CS447: Natural Language Processing
http://courses.engr.illinois.edu/cs447

Lecture 25: Question Answering

Julia Hockenmaier
juliahmr@illinois.edu
3324 Siebel Center

Where are we at?

This week:

Question-Answering (today)

Dialogue (Friday)

Next week:

Large language models (Wednesday and Friday)

Last lecture:

TBD

What is Question Answering?

Question Answering (QA)

Question Answering can mean different things:

— Being able to **query a collection of documents** that is known (or assumed) to contain answers (as short text spans in these documents)

— Being able to answer questions based on a single document by returning short text spans in the document that answer these questions (**“reading comprehension”**)

— Being able to **query a “knowledge base”** (e.g. a database of known facts) in natural language. This may require a **semantic parser** to translate the natural language question into, say, SQL

— Being able to answer knowledge questions about a domain (e.g. take **multiple choice exams** on science questions)

Reading: Chapter 14

Retrieval-Based Factoid QA

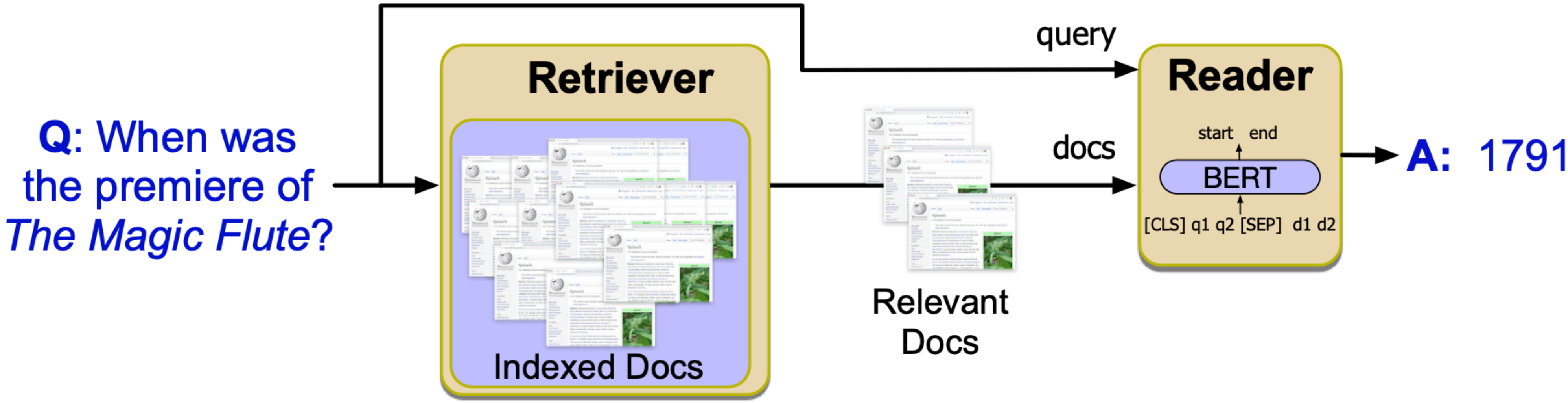
Factoid Questions: QA as IR

Questions about simple facts ("factoids") that are answered by searching a large document collection for **short snippets of texts** that contain the answer

Question	Answer
Where is the Louvre Museum located?	in Paris, France
What's the abbreviation for limited partnership?	L.P.
What are the names of Odin's ravens?	Huginn and Muninn
What currency is used in China?	the yuan
What kind of nuts are used in marzipan?	almonds
What instrument does Max Roach play?	drums
What's the official language of Algeria?	Arabic
How many pounds are there in a stone?	14

This means we can treat QA as an **information retrieval (IR)** task

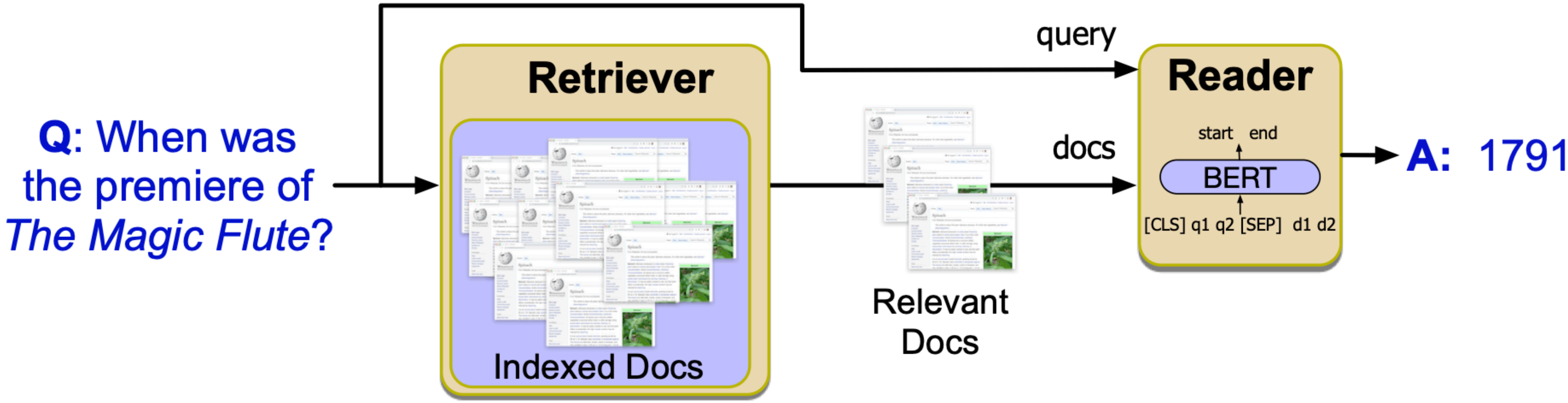
Retrieve-and-Read Pipeline



Assumptions:

- We have access to a **large collection of documents** that we have processed in advanced ("indexed documents")
- The question can be answered by **returning a snippet** of text ("span") from one (or more) of these documents

Retrieve-and-Read Pipeline



Procedure:

- Identify a (small) subset of documents that are **relevant** to the question
- Identify (and return) **the most likely answer span**

Document and Passage Retrieval

The IR engine returns a ranked list of relevant **documents** from the collection.

Because answers are short snippets, the top *n* documents are split into shorter **passages** (e.g. paragraphs).

We can filter passages to identify more relevant passages at this stage, e.g. based on how many named entities they contain, how many question words (or n-grams) they contain, the answer type, etc.

Ad-Hoc Information Retrieval

User poses a natural language **query** to an IR system

(ad-hoc: the query could be about anything, and is not known in advance)

Each query consists of a number of **terms** (tokens or phrases)

The IR system returns a **ranked list** of relevant documents

Documents: web pages, scientific papers, news articles, paragraphs, etc.

Relevance: how similar is the document to the query?

Determining Relevance: tf-idf

Traditional approach to determining relevance:

- Represent **query** q and **document** d as vectors **q**,**d** whose elements correspond to terms t.
- The entry for term t in **q** or **d** depends on its **tf-idf** value
  $q[t] = \text{tf-idf}_{t,q} \quad d[t] = \text{tf-idf}_{t,d}$

tf (term frequency): based on #occurrences of term t in the document d

idf (inverse document frequency): based on #documents that contain term t

$$\textit{tf-idf}_{t,d} = \textit{tf}_{t,d} \times \textit{idf}_{t,d} = \log_{10}\big(\textit{count}(c,d)+1\big) \times \log_{10} \frac{N}{\#d : \textit{count}(t,d) > 0}$$

- **Relevance** of document d for query q: cosine similarity

$$\text{score}(q,d) = \cos(\mathbf{q},\mathbf{d}) = \frac{\mathbf{q}\cdot\mathbf{d}}{|\mathbf{q}||\mathbf{d}|}$$

Determining relevance with LLMs

Instead of using explicit term-based vectors,
we can use BERT (or other large language models)
to compute a document embedding vector for **q** and **d**

Back to QA: Answer Extraction

Given a set of relevant documents/passages, return the span that contains the answer.

Baseline model (for some types of questions)

Run an NER system, and return the entities whose type matches the answer type

More generally, answer extraction can be treated as a **sequence labeling task**

Evaluation: MRR

The **mean reciprocal rank (MRR)** metric is used to evaluate system that return a ranked list of items (here: answer spans):

Q: Where was Elvis born?
Answers:

1. Memphis, Tennessee
2. Tupelo, MS ← Correct (rank(Q) = 2)
3. Graceland

Define rank(Q) as the highest rank of any correct answer for Q, and rRank(Q)=1/rank(Q) when at least one correct answer is returned, and rRank(Q)=0 when no correct answer is returned

The system's MRR score on a pool of *N* questions is then defined as the average (mean) reciprocal rank on all questions

$$\mathrm{MRR} = \frac{1}{N}\mathrm{rRank}(Q_i)$$

Reading comprehension
as span-extraction QA

Reading comprehension tests often ask children to answer questions based on a short paragraph.

Although reading comprehension can be formulated as a **multiple-choice** task, or a **free answer** task (which is difficult to evaluate), the **span-extraction** perspective requires that answers correspond to text spans

SQuAD

Beyoncé Giselle Knowles-Carter (born September 4, 1981) is an American singer, songwriter, record producer and actress. Born and raised in Houston, Texas, she performed in various singing and dancing competitions as a child, and rose to fame in the late 1990s as lead singer of R&B girl-group Destiny's Child. Managed by her father, Mathew Knowles, the group became one of the world's best-selling girl groups of all time. Their hiatus saw the release of Beyoncé's debut album, Dangerously in Love (2003), which established her as a solo artist worldwide, earned five Grammy Awards and featured the Billboard Hot 100 number-one singles "Crazy in Love" and "Baby Boy".
Q: "In what city and state did Beyoncé grow up?" A: "Houston, Texas"
Q: "What areas did Beyoncé compete in when she was growing up?" A: "singing and dancing"
Q: "When did Beyoncé release Dangerously in Love?" A: "2003"

Humans were asked to write questions for Wikipedia paragraphs and provide spans as answers.

The best systems outperform humans, even on SQuAd 2.0, which has "unanswerable questions" (no span to be returned)

Leaderboard: https://rajpurkar.github.io/SQuAD-explorer/

A BiLSTM-based QA System

Basic architecture: Two biLSTMs (for question and passage):

— The **question LSTM** computes a single question **vector q**

— The **passage LSTM** predicts **start and end positions** of the answer span, based on two learned classifiers that depend on each passage word's embedding $\mathbf{p}_i$ and on the question vector **q**

$$P_{\text{start}}(i) \propto \exp(\mathbf{p}_i \mathbf{W}_s \mathbf{q}_i) \qquad P_{\text{end}}(i) \propto \exp(\mathbf{p}_i \mathbf{W}_e \mathbf{q}_i)$$

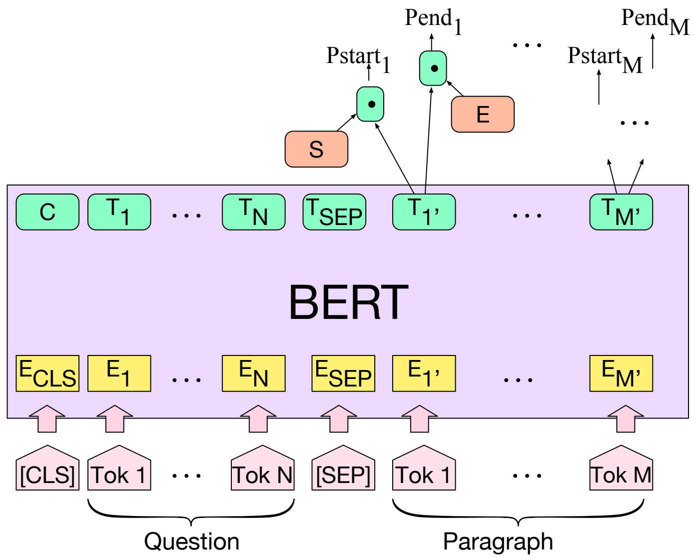
The **question vector q** is a weighted average of the biLSTM-based embeddings of the question words: $\mathbf{q} = \sum_j b_j \mathbf{q_j}$

Question word weights b_j are given by the normalized, exponentiated dot product of each word embedding with a single, learned, relevance weight vector **w**

$$b_j = \exp(\mathbf{w} \cdot \mathbf{q}_j) \big/ \sum_i \exp(\mathbf{w} \cdot \mathbf{q}_i)$$

Passage: Each token is input as an embedding (e.g. GloVe), concatenated with its POS tag/NER label, a 0/1 flag indicating whether it occurs in the question, and possibly an token-specific attention-based embedding of the question

A BERT-based QA system



BERT is a very large pre-trained transformer-based model that provides contextual embeddings

BERT reads the question and passage as a single string, separated by a SEP token.
(this is standard for tasks where BERT has to consider two sequences)

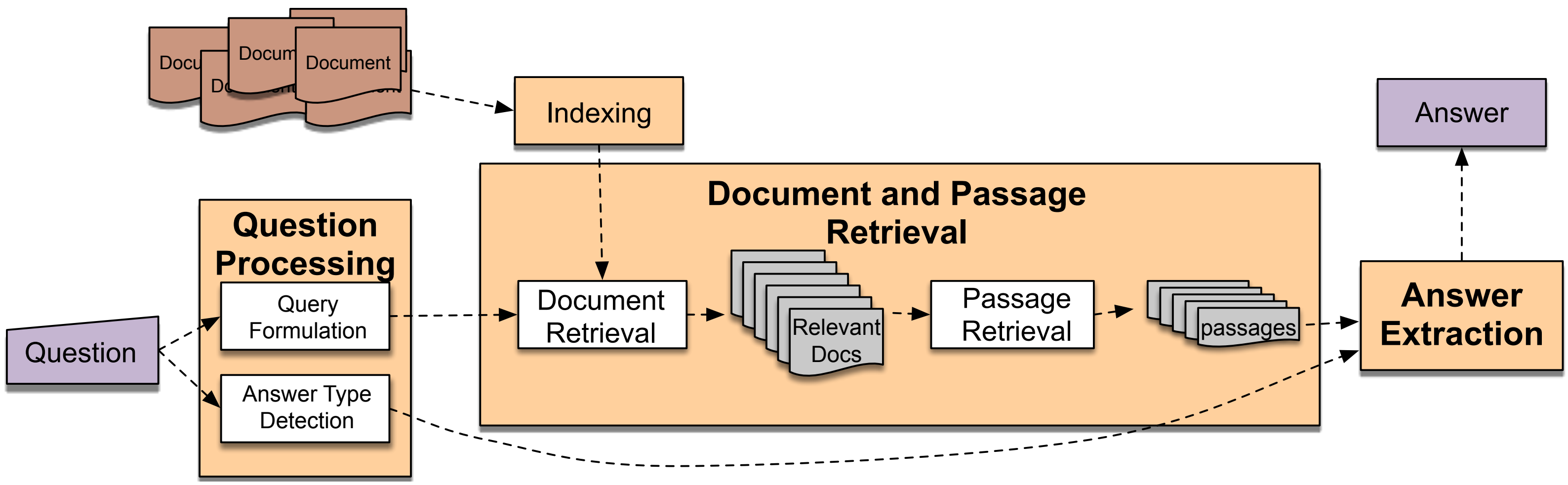
To use BERT for QA:

— define new start and end token embeddings S and E

— fine-tune the output layer, again to predict start and end, e.g. via $P_{\text{start}}(i) \propto \exp(\mathbf{p}_i S)$ $\quad P_{\text{end}}(i) \propto \exp(\mathbf{p}_i E)$

Classical IR-based QA

A simple IR-QA pipeline



Stage 1: Question Processing

— Query Formulation

— Answer Type Detection

Stage 2: Document and Passage Retrieval

— Document Retrieval

— Passage Retrieval

Stage 3: Answer Extraction

Question Processing

We need to get from a **natural language question…**

Which US state capital has the largest population?

…to a **query string** for the IR system:

Query = `"US state capital has largest population"`

… an **answer type:**

Answer Type = **CITY**

… and the **focus** (which words in the question are likely to be replaced by the answer):

Focus = `"which US state capital"`

Answer Type Identification

The answers to many common factoid questions
fall into a small number of categories (**answer types**).

Knowing the answer type can be very helpful.

In the simplest case, the **question word** alone
is sufficient to identify the answer type:

- **Who...** → PERSON
- **Where...** → LOCATION
- **When...** → TIME

But in many cases, one has to consider at least the first noun
after the question word, or the verb

- Which **city...** → CITY
- How much does ... **cost** → MONEY

Answer types (Li & Roth ’02,’05)

Entities:
Animals, body parts, colors, creative works (books, films,…), currency, diseases/medicine, products,…

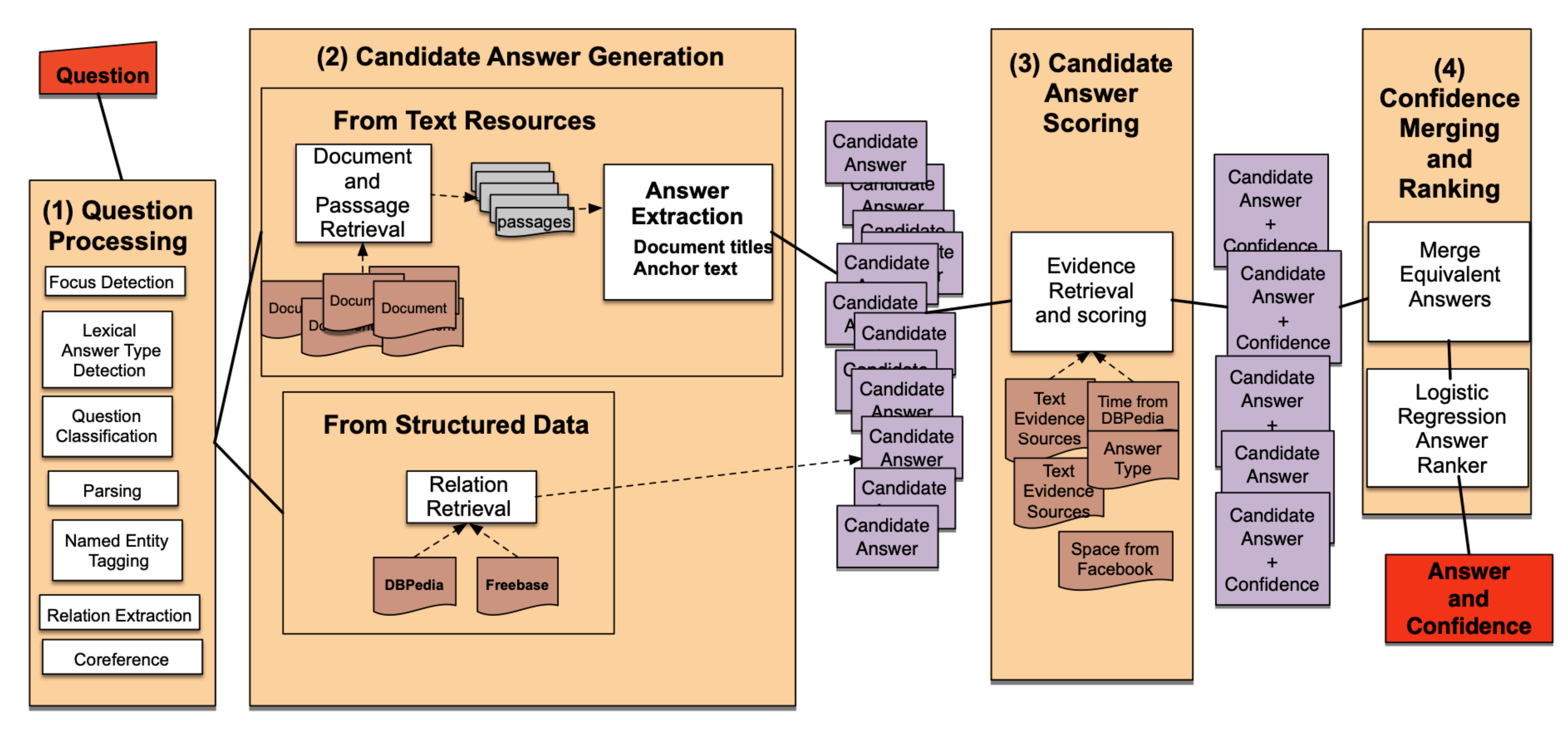
Humans:
Individuals (who was the first person on the moon?), descriptions (who was Confucius?), groups, etc.

Locations:
City, country, mountain, state, …

Descriptions:
Definitions (what is X?), manner (how can you do X),…

Numeric:
Code (e.g. phone numbers), counts, dates, distances, sizes, order (ranks of entities), temperatures, speeds, weights, …

IBM Watson DeepQA



IBM’s Watson wins at Jeopardy!

https://www.youtube.com/watch?v=P18EdAKuC1U

https://dl.acm.org/doi/10.1147/JRD.2012.2184356

Knowledge-Based QA

Knowledge-Based QA

Paradigm 1: Graph-based QA:

Assumes you have a knowledge based of "facts"
(facts = RDF triplets: predicate with two arguments, can also be expressed as a knowledge graph):

```
Ada Lovelace    birth-year   1815
Claude Shannon  birth-year   1916
William Shakespeare author   Hamlet
…                    …          ….
```

[data sets: SimpleQuestions, FreebaseQA, WebQA etc.]

When was Ada Lovelace born?

Who was born in 1815?

…

Knowledge-Based QA

Paradigm 2: QA by semantic parsing

Assumes facts are given in a database.

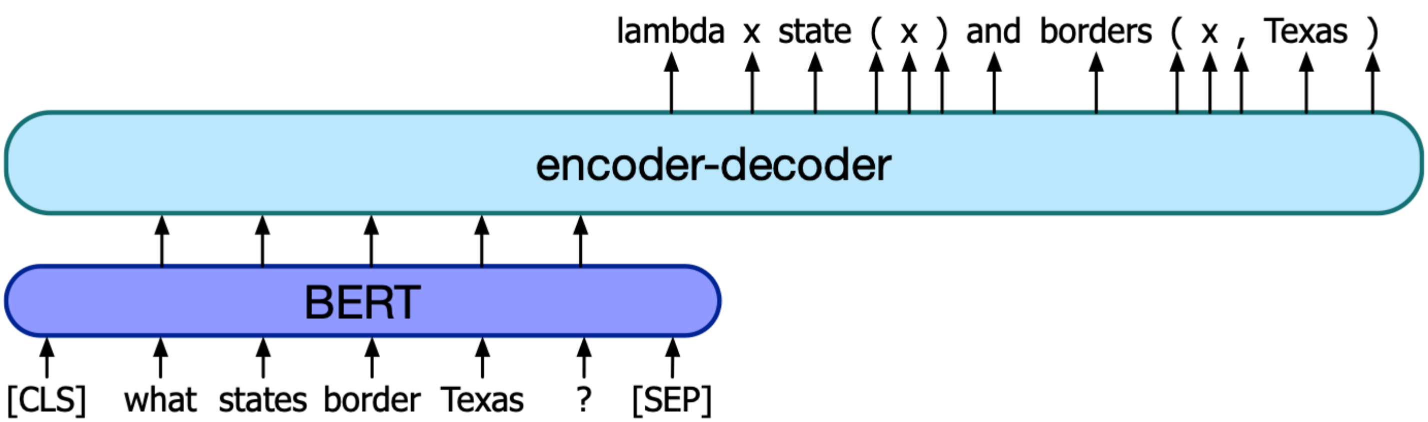
QA can be done by translating query to DB query

Question	Logical form
What states border Texas?	$\lambda x.\text{state}(x) \wedge \text{borders}(x, \text{texas})$
What is the largest state?	$\text{argmax}(\lambda x.\text{state}(x), \lambda x.\text{size}(x))$
I'd like to book a flight from San Diego to Toronto	SELECT DISTINCT f1.flight_id FROM flight f1, airport_service a1, city c1, airport_service a2, city c2 WHERE f1.from_airport=a1.airport_code AND a1.city_code=c1.city_code AND c1.city_name= 'san diego' AND f1.to_airport=a2.airport_code AND a2.city_code=c2.city_code AND c2.city_name= 'toronto'
How many people survived the sinking of the Titanic?	`(count (!fb:event.disaster.survivors fb:en.sinking_of_the_titanic))`
How many yards longer was Johnson's longest touchdown compared to his shortest touchdown of the first quarter?	ARITHMETIC diff(SELECT num(ARGMAX(SELECT)) SELECT num(ARGMIN(FILTER(SELECT))))

Semantic Parsing for QA

Earlier approaches: often grammar-based
(e.g. based on CCG)

Current approaches: seq2seq based models:

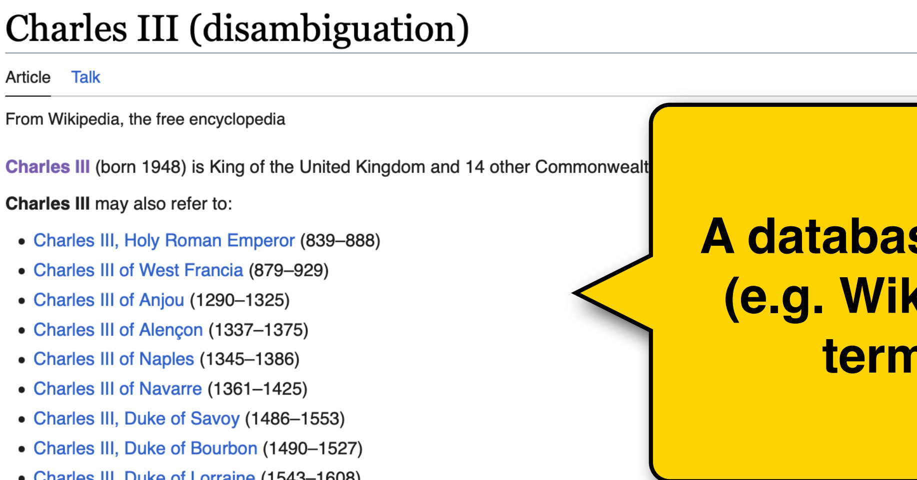


Entity Linking

Map mentions of entities in text to the corresponding entry in an ontology (e.g. based on Wikipedia pages):

Mention detection: which spans are entity mentions?

Mention disambiguation: which entry does a mention refer to?



Charles III (disambiguation)

Article Talk

From Wikipedia, the free encyclopedia

Charles III (born 1948) is King of the United Kingdom and 14 other Commonwealt

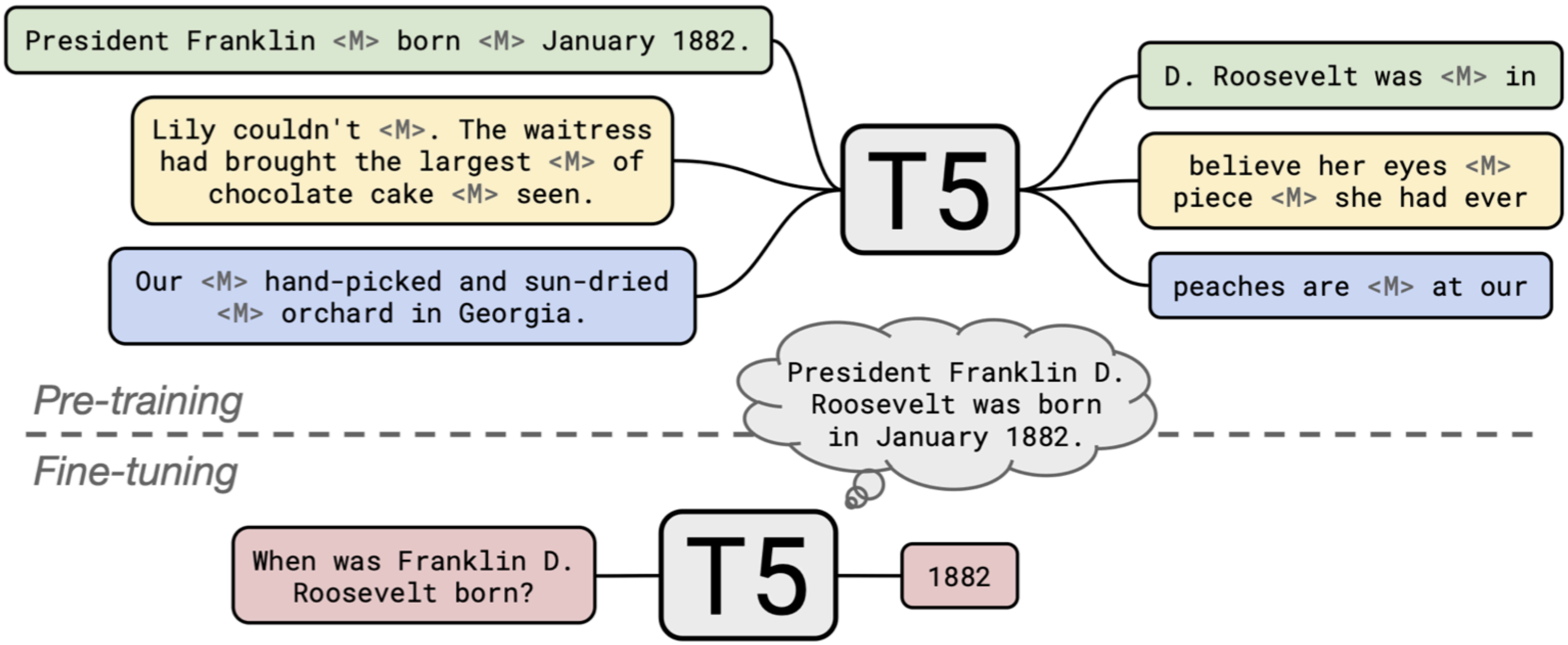
Charles III may also refer to:

- Charles III, Holy Roman Emperor (839–888)
- Charles III of West Francia (879–929)
- Charles III of Anjou (1290–1325)
- Charles III of Alençon (1337–1375)
- Charles III of Naples (1345–1386)
- Charles III of Navarre (1361–1425)
- Charles III, Duke of Savoy (1486–1553)
- Charles III, Duke of Bourbon (1490–1527)
- Charles III, Duke of Lorraine (1543–1608)

A database obtained from Wikipedia (e.g. Wikidata) would use different terms for each Charles III.

This is useful for **knowledge-based QA** (and other applications)

Using LLMs (e.g. T5) for QA



Pre-training: masked language modeling
(fill in masked out word)

Fine-tuning: predict answer (for QA datasets)

Other QA tasks

More recent developments in QA

QA is a very active area of research

Retrieval-based QA is often seen as too simplistic (especially when billed as “reading comprehension”)

More recent developments include datasets whose answers require several steps of reasoning (multi-hop QA), as well answers that require commonsense knowledge.

Visual QA: answer questions about an image.

Science exams as testbed for QA

Task: Answer **multiple choice questions**
from 8th-grade science exams

1. Which equipment will best separate a mixture of iron filings and black pepper?
(1) magnet (2) filter paper (3) triple-beam balance (4) voltmeter

This requires a lot of **background knowledge** that has to be acquired from somewhere (e.g. textbooks), and reasoning capabilities

https://allenai.org/content/docs/Aristo_Milestone.pdf