cover the details in the questions/answers

To assist with answering questions, the contents of the available docs can be augmented with knowledge contained in *domain specific data structures* eg simple lists of facts, conventional structured databases or purpose-built knowledge structures such as **taxonomies** and **ontologies**

- **Taxonomies**: a means of capturing info about a particular topic in a hierarchical structure.
  - subcategory of a category inherits properties of its parent
  - motor vehicles --> cars, vans, lorries; cars --> family, sport etc
- **Ontologies** generally include all the info captured in a taxonomy but also details about *rules and relationships* between categories
  - include semantic info used in problem solving and decision making

Data based QA

Used when convoluted statements contain the answer to a question. Needs to be sophisticated and robust enough to handle any way in which the answer may appear in text.

Can reasonably assume that, on the WWW, at least one doc should contain a certain pattern heavily related to some qs. Expect the answer to follow these patterns, but in cases of multiple answers we may need to settle on a best on that is neither over or under specified.

WHen no obvious answer is found in the text, redundancy can oftern provide rasonable "guess" at the correct answer

Evaluation:

Important in development of QA systems. Many metrics

- Correct: % of all qs posed answered correctly
- Precision: % of questions answered where the answer is correct
- Recall: % of questions that ***could*** be answered correctly from the document collection that have been answered correctly
- No answer: % of questions that cannot be answered from the doc collection and have been identified as such.

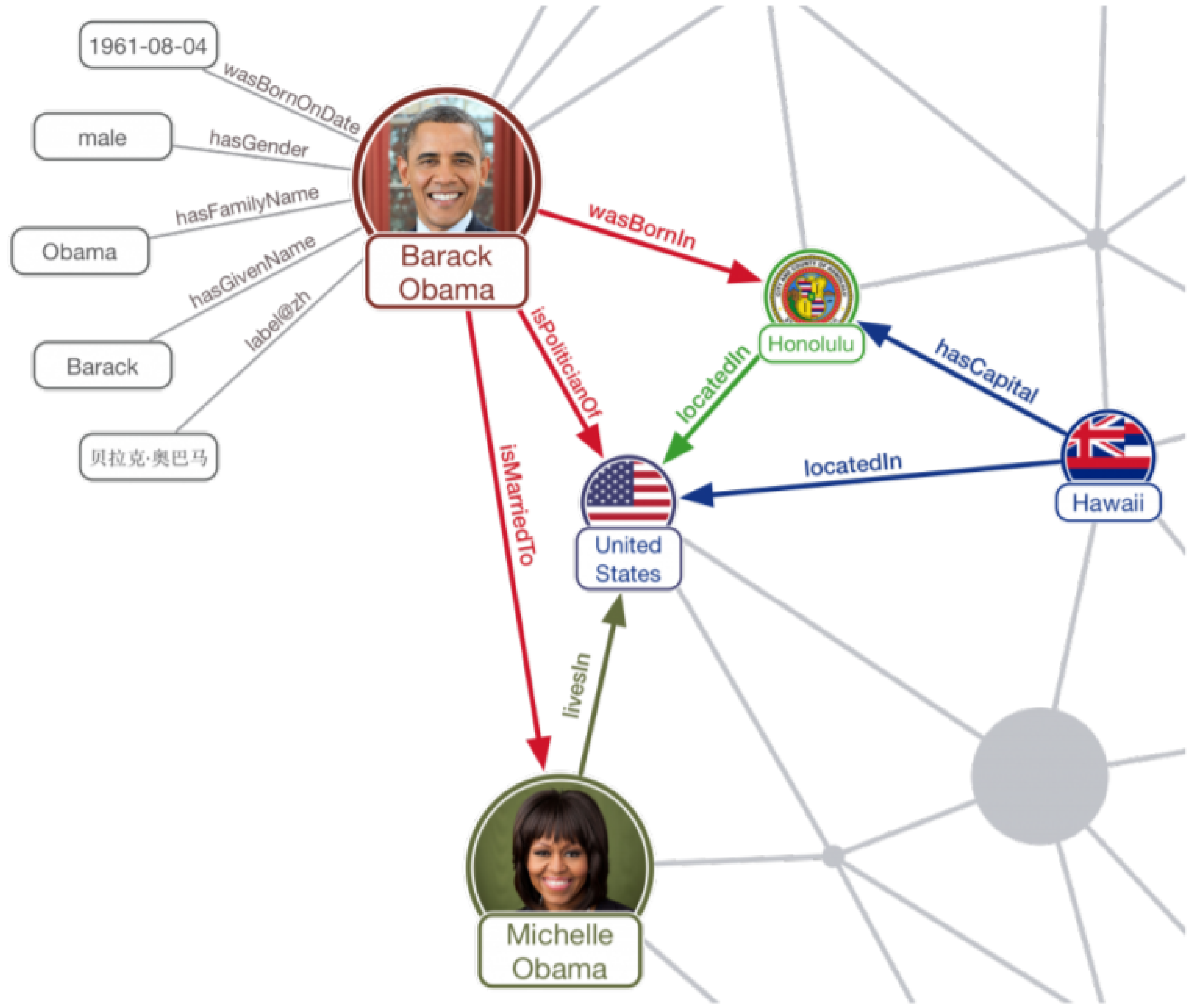
Evaluation needs

- set of docs used to answer questions
- set of questions (don't need to be able to answer all qs from doc set)
- answers to the questions

Knowledge Graph: useful & powerful way to capture interreated info. Aims to encode info from text sources eg web pages, news, books etc in a ormal representation. An *extension of ontologies* which encode info on a topic and its relationships. Underly services provided by Siri, Amazon Echo etc.

Info contained in a knowledge graph is referred to as *structured data*, since it is encoded in a form easily accessed by computers. Focus on

- entities: persons, organisations, locations, products, objects
  - have typical attributes eg people have a first and last name, DOB, parents, occupation etc
- relations: join entities
- facts: combo of entities and relations



- Use **semantic parsing** method to map natural language q into a formal query statement in a form which can be efficiently executed against a knowledge graph.
- attempt to answer the q from the knowledge graph using the formatted question.

Knowledge graphs have a capacity to represent an infinite no of facts never seen in an idividual text, but knowledge graphs can be incolmplete. Relations captured in the knowledge graph may not be expressive enough to represent the answer to a q

Hybrid combos between doc-based question answerin using info retrieval and question answering based on Knowledge Graphs offer

- potential for improved efficiency answering easy questions using KGraph
- more specialist qs using docs

IBM Watson

Developed by IBM as an example of a system which aggregates large amounts of data of multiple types from many different sources to achieve an objective. An example of a

"Grand Challenge" in computing which sets out to promote rapid development in a new area. This was aimed to beat a human champion at the TV Game "Jeoprady!"

A **cognitive computing application**, incorporates the following principles

- **Learn**: use data to make inference about something eg a domain, topic, person etc based on training and observation
- **Model**: create a model of something and dynamically incorporate data into the model as it is observed
- **Generate Hypothesis**: Assume that there is not a single correct answer, the system is probabilistic and uses data to score multiple hypotheses which potentially address a particular problem

Google Knowledge Graph

An extensive knowledge graph based on content harvested from the web

Contains info about and relationships between hundreds of millions of objects or "entities", contains billions of facts

Many search queries focus on entities and Google uses its knowledge graph to enrich the info returned to the searcher for these queries.

Standard SERP including ranked docs is preserved, and augmented with additional info. Use crowdsourcing

Google info Cards

Google use their knowledge graph to construct summary cards about searched for entities. Cards displayed when a significant entity appears in the search query. Google includes facts for each entity which are of most interest for that object.

"People also search for.." on the card, lists related ppl,places etc - related entities that people search for when searching for the current entity. Have "report a problem" option for shen data may be incorrect.

Cards actually encourage more searching, maintain clicks through to websites

Query-doc mismatch

Representing words/phrases in a form tht captures their semantic relationship is a way to tackle mismatch problem

- **Word embedding** methods enable words and phrases to be described in terms of *vectors* representing their shared properties.
- comparing word vectors created using word embedding methods allows thier semantic properties to be compared, potentially overcoming the mismatch problems
- the vector representation for each word has size and direction
- The angle between the vector representations of the word represents semantic similarity between the words
- vector for each word is trained automatically using machine learning with a large corpus of text representative of the data for which the word vectors should be used.
    - elements of the vector for each word are trained using the contexts (words surrounding) the word is observed in the training text. Semantically similar words typically surrounded by the same words and thus have similar vectors

The similaritiy between vectors for words w_i and w_K can be computed using **cosine similarity**

$$\cos\theta = \frac{\mathbf{w_i}.\mathbf{w_k}}{|w_i||w_k|}$$

if term *k* appears in the query and term *i* in a doc, a revised term weight for term *i* in doc *j* could be

$$w(i_{doc}, k_{query}) = w(i, j) \times \cos\theta$$

- the term weight is effectively reduced according to level of similarity between i and k
- This would require some modifications and extentions to the inverted file based matching scheme (see Text Retrieval)

Word embedding and other vector-based representation methods are currently v popular in many areas of natural language processing including

- natural language understanding
- machine translation
- automatic speech recognition

- conversation-based interfaces