Question Answering

https://web.stanford.edu/~jurafsky/slp3/23.pdf

Major paradigms

To fill human information needs that arise when talking to a virtual assistant, search engine, or querying a database:

- Information-retrieval based QA
- Knowledge-based QA
- Language model-based QA
- Hybrid approaches

Most QA systems focus on **factoid questions**

Questions that can be answered with simple facts

e.g., Where is the Louvre Museum located? What is the average age of the onset of autism?

Information-retrieval (IR) based QA

Overview

- Also called, Open Domain QA
- Relies on the vast amount of text on the web / collection of text like PubMed
  - Find relevant passages
  - Read the retrieved passages
  - Draw an answer from spans of text

Knowledge-based QA

Overview

- Build a semantic representation of the query
- E.g., What states border Texas?

$$\lambda.x.state(x) \bigwedge borders(x, texas)$$

- Use this meaning representation to query databases of facts

Information-retrieval (IR) based QA

- Most well-known IR system: Search Engine!
- Task: “ad hoc retrieval”
  - User poses a **query** to an IR system
  - IR system returns an ordered set of **documents** from some **collection**

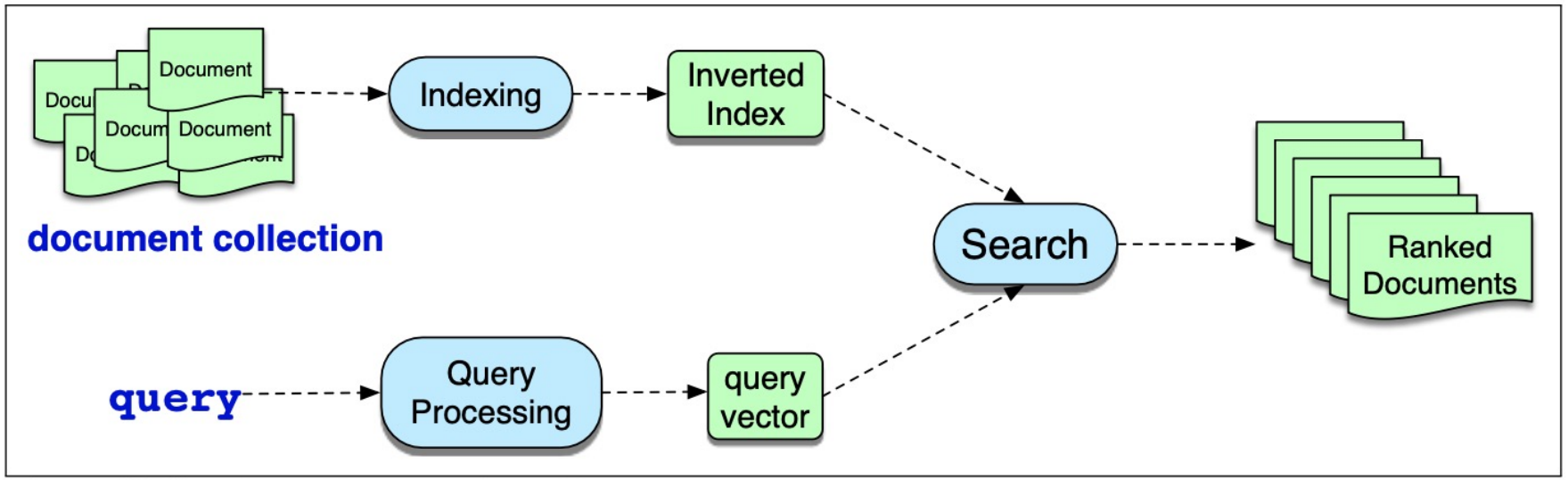


Figure 23.1 The architecture of an ad hoc IR system.

Information-retrieval (IR) based QA

- **Map** queries and document to **vectors**
- Compute **cosine similarities** between the vectors to **rank** documents

Common term-weighting scores for the vectors:

TF-IDF weighting (term frequency, inverse document frequency)

$$\mathrm{tf}_{t,d} = \log_{10}(\mathrm{count}(t,d)+1)$$

$$\mathrm{idf}_t = \log_{10}\frac{N}{\mathrm{df}_t}$$

$$\text{tf-idf}(t,d) = \mathrm{tf}_{t,d}\cdot\mathrm{idf}_t$$

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

Information-retrieval (IR) based QA

- Document scoring

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

$$\text{score}(q,d) = \sum_{t\in\mathbf{q}} \frac{\text{tf-idf}(t,q)}{\sqrt{\sum_{q_i\in q}\text{tf-idf}^2(q_i,q)}} \cdot \frac{\text{tf-idf}(t,d)}{\sqrt{\sum_{d_i\in d}\text{tf-idf}^2(d_i,d)}}$$

$$\text{score}(q,d) = \sum_{t\in q} \frac{\text{tf-idf}(t,d)}{|d|}$$

Information-retrieval (IR) based QA

- Document scoring walkthrough (preprocessing: document is lowercased, punctuation removed)

Query:	sweet love
Doc 1:	Sweet sweet nurse! Love?
Doc 2:	Sweet sorrow
Doc 3:	How sweet is love?
Doc 4:	Nurse!

How about stop words?

Information-retrieval (IR) based QA

- Document scoring walkthrough (preprocessing: document is lowercased, punctuation removed)

Query:	sweet love
Doc 1:	Sweet sweet nurse! Love?
Doc 2:	Sweet sorrow
Doc 3:	How sweet is love?
Doc 4:	Nurse!

	Document 1					Document 2				
word	**count**	**tf**	**df**	**idf**	**tf-idf**	**count**	**tf**	**df**	**idf**	**tf-idf**
love	1	0.301	2	0.301	0.091	0	0	2	0.301	0
sweet	2	0.477	3	0.125	0.060	1	0.301	3	0.125	0.038
sorrow	0	0	1	0.602	0	1	0.301	1	0.602	0.181
how	0	0	1	0.602	0	0	0	1	0.602	0
nurse	1	0.301	2	0.301	0.091	0	0	2	0.301	0
is	0	0	1	0.602	0	0	0	1	0.602	0
$\lvert d_1\rvert = \sqrt{.091^2 + .060^2 + .091^2} = .141$						$\lvert d_2\rvert = \sqrt{.038^2 + .181^2} = .185$				

$$\text{score}(q,d) = \sum_{t \in q} \frac{\text{tf-idf}(t,d)}{|d|}$$

Doc	$\lvert d\rvert$	**tf-idf**(*sweet*)	**tf-idf**(*love*)	**score**
1	.141	.060	.091	1.07
3	.274	.038	.091	0.471
2	.185	.038	0	0.205
4	.090	0	0	0

Information-retrieval (IR) based QA

$$\text{score}(q,d) = \sum_{t \in q} \frac{\text{tf-idf}(t,d)}{|d|}$$

- Motivation: we'll only rank documents that have the query terms
- Inverted index is one such data structure (other methods: bigram indexing, efficient hashing, ...)
  - Given a query term, it will give a list of documents that contain the term
  - 2 parts: dictionary and postings

Query:	sweet love
Doc 1:	Sweet sweet nurse! Love?
Doc 2:	Sweet sorrow
Doc 3:	How sweet is love?
Doc 4:	Nurse!

how {1}	→	3 [1]
is {1}	→	3 [1]
love {2}	→	1 [1] → 3 [1]
nurse {2}	→	1 [1] → 4 [1]
sorry {1}	→	2 [1]
sweet {3}	→	1 [2] → 2 [1] → 3 [1]

Information-retrieval (IR) based QA

Evaluation

- Precision: fraction of returned documents that are relevant
- Recall: fraction of all relevant documents that are returned

Need to compare if one IR system ranks the relevant documents *higher*

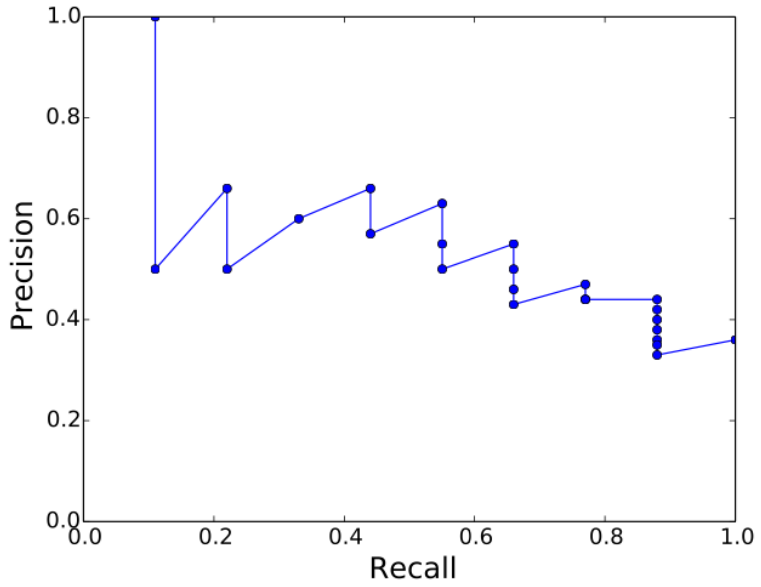
- $\text{Precision}_{\text{rank}}$ and $\text{Recall}_{\text{rank}}$

Information-retrieval (IR) based QA

Evaluation

Need to compare if one IR system ranks the relevant documents *higher*

- $\text{Precision}_{\text{rank}}$ and $\text{Recall}_{\text{rank}}$



Rank	Judgment	Precision_{Rank}	Recall_{Rank}
1	R	1.0	.11
2	N	.50	.11
3	R	.66	.22
4	N	.50	.22
5	R	.60	.33
6	R	.66	.44
7	N	.57	.44
8	R	.63	.55
9	N	.55	.55
10	N	.50	.55
11	R	.55	.66
12	N	.50	.66
13	N	.46	.66
14	N	.43	.66
15	R	.47	.77
16	N	.44	.77
17	N	.44	.77
18	R	.44	.88
19	N	.42	.88
20	N	.40	.88
21	N	.38	.88
22	N	.36	.88
23	N	.35	.88
24	N	.33	.88
25	R	.36	1.0

Figure 23.4 Rank-specific precision and recall values calculated as we proceed down through a set of ranked documents (assuming the collection has 9 relevant documents).

Information-retrieval (IR) based QA

Need to compare if one IR system ranks the relevant documents *higher*

- $\text{Precision}_{\text{rank}}$ and $\text{Recall}_{\text{rank}}$
- For multiple queries, can use **Interpolated precision**

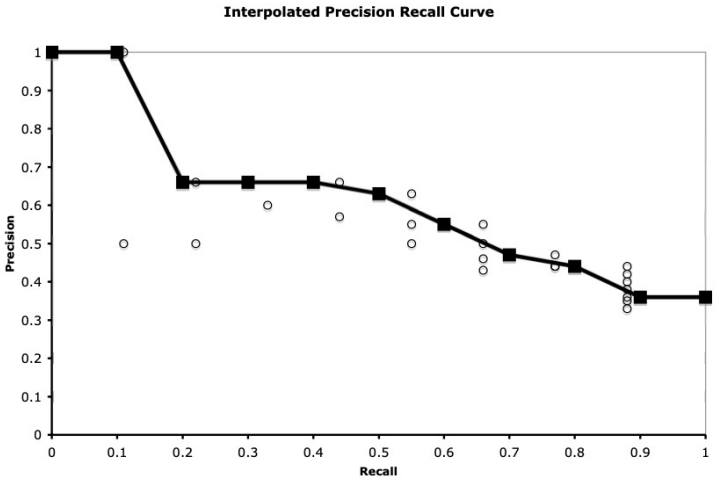
$$\text{IntPrecision}(r) = \max_{i>=r} \text{Precision}(i)$$

- Also smooth out the precision-recall curve
- A better IR system is one with higher precision across recall values

Precision_{Rank}	Recall_{Rank}
1.0	.11
.50	.11
.66	.22
.50	.22
.60	.33
.66	.44
.57	.44
.63	.55
.55	.55
.50	.55
.55	.66
.50	.66
.46	.66
.43	.66
.47	.77
.44	.77
.44	.77
.44	.88
.42	.88
.40	.88
.38	.88
.36	.88
.35	.88
.33	.88
.36	1.0

Interpolated Precision	Recall
1.0	0.0
1.0	.10
.66	.20
.66	.30
.66	.40
.63	.50
.55	.60
.47	.70
.44	.80
.36	.90
.36	1.0

Interpolated data points from Fig. 23.4.



Information-retrieval (IR) based QA

Evaluation

Average Precision (AP) for a single query:

$$\mathrm{AP} = \frac{1}{|R_r|} \sum_{d \in R_r} \mathrm{Precision}_r(d)$$

Mean average precision (MAP) for multiple queries Q:

$$\mathrm{MAP} = \frac{1}{|Q|} \sum_{q \in Q} \mathrm{AP}(q)$$

Rank	Judgment	$\mathrm{Precision}_{Rank}$	Recall_{Rank}
1	R	1.0	.11
2	N	.50	.11
3	R	.66	.22
4	N	.50	.22
5	R	.60	.33
6	R	.66	.44
7	N	.57	.44
8	R	.63	.55
9	N	.55	.55
10	N	.50	.55
11	R	.55	.66
12	N	.50	.66
13	N	.46	.66
14	N	.43	.66
15	R	.47	.77
16	N	.44	.77
17	N	.44	.77
18	R	.44	.88
19	N	.42	.88
20	N	.40	.88
21	N	.38	.88
22	N	.36	.88
23	N	.35	.88
24	N	.33	.88
25	R	.36	1.0

Information-retrieval (IR) based QA

- Sparse vectors (TF-IDF) work only if there is exact overlap of words between query and document
  - E.g., user search for “tragic love story” but Shakespeare writes instead about “star-crossed lovers”
- Instead of sparse vectors, use dense embeddings
  - E.g., BERT

$$h_q = \text{BERT}_Q(\text{q})[\text{CLS}]$$
$$h_d = \text{BERT}_D(\text{d})[\text{CLS}]$$
$$\text{score}(d,q) = h_q \cdot h_d$$

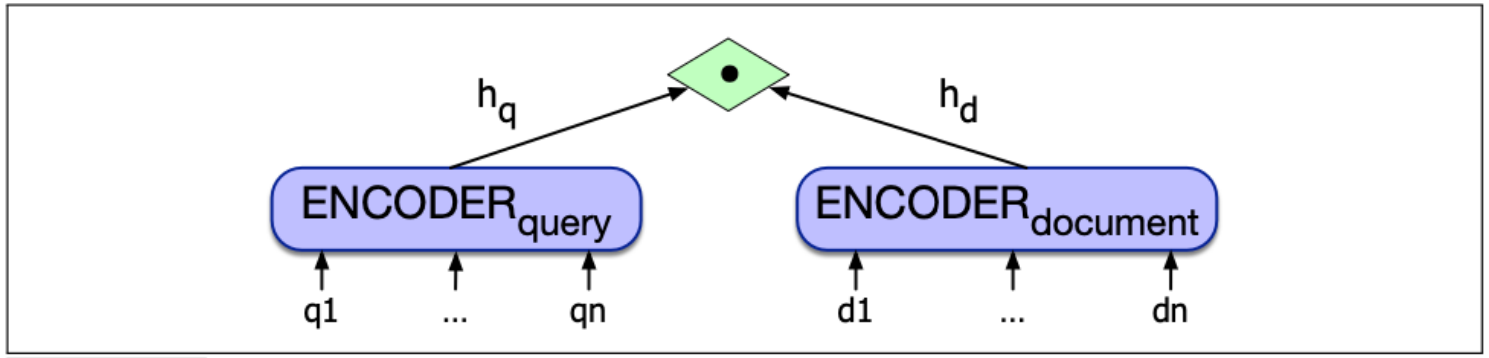


Figure 23.8 BERT bi-encoder for computing relevance of a document to a query.

Or average pooling, etc. instead of CLS; how to do fine-tuning of these encoders, ... (Open research area)

Information-retrieval (IR) based QA

- What we saw so far:
  - **Retrieval** process
- Next stage is to do the **reading** or **neural comprehension** process
  - Find spans of text that are likely to answer the question (**reading comprehension** task)

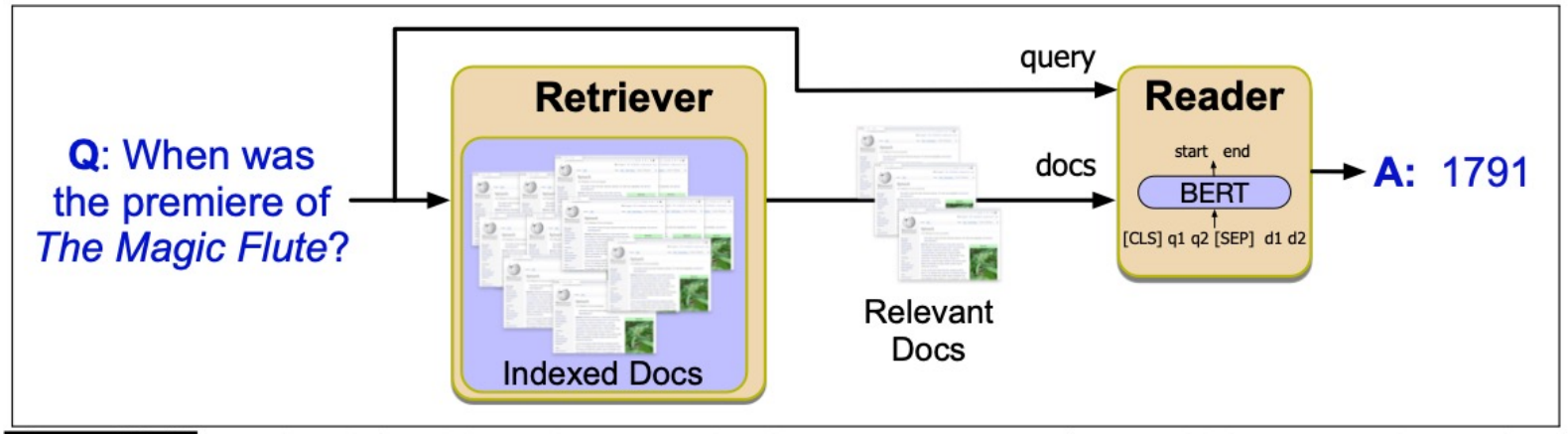


Figure 23.10 IR-based factoid question answering has two stages: **retrieval**, which returns relevant documents from the collection, and **reading**, in which a neural reading comprehension system extracts answer spans.

Open Domain QA
-- retrieve
-- read

Information-retrieval (IR) based QA

- Reading comprehension datasets
  - Tuples of (*passage*, *question*, *answer*)
  - Train a model so that given a *passage* and a *question*, it predicts a **span** in the *passage* as the *answer*
  - SQUAD: Stanford Question Answering Dataset
    - Passages from Wikipedia and associated questions whose answers are spans in the passage
    - Add some questions that are unanswerable
    - 150K questions
  - HotpotQA dataset
    - Multiple documents are shown to annotators are context and they are asked to come up with questions that require **reasoning** about **all** of the documents

Information-retrieval (IR) based QA

- Reading comprehension datasets
  - 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"

Figure 23.11 A (Wikipedia) passage from the SQuAD 2.0 dataset (Rajpurkar et al., 2018) with 3 sample questions and the labeled answer spans.

POTENTIAL BIAS:
Annotators read the *passage* before writing the *questions*, so they may use some **words** from the *answer* itself in their *questions*

So: TriviaQA dataset, Natural Questions dataset, ...

Most of the dataset is also in English! So: TyDi QA

Information-retrieval (IR) based QA

- Reading comprehension task
  - ”Extractive QA”: the answer is a span of text in the passage
  - Span labeling: identifying in the passage a span (a continuous string of text) that constitutes an answer.

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"

Figure 23.11 A (Wikipedia) passage from the SQuAD 2.0 dataset (Rajpurkar et al., 2018) with 3 sample questions and the labeled answer spans.

Information-retrieval (IR) based QA

- Reading comprehension task
  - ”Extractive QA”: Span labeling
  - Given a question *q* of *n* tokens: q_1, ..., q_n and a passage *p* of *m* tokens p_1, ... p_m, goal is to compute P(*a*|*q*, *p*) that each possible span *a* is the answer
  - Simplifying:
    - For each token p_i in the passage, compute two probabilities:
    - $p_{start}(i)$: the probability that p_i is the start of the answer span
    - $p_{end}(i)$: the probability that p_i is the end of the answer span

Information-retrieval (IR) based QA

- Extractive QA
  - For each token p_i in the passage, compute two probabilities:
  - $p_{start}(i)$: the probability that p_i is the start of the answer span
  - $p_{end}(i)$: the probability that p_i is the end of the answer span

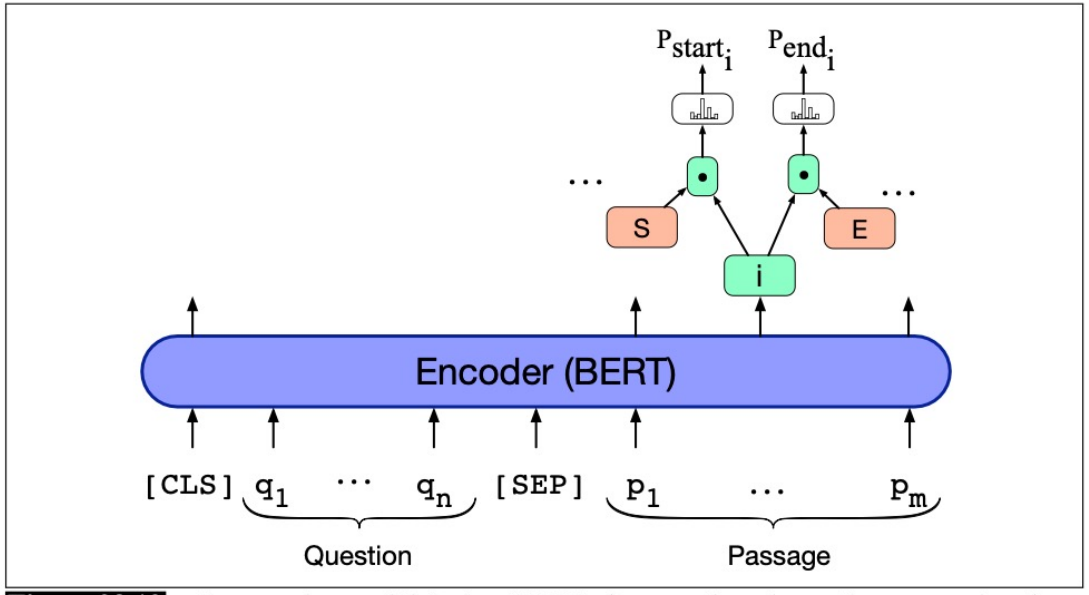


Figure 23.12 An encoder model (using BERT) for span-based question answering from reading-comprehension-based question answering tasks.

$$P_{\text{start}_i} = \frac{\exp(S \cdot p'_i)}{\sum_j \exp(S \cdot p'_j)}$$

$$P_{\text{end}_i} = \frac{\exp(E \cdot p'_i)}{\sum_j \exp(E \cdot p'_j)}$$

$$L = -\log P_{\text{start}_i} - \log P_{\text{end}_i}$$

Knowledge-base QA

But before,

- Entity Linking
  - Task of associating a mention in text with the representation of some real-world entity in an ontology
  - If that ontology is Wikipedia (each wiki page is an entity), this task is also known as “Wikification”
- Roughly two stages:
  - Mention detection
  - Mention disambiguation

Entity Linking

- ELQ algorithm

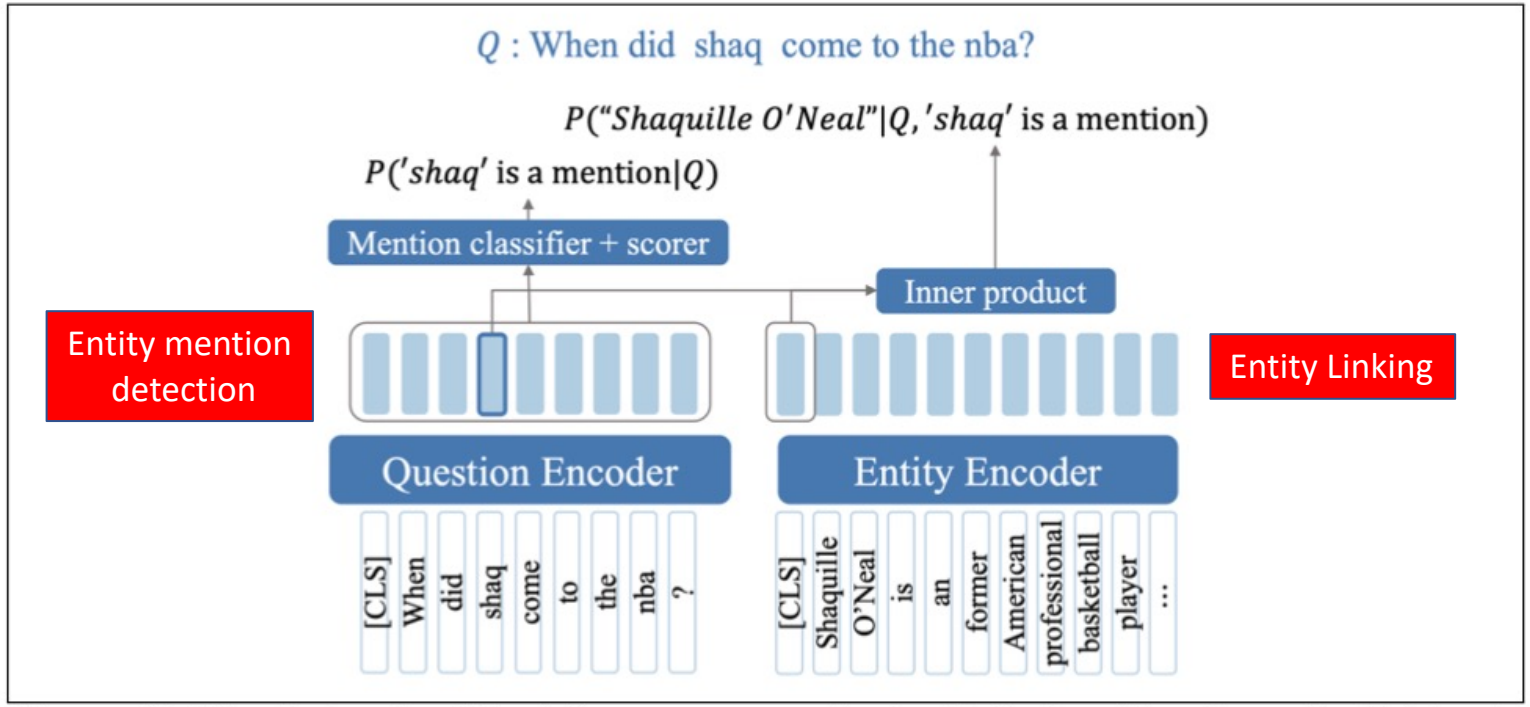
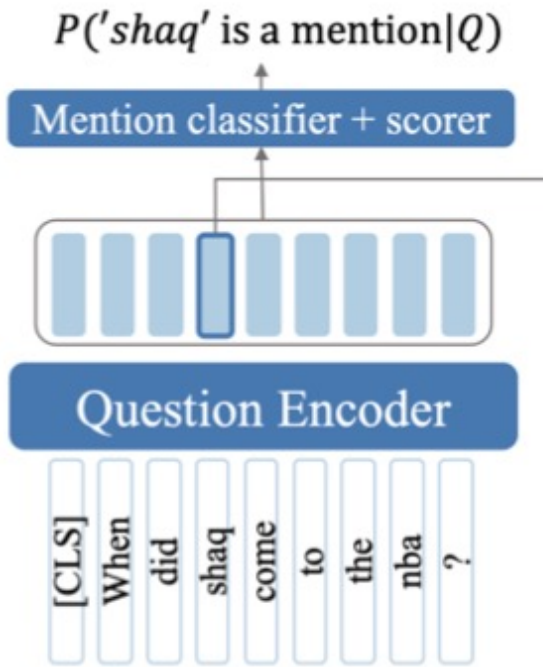


Figure 23.13 A sketch of the inference process in the ELQ algorithm for entity linking in questions (Li et al., 2020). Each candidate question mention span and candidate entity are separately encoded, and then scored by the entity/span dot product.

Entity Linking

- ELQ algorithm: entity mention detection



$$[\mathbf{q}_1 \cdots \mathbf{q}_n] = \text{BERT}([\text{CLS}] q_1 \cdots q_n [\text{SEP}])$$

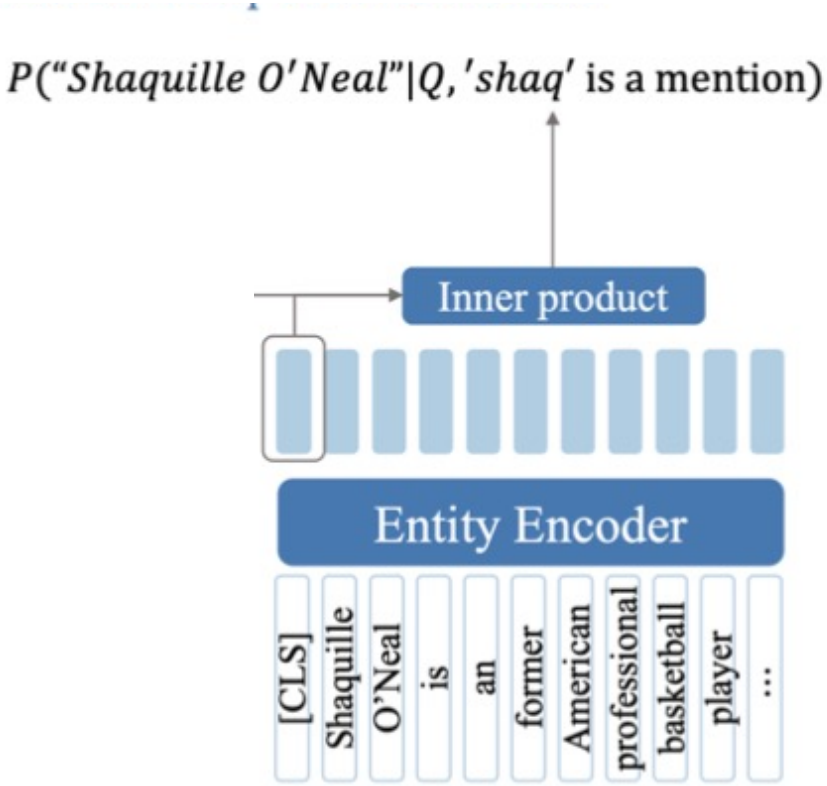
$$s_{\text{start}}(i) = \mathbf{w}_{\text{start}} \cdot \mathbf{q}_i, \quad s_{\text{end}}(j) = \mathbf{w}_{\text{end}} \cdot \mathbf{q}_j,$$

$$s_{\text{mention}}(t) = \mathbf{w}_{\text{mention}} \cdot \mathbf{q}_t$$

$$p([i,j]) = \sigma\left(s_{\text{start}}(i) + s_{\text{end}}(j) + \sum_{t=i}^{j} s_{\text{mention}}(t)\right)$$

Entity Linking

- ELQ algorithm: entity linking



For each entity,

- Get the title and the first 128 tokens of the Wikipedia page

$$\mathbf{x}_e = \text{BERT}_{[\text{CLS}]}([\text{CLS}]t(e_i)[\text{ENT}]d(e_i)[\text{SEP}])$$

- Probability that a span [i, j] corresponds to an entity e, P(e|[i,j])

$$\mathbf{y}_{i,j} = \frac{1}{(j-i+1)} \sum_{t-i}^{j} \mathbf{q}_t$$

$$s(e,[i,j]) = \mathbf{x}_e \mathbf{y}_{i,j}$$

$$p(e|[i,j]) = \frac{\exp(s(e,[i,j]))}{\sum_{e' \in \mathcal{E}} \exp(s(e',[i,j]))}$$

Entity Linking

- ELQ algorithm training
  - Train the entity mention detection and entity linking jointly
  - Optimize sum of their losses:

$$\mathcal{L}_{\text{MD}} = -\frac{1}{N} \sum_{1 \le i \le j \le min(i+L-1,n)} \left(y_{[i,j]} \log p([i,j]) + (1 - y_{[i,j]}) \log(1 - p([i,j])) \right)$$

with $y_{[i,j]} = 1$ if $[i, j]$ is a gold mention span, else 0.

$$\mathcal{L}_{\text{ED}} = -log p(e_g | [i,j])$$

where e_g is the gold entity for mention $[i, j]$.

Knowledge base QA

- Two common paradigms:
  - Graph-based QA
    - What we've seen before: entities as nodes, relations as edges
  - QA by semantic parsing
    - Use semantic parsing methods
  - Both require entity linking

Knowledge base QA

- Graph-based QA
  - Given RDF triples in knowledge base (subject, predicate, object), we can use these triples to answer questions

subject	predicate	object
Ada Lovelace	birth-year	1815

Can answer questions like “when was Ada Lovelace born?” or “who was born in 1815?”

e.g., FreebaseQA dataset, SimpleQuestions dataset, WebQuestions dataset, ...

Knowledge base QA

- Graph-based QA
  - Entity linking
  - Relation detection and linking

"When was Ada Lovelace born?" → `birth-year (Ada Lovelace, ?x)`

$$\mathbf{m}_r = \text{BERT}_{\text{CLS}}([\text{CLS}]q_1 \cdots q_n[\text{SEP}])$$

$$s(\mathbf{m_r}, r_i) = \mathbf{m_r} \cdot \mathbf{w_{r_i}}$$

$$p(r_i|q_1, \cdots, q_n) = \frac{\exp(s(\mathbf{m_r}, r_i))}{\sum_{k=1} N_R \exp(s(\mathbf{m_r}, r_k))}$$

Separate vector trained for each relation r_i

Knowledge base QA

- QA by semantic parsing
  - Use semantic parser to map the question to a structured program
  - Program is then run to produce answer
  - Trained on tuples consisting of questions and their logical forms: GeoQuery dataset, DROP dataset, ATIS dataset (flight queries → SQL forms)

Knowledge base QA

- QA by semantic parsing

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))))

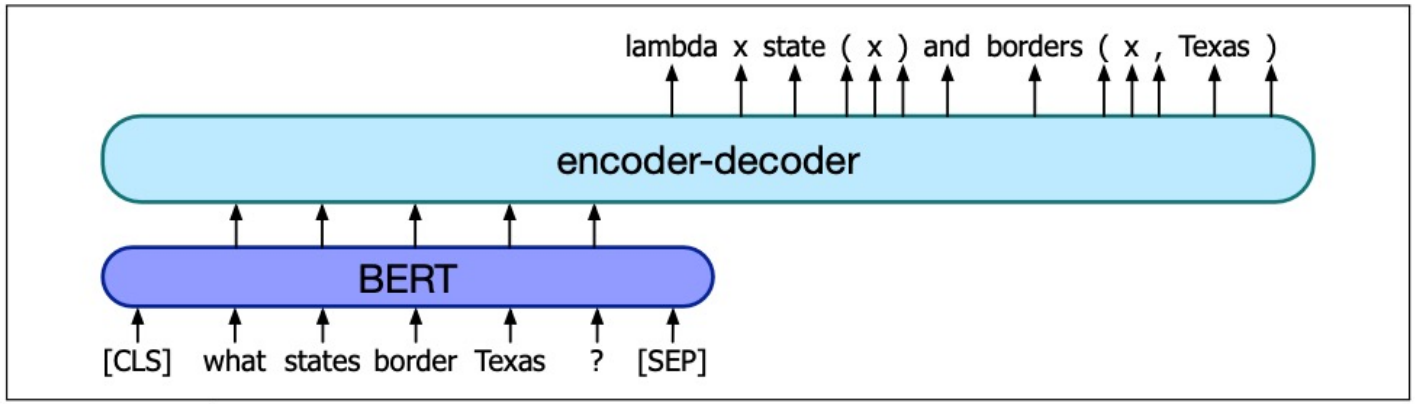


Figure 23.15 An encoder-decoder semantic parser for translating a question to logical form, with a BERT pre-encoder followed by an encoder-decoder (biLSTM or Transformer).

Other QA approaches

- Using Language Model for QA
- Classical QA approach

Language Model – based QA

- Query a pre-trained language model (LM)
  - Force the LM to answer a question based on information stored in its parameters
  - T5 LM: encoder-decoder based architecture
    - For QA, fine-tune by giving it a question and training it to output the answer in the decoder
    - But does not perform as well as systems designed specifically for QA, poor interpretability, no evidence given for answers, ...

Language Model – based QA

- Query a pre-trained language model (LM)
  - T5 LM: encoder-decoder based architecture

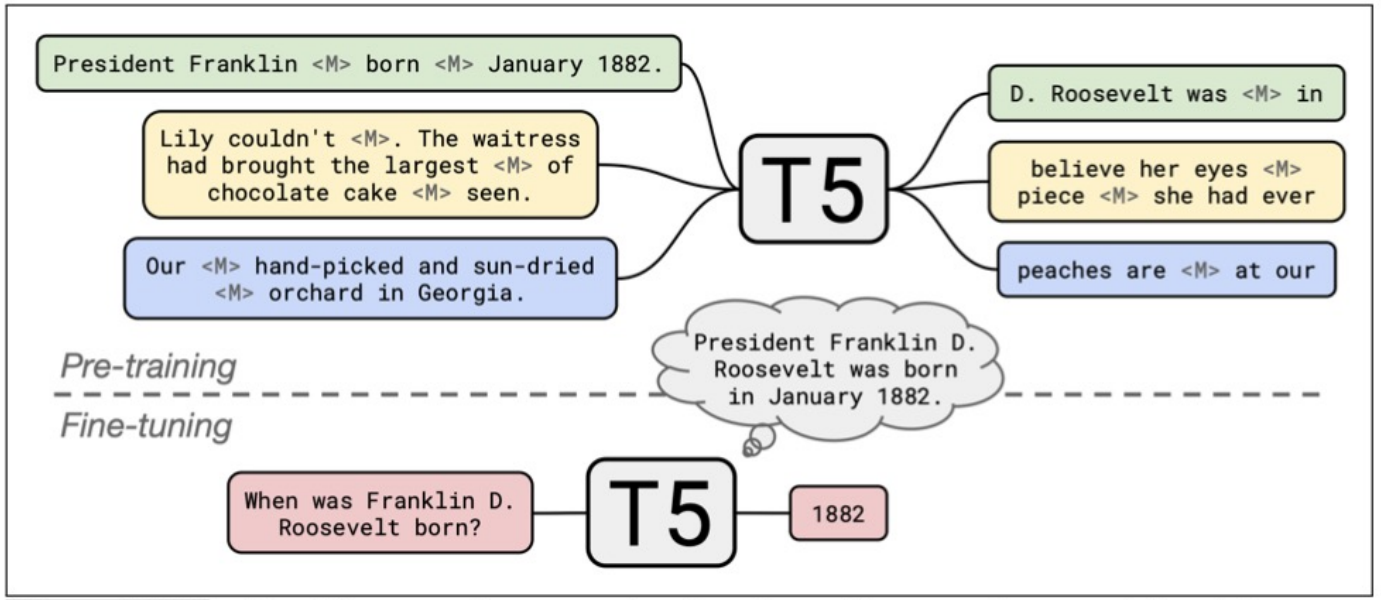


Figure 23.16 The T5 system is an encoder-decoder architecture. In pretraining, it learns to fill in masked spans of task (marked by <M>) by generating the missing spans (separated by <M>) in the decoder. It is then fine-tuned on QA datasets, given the question, without adding any additional context or passages. Figure from Roberts et al. (2020).

Classic QA model

- Watson DeepQA system that won Jeopardy! In 2011

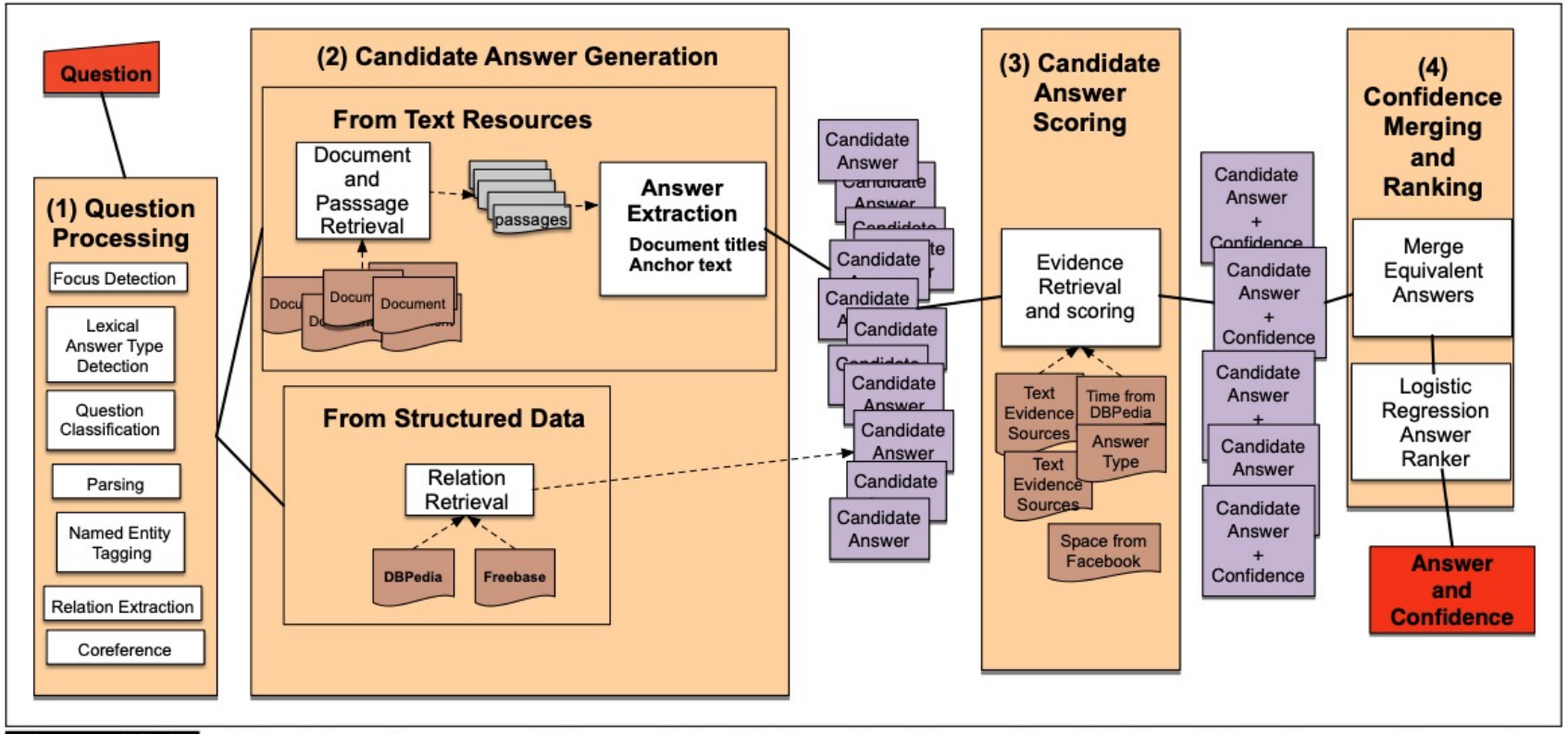


Figure 23.17 The 4 broad stages of Watson QA: (1) Question Processing, (2) Candidate Answer Generation, (3) Candidate Answer Scoring, and (4) Answer Merging and Confidence Scoring.

Evaluation of QA models

- Mean reciprocal rank
  - For systems that return a short **ranked** list of answers or passages for each test question
  - The **reciprocal** of the **rank** of the **first correct answer**
  - MRR is **the average of the scores for each question** in the test set

$$\text{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

Evaluation of QA models

- Other metrics (ignoring punctuations and articles like *a*, *an*, *the*)
  - Exact match: The % of predicted answers that match the gold answer exactly
  - F1 score: The average word/token overlap between predicted and gold answers
- Test sets
  - TREC QA dataset
  - AI2 reasoning challenge (multiple choice questions that are hard to answer using simple lexical methods)

Which property of a mineral can be determined just by looking at it?
(A) luster [correct] (B) mass (C) weight (D) hardness