Open-Domain QA

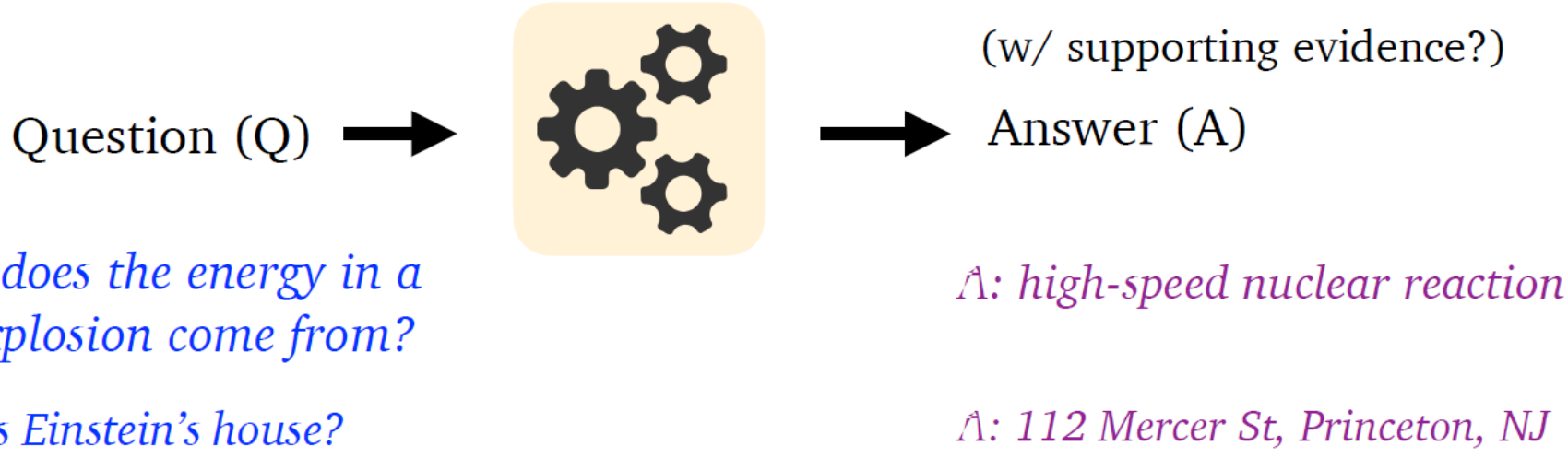
2022.9.9

分享人：陈思芹

目录

开领域问答

问答任务（Question Answering，QA）是自然语言处理中的基本任务之一。



- Closed-domain QA：给定支撑素材，类似阅读理解
- Open-domain Question Answering (OpenQA)：

  基于大型的数据源（如，100万+文档），搜索并提取答案

发展

最早的 QA 系统研究起于 1961 年， 针对闭领域问答的研究。

应用:

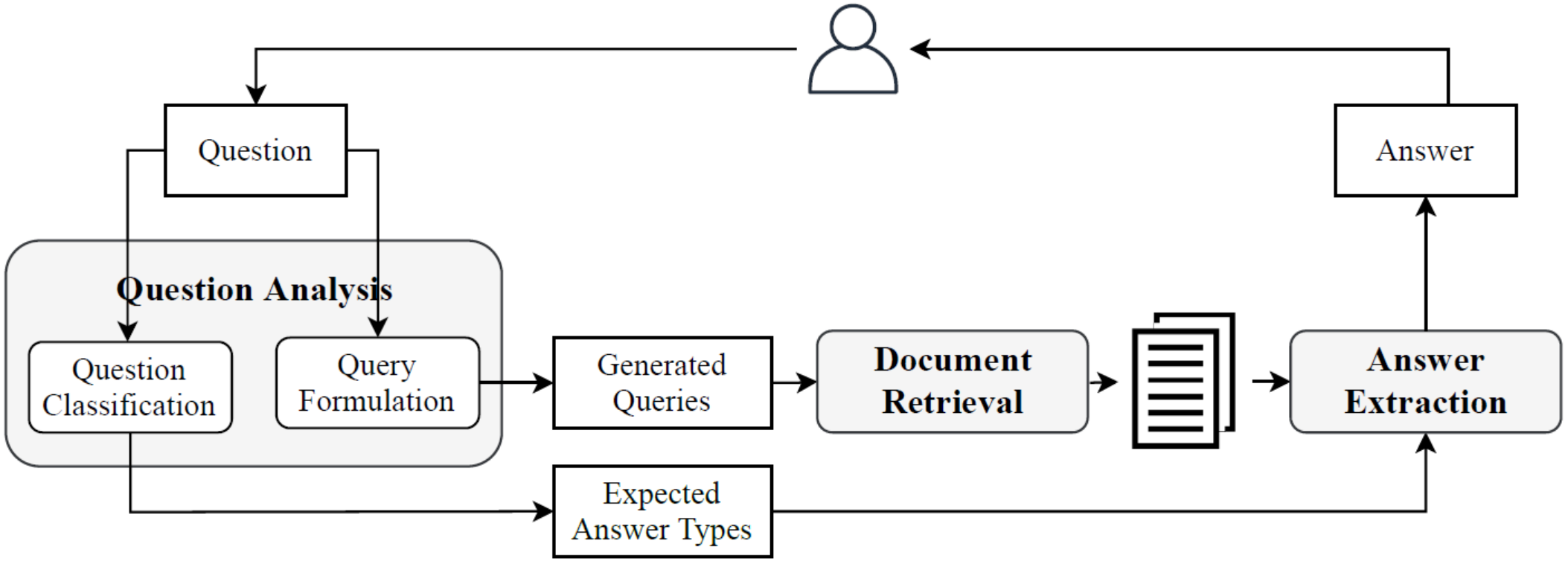
- Machine Reading Comprehension (MRC)
- 体育比赛 (American baseball games)
- 学术研究 (LUNAR/MURAX, academic tool)

1999 年, TREC 提出了 OpenQA 任务的定义。

World Wide Web 的广泛使用，启发人们将搜索引擎作为检索器引入开领域问答。

发展

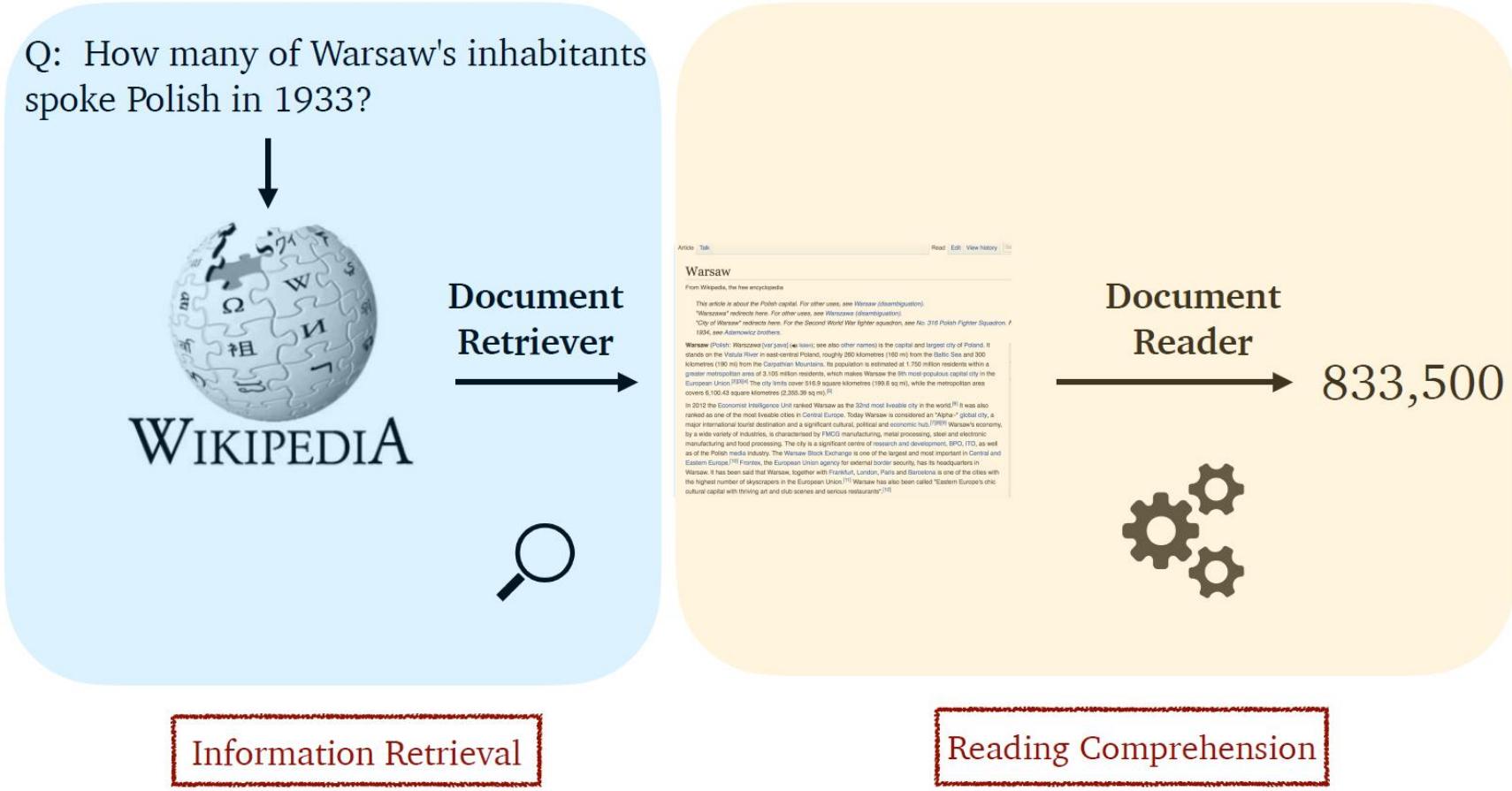
早期的三阶段的开领域问答框架：问题分析、文档检索和答案提取



- 基于启发式规则
- （动物，食物，人名，地名，数值……）
- 对实体进行分类找答案

发展

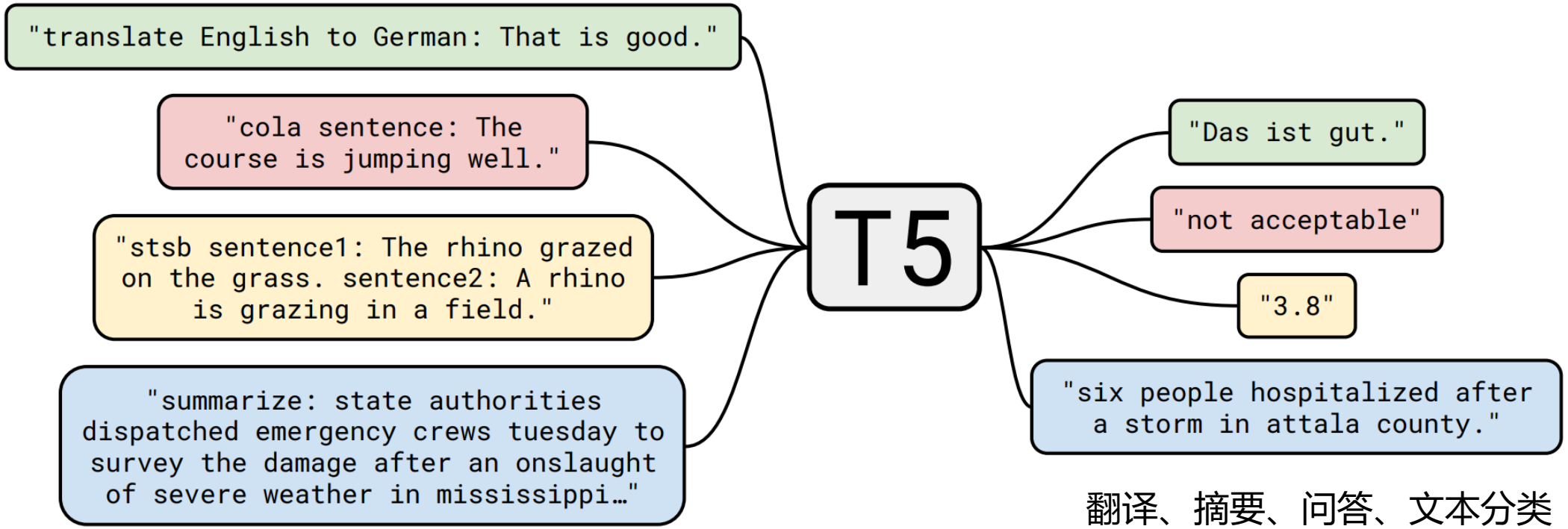
两阶段的开领域问答框架：取消了问题分析，目前主流的 Retriever-Reader 框架



发展

前沿研究：Retrieval-free

T5：Text-to-Text Transfer Transformer（2019年），一个能实现文本转换的大型生成模型

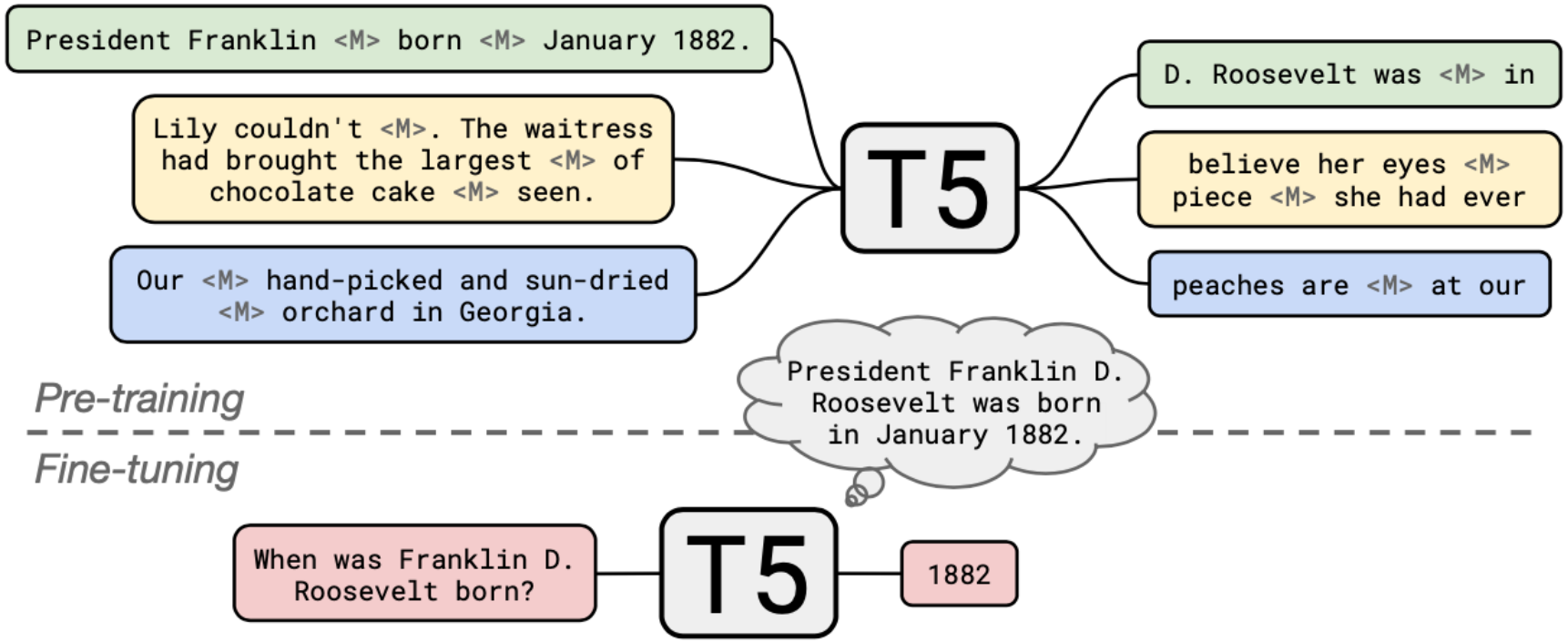


翻译、摘要、问答、文本分类

发展

前沿研究：Retrieval-free

依赖于模型的庞大参数量，经过充分训练后的 T5 能跳过检索阶段，直接生成答案



问答任务分类

QA task

- Closed QA：给定素材
- Open QA：大规模数据源
  - texts
  - tables
  - Knowledge bases

Knowledge Bases

Structured

Tables

Category	Structure	Country	City	Height (metres)	Height (feet)
Mixed use	Burj Khalifa	United Arab Emirates	Dubai	829.8	2,722
Self-supporting tower	Tokyo Skytree	Japan	Tokyo	634	2,080
Mixed use	Shanghai Tower	China	Shanghai	632	2,073
Clock building	Abraj Al Bait Towers	Saudi Arabia	Mecca	601	1,972
Military structure	Large masts of INS Kattabomman	India	Tirunelveli	471	1,545
Mast radiator	Lualualei VLF transmitter	United States	Lualualei, Hawaii	458	1,503
Twin towers	Petronas Twin Towers	Malaysia	Kuala Lumpur	452	1,482
Residential	432 Park Avenue	United States	New York	425.5	1,396
Chimney	Ekibastuz GRES-2 Power Station	Kazakhstan	Ekibastuz	419.7	1,377
Radar	Dimona Radar Facility	Israel	Dimona	400	1,312
Lattice tower	Kiev TV Tower	Ukraine	Kiev	385	1,263
Electricity pylon	Zhoushan Island Overhead Powerline Tie	China	Zhoushan	370	1,214

Semi-structured

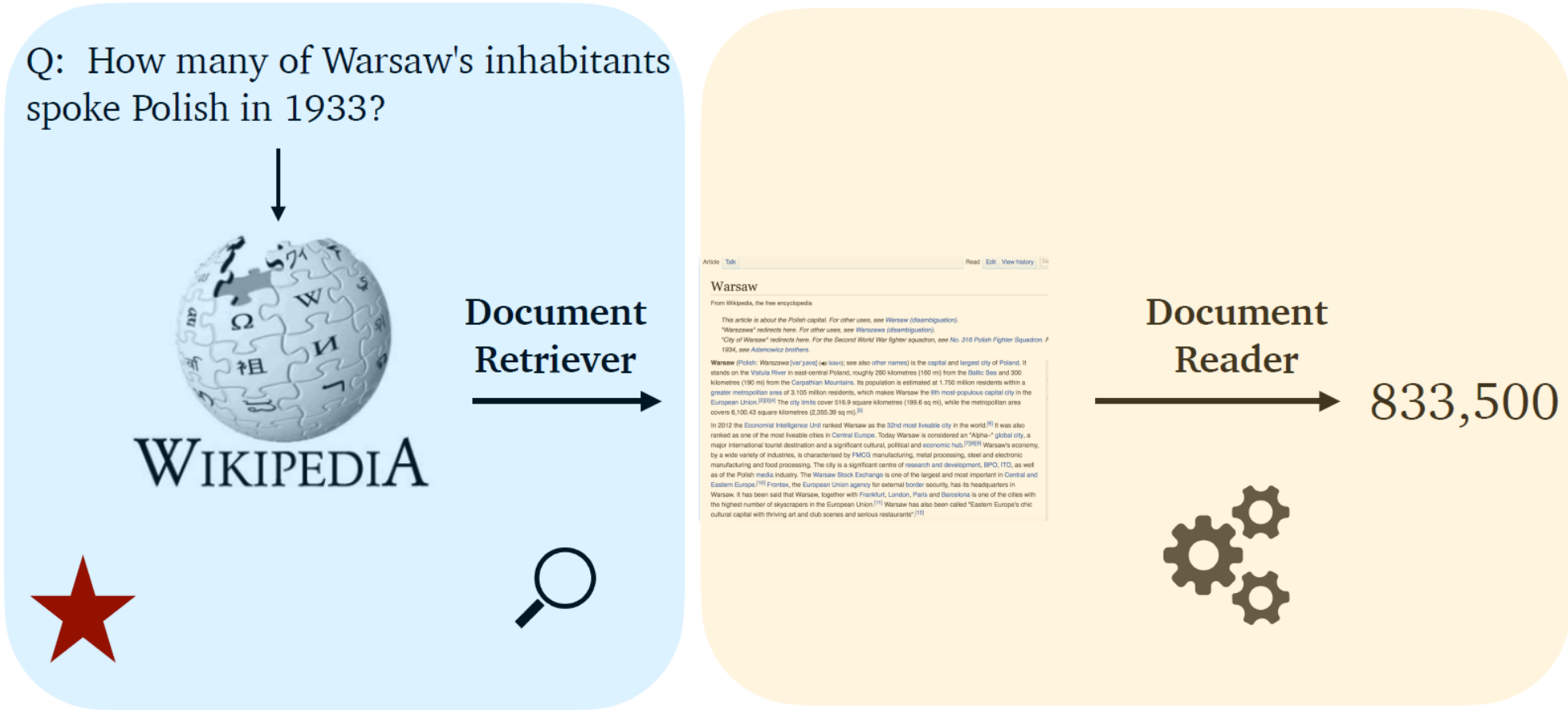
Web Documents & Wikipedia

Unstructured

目录

检索部分

Retriever：从大量检索单位的语料库中，找到和问句相关的段落或表格



2.1 TF-IDF

- **TF-IDF** 是一个简单的加权技术，可以用来衡量一个字词对于文档的重要程度。
- 两部分组成：
  - **TF(Term Frequency)**，也称为词频。即某个词在文档中出现的次数。可将实际的次数除以文档的总词数进行标准化：

$$\text{词频(TF)} = \frac{\text{某个词在文章中的出现次数}}{\text{文章的总词数}}$$

  - **IDF(Inverse Document Frequency)**，也称为逆文档频率。用于衡量一个词在语料库中的稀有程度。

$$\text{逆文档频率(IDF)} = \log\left(\frac{\text{语料库的文档总数}}{\text{包含该词的文档数} + 1}\right)$$

2.1 TF-IDF

- TF-IDF 的计算：将 TF 和 IDF 相乘

$$\mathrm{TF-IDF} = \text{词频}(\mathrm{TF}) \times \text{逆文档频率}(\mathrm{IDF})$$

- **TF-IDF 与一个词在文档中的出现次数成正比，与该词在整个语料库中的出现次数成反比。**

- 作用：自动提取文档的关键词
- 例子：一篇长度为1000个词的文档，“中国”、“蜜蜂”、“养殖”各出现20次。但它们在语料库中出现的次数不同。
- 计算 TF-IDF 后，可以“蜜蜂”作为关键词。

	包含该词的文档数（亿）	IDF	TF-IDF
中国	62.3	0.603	0.0121
蜜蜂	0.484	2.713	0.0543
养殖	0.973	2.410	0.0482

2.2 BM25

- **BM25**（BM: best matching）算法是一种计算句子与文档相关性的算法。可以看做 TF-IDF 的改进。
- 先将输入的句子分词，然后计算句子中每个词与文档的相关性，最后进行加权求和。
- 计算公式:

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgdl})}$$

  - Score(D,Q)：D 代表文档，Q 代表问句
  - q_i：问句中的各分词
  - IDF（q_i）:该词的逆文档频率
  - $f(q_i,D)$:即TF，该词在文档中的词频

2.2 BM25

- 和 TF-IDF 不同的是，BM25 对词频做了“标准化处理”。
- 词频和相关性之间的关系是非线性的，每一个词对于文档相关性的分数不会超过一个特定的阈值，当词出现的次数达到一个阈值后，其影响不再线性增长。

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgdl})}$$

  - k1：自由参数（k1>=0），默认1.2。对文档中的词项频率进行缩放控制。
  - b：自由参数（0≤ b≤ 1）， 默认0.75。决定文档长度对缩放的影响程度。
  - |D|：文档长度； avgdl：整个文档集的平均长度。越长的文档包含的单词越多，词频数也就越高。因此需要标准化。

2.3 倒排索引

- 倒排索引（Inverted Index）：将文档-关键词的对应，转换成关键词-文档的对应
- 作用：避免对所有文档做相似度计算，加快检索速度

例子：

文章1的内容为：Tom lives in Guangzhou, I live in Guangzhou too.

文章2的内容为：He once lived in Shanghai.

文章1的所有关键词为：[tom] [live] [guangzhou] [i] [live] [guangzhou]

文章2的所有关键词为：[he] [live] [shanghai]

关键词	文章号
guangzhou	1
he	2
i	1
live	1,2
shanghai	2
tom	1

关键词	文章号[出现频率]	出现位置
guangzhou	1[2]	3，6
he	2[1]	1
i	1[1]	4
live	1[2]	2，5，
	2[1]	2
shanghai	2[1]	3
tom	1[1]	1

2.3 倒排索引

- BM25 实战：可以使用 ElasticSearch 相似性模块内置的 BM25 实现。

```
es.indices.close(index=index)
es.indices.put_settings(index=index, settings={
    "similarity" : {
        "default" : {
            "type" : "BM25",
            "b":b,
            "k1":k1
        }
    }
})
es.indices.open(index=index)
```

```
query = {
    "match": {
        "content": question
    }
}
## 搜索相似度的表格
top_k=200
res = es.search(index=index, size=top_k, query=query)
```

- K1,b 等参数可以根据需要自行设置。

2.4 DPR

Karpukhin et al., 2020. Dense Passage Retrieval for Open-Domain Question Answering

- 问题：
  - TF-IDF 和 BM25 算法属于稀疏向量模型，仅考虑了词的匹配，没有考虑到语义的相关性
  - 检索器不能学习

- **DPR**（Dense Passage Retrieval），密集向量检索
  - 通过预训练模型（如 BERT）将问句和段落转换成代表语义信息的向量形式
  - 计算向量的内积得到两者的相似得分

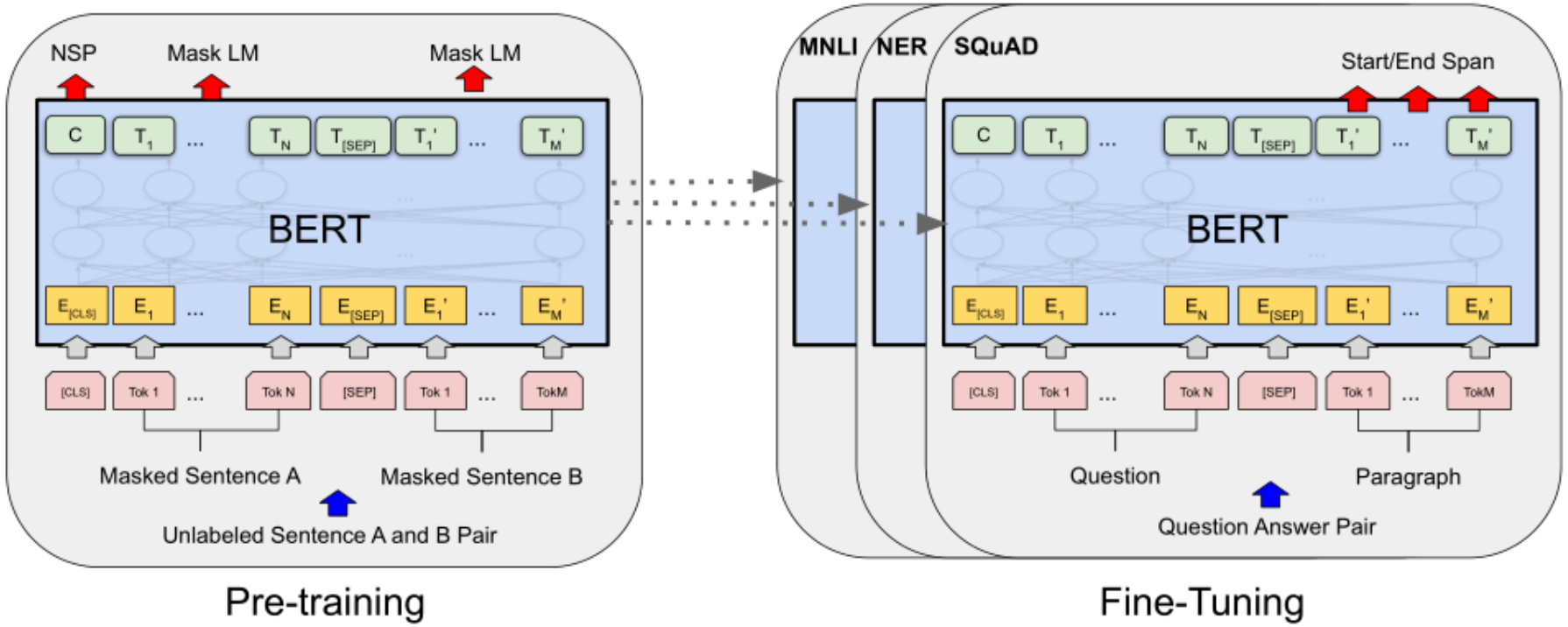
  - 出现条件
    - 预训练模型的出现，如 BERT
    - 最大内积搜索技术（MIPS）的发展，如相似向量检索库 FASSI 等的出现

2.4 DPR

Karpukhin et al., 2020. Dense Passage Retrieval for Open-Domain Question Answering

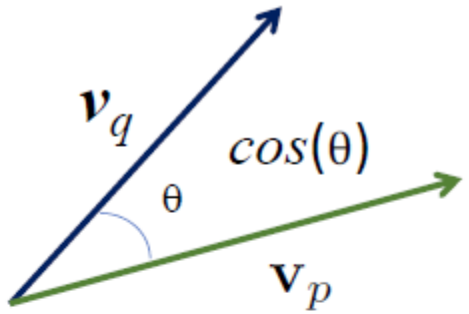
BERT：

- Google AI 提出的预训练模型
- Pre-training：利用无标签的自由文本，掩码预测、预测下一句等任务训练，耗时长
- Fine-Tuning：利用有标签数据训练模型，以适用于不同的下游任务，耗时短



2.4 DPR

Karpukhin et al., 2020. Dense Passage Retrieval for Open-Domain Question Answering



$d_1 >> d_2$

sparse repr: [0… 1 …. 1 …0..1] $\in \mathbb{R}^{d_1}$

dense repr: [1.03, -5.72, 6.42, .., 9.91] $\in \mathbb{R}^{d_2}$

sparse

"How many provinces did the Ottoman empire contain in the 17th century?"

"What part of the atom did Chadwick discover?"

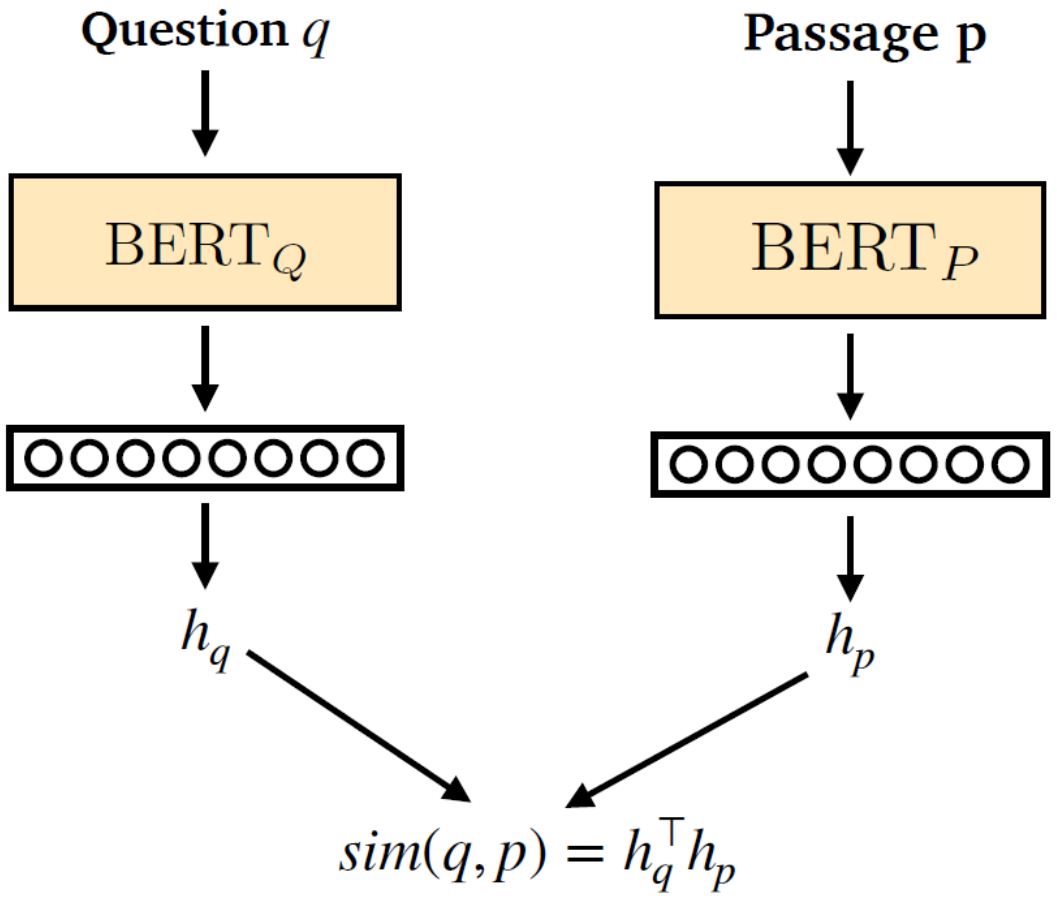
dense

"Who is the **bad guy** in lord of the rings?"

*Sala Daker is an actor and stuntman from New Zealand. He is best known for portraying the **villain** Sauron in the Lord of the Rings trilogy by Peter Jackson.*

2.4 DPR

Karpukhin et al., 2020. Dense Passage Retrieval for Open-Domain Question Answering



How to get positives and negatives?

$$\mathcal{D} = \{\langle q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^- \rangle\}_{i=1}^m$$

$$L(q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^-) = -\log \frac{e^{\text{sim}(q_i, p_i^+)}}{e^{\text{sim}(q_i, p_i^+)} + \sum_{j=1}^n e^{\text{sim}(q_i, p_{i,j}^-)}}$$

2.4 DPR

Karpukhin et al., 2020. Dense Passage Retrieval for Open-Domain Question Answering

- negative 段落选取的三种方案：
  - Random：随机在语料库中抽取一定数量的段落
  - BM25：取与问句的BM25 相似得分较高的、同时没有包含答案的段落
  - Gold：与其他问句匹配的段落，但与该问句不匹配

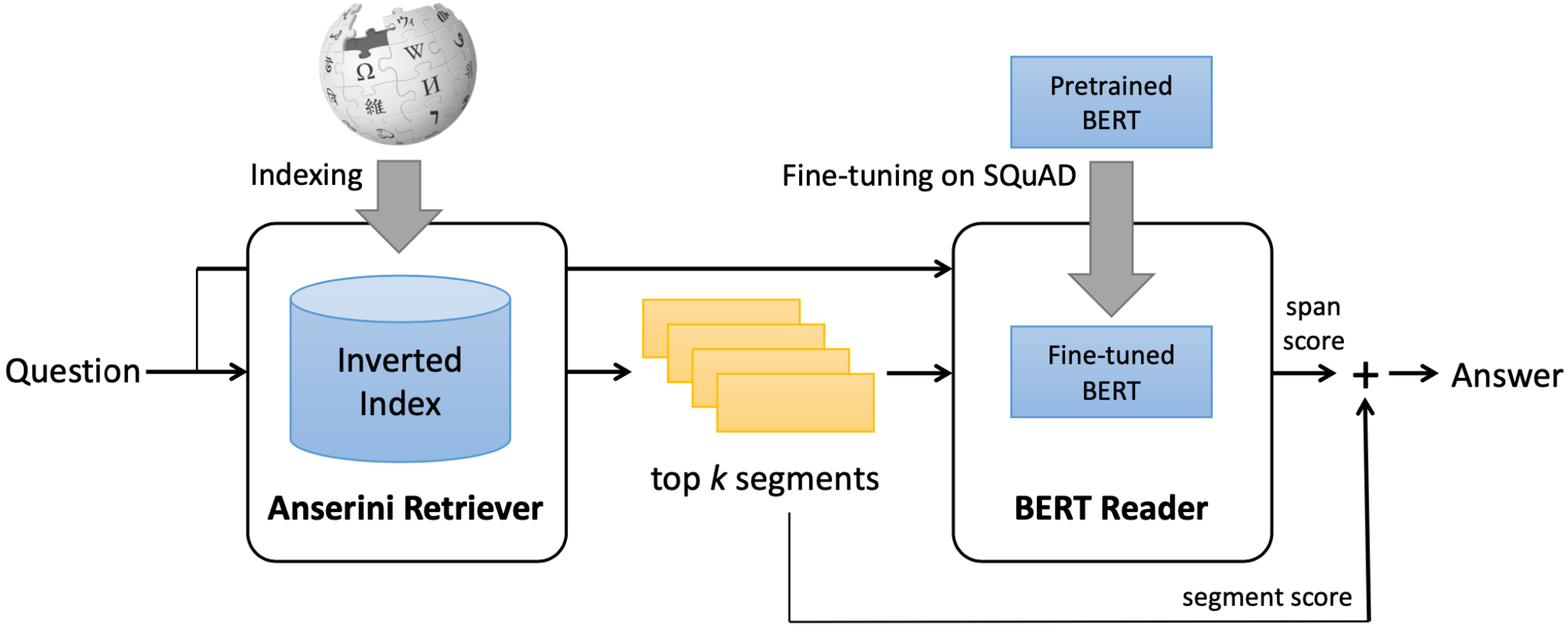
- In-batch negative（B-1 个，论文中 B=64） + Hard negative（BM25 分数最高的 1 个）

- 即能从In-batch negative 中学习到大范围的语义不同 ，又能从hard negative 中学习到细节上的不同语义，有利于提升模型的泛化能力。

2.4 DPR

Karpukhin et al., 2020. Dense Passage Retrieval for Open-Domain Question Answering

- 提取答案：基于 bert 的段落阅读理解



目录

3.1 DTR

2021，Open Domain Question Answering over Tables via Dense Retrieval

- 表格存在一定结构
- 行列中的内容存在统一格式和内在相关性
- BERT 对文本编码取得了较好的效果，但缺乏对表格结构的理解

DTR 是一个密集向量的表格检索器（Dense Table Retriever）

- 检索流程与 DPR 有着较大的相似性
- 将 BERT 替换成针对表格的预训练模型 Tapas
- 基于 DTR 的 Open-domain Table QA 模型

3.1 DTR

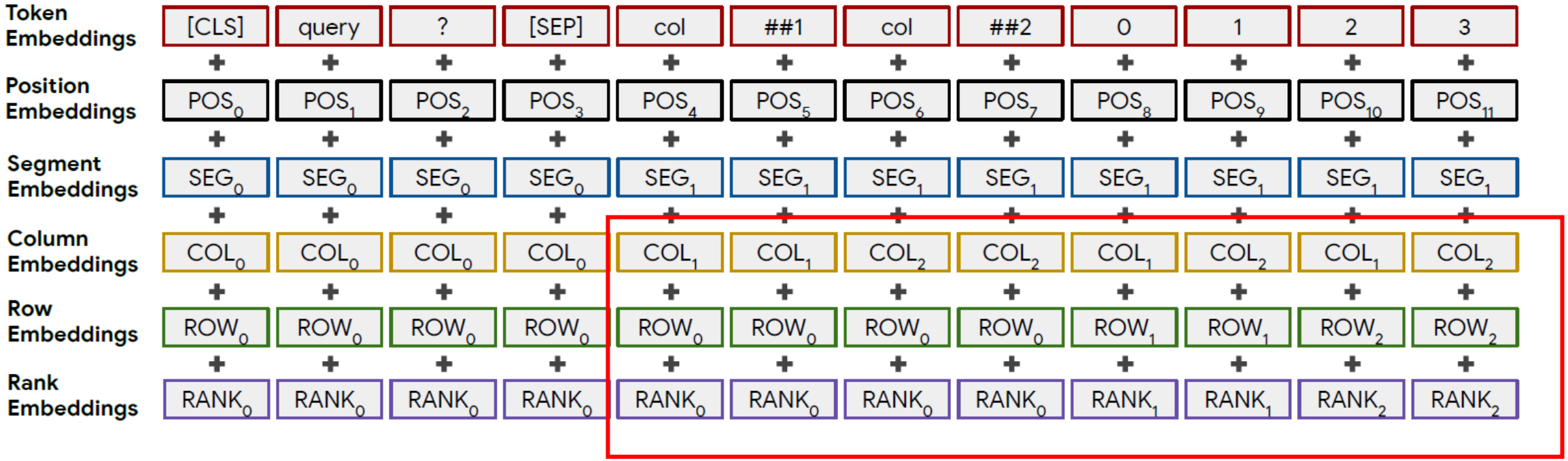
2021，Open Domain Question Answering over Tables via Dense Retrieval

Tapas

- 附加额外的编码，用于编码表格的结构
  - Column/Row Embedding,
  - Rank Embedding

Table

col1	col2
0	1
2	3



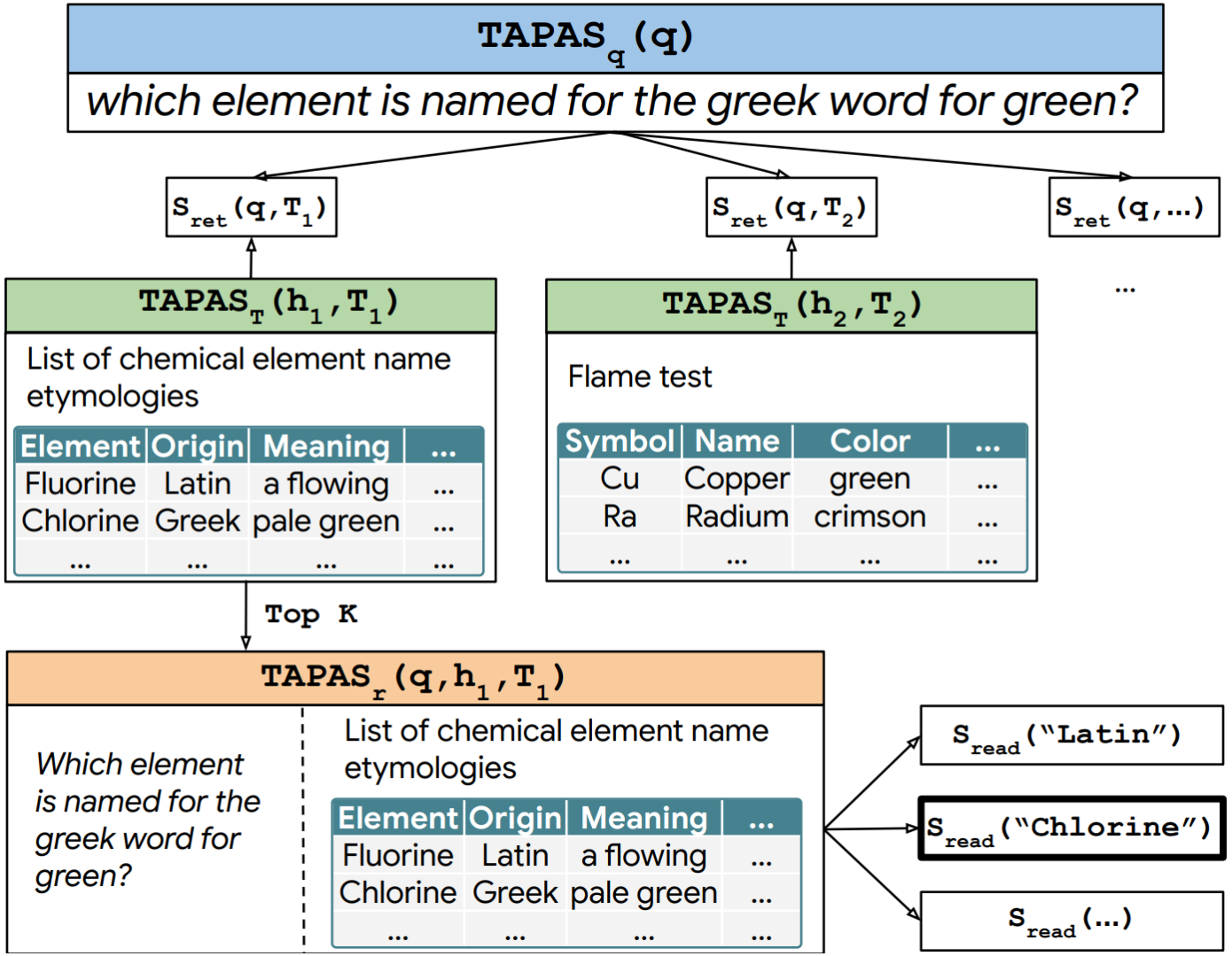
3.1 DTR

2021，Open Domain Question Answering over Tables via Dense Retrieval

基于 DTR 的表格问答

- 3 个不同的 Tapas 模型
  - $Tapas_q$：编码问句
  - $Tapas_T$：编码表格
  - $Tapas_r$：闭领域 Table QA

- 训练 DTR（$Tapas_q$ 和$Tapas_T$）
  - 重新预训练
  - 21.3M 的 Text-Table 对，逆完型填空任务
  - 微调过程和 DPR 类似



3.1 DTR

2021，Open Domain Question Answering over Tables via Dense Retrieval

NQ-Table 数据集：提取 NQ 中包含表格的部分问题；表格去重

DTR 表格检索

Model	R@1	R@10	R@50
BM25	16.77	40.06	58.39
DTR-Text	32.90	72.00	86.86
DTR-Schema	34.36	74.24	88.37
DTR	36.24	76.02	90.25
DTR +hnbm25	42.17	80.51	92.31
DTR +hn	**42.42**	**81.13**	**92.56**
DTR -pt	16.64	47.80	68.68

基于 DTR 的表格问答

Retriever	Reader	EM	F1	Oracle EM	Oracle F1
BM25	TAPAS	21.46	28.24	29.51	40.79
DTR-Text	BERT	29.58	37.38	39.39	51.48
DTR-Text	TAPAS	33.78	43.49	42.83	56.46
DTR-Schema	TAPAS	32.75	42.19	42.63	55.05
DTR	TAPAS	35.50	45.44	46.09	59.01
DTR +hnbm25	TAPAS	36.61	46.74	47.46	60.72
DTR +hn	TAPAS	**37.69**	**47.70**	**48.20**	**61.50**

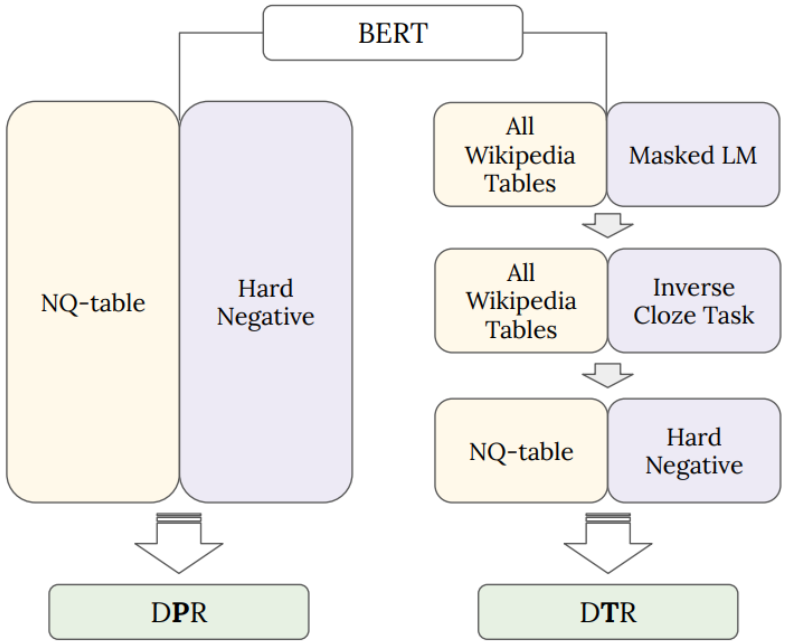
3.2 表格检索有必要使用专门针对表格的模型设计吗?

2022，Table Retrieval May Not Necessitate Table-specific Model Design

DTR 的缺陷：

- 在强调内容的检索任务中，对于表格结构的 Embedding 是否有必要?
- 相较 DPR，DTR 包含了复杂的预训练流程

Embeddings	DPR	DTR
token	BERT vocab	BERT vocab
segment	0 for all tokens	0 for text, 1 for table
position	sequential	cell-wise reset
row	-	row index
column	-	column index
rank	-	rank of token value



是否可以用基于 text 的 Retriever ，达到与针对 table 设计的 Retriever 接近的检索效果?

3.2 表格检索有必要使用专门针对表格的模型设计吗?

2022，Table Retrieval May Not Necessitate Table-specific Model Design

假设：在文本检索数据集上学习的强大内容匹配的能力，可以转移到表格检索任务上

比较了 DPR 和 DTR

- 两者均取官方发布的检查点
- DPR-table 在原检查点的基础上，继续在 NQ-table 上进行微调

结果显示 DPR-table 在 small k 上的表现更好

Model	Retrieval Accuracy				
	@1	@5	@10	@20	@50
DTR (medium)	62.32	82.51	86.75	91.51	94.26
DTR (large)	63.98	84.27	**89.65**	**93.48**	**95.65**
BERT-table	60.97	79.81	85.51	88.20	91.62
DPR	57.04	80.54	86.13	89.54	92.34
DPR-table	**67.91**	**84.89**	88.72	90.58	92.86

实验 1：探究是否只使用线性化的输入也可以在一定程度上体现表格结构

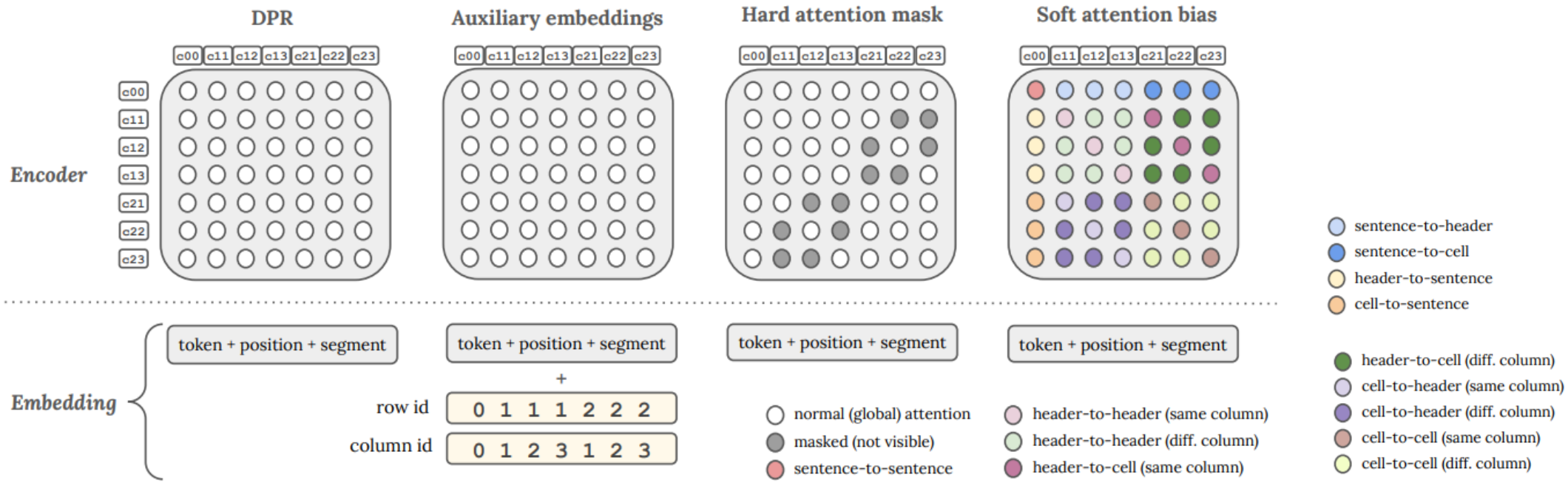
- 随机打乱表格行列顺序后，检索性能是否有一定下降?
- 增加行列间的分隔符，检索性能是否有一定提升?

3.2 表格检索有必要使用专门针对表格的模型设计吗?

2022，Table Retrieval May Not Necessitate Table-specific Model Design

实验 2：探究对表格做额外的结构编码是否可以提升表格的检索效果

- 增加 row/column 编码
- 使用 Hard attention mask 表明单元格之间是否有行列关系
- 使用 Soft attention bias 表明单元格之间的不同种类关系



3.2 表格检索有必要使用专门针对表格的模型设计吗?

2022，Table Retrieval May Not Necessitate Table-specific Model Design

实验结论

- 当打乱行列间顺序，尤其是打乱了同行的单元格的顺序后，检索效果变差。说明文本检索器能通过线性化一定程度上捕捉到表格的结构模式。

- 当增加行列间分隔符后（单元格之间加“|”，行结尾加“.”），检索效果有明显提升。

- 针对表格的额外编码的三种方法，均对检索效果的提升微乎其微。

表格检索中，针对表格的复杂模型设计可能没有必要。可以用增加间隔符这样的简单方法，使针对文本的 `Retriever` 达到类似的学习表格结构的效果。

目录

4.1 论文工作

Retrieval Augmented via Execution Guidance in Open-domain Table QA

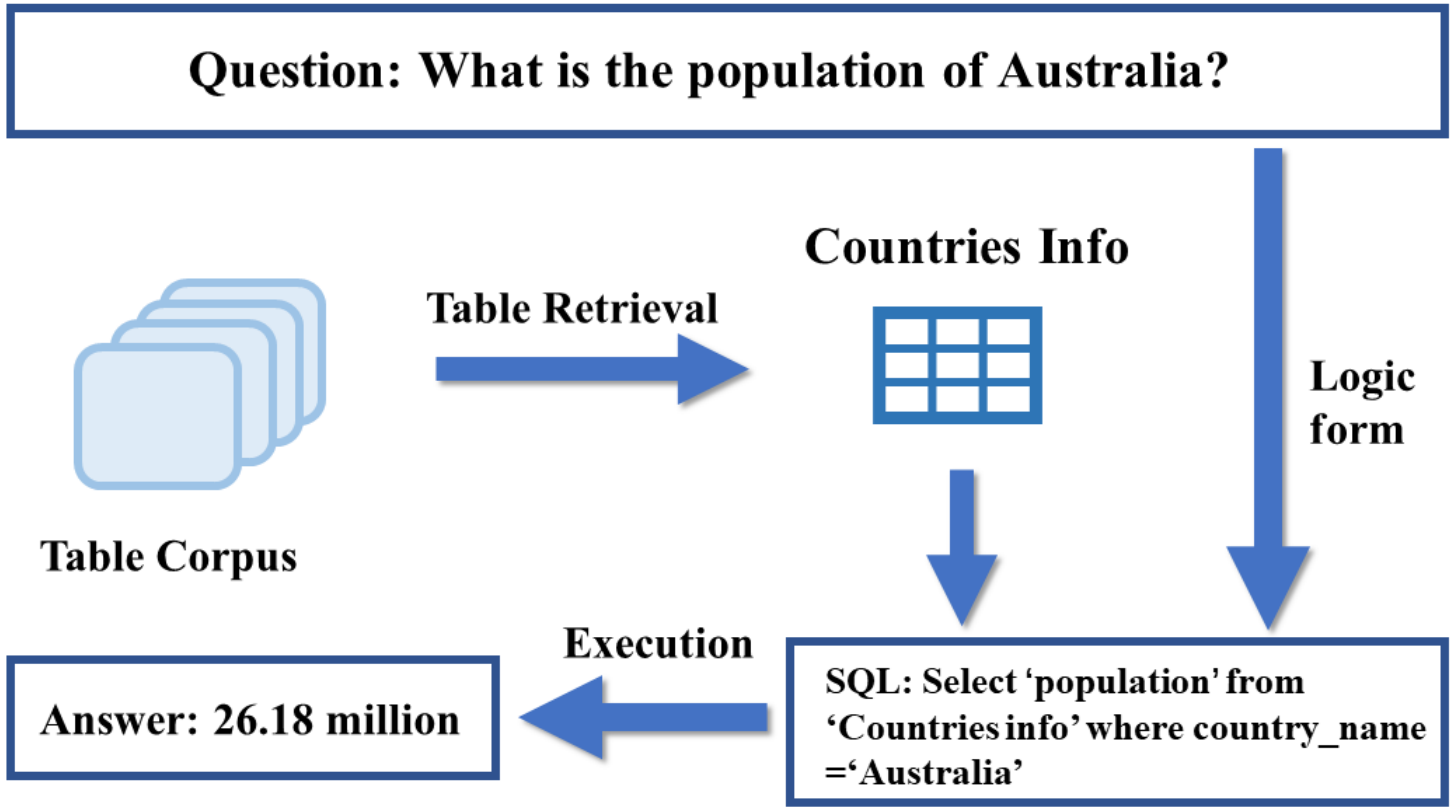
基于执行引导的表格检索增强

- 问题：
  - 基于表格结构编码的表格检索存在缺陷
  - 弱监督的答案提取需要将整个表格输入模型，对表格大小有较大限制

- 论文工作：
  - 开领域 Text-to-SQL 任务探索。
  - 将执行引导机制，引入开领域表格问答的检索阶段，以实现表格检索的强化。
  - 基于原本的 WikiSQL数据集，生成开领域的 Open-domain WikiSQL 数据集。

4.1 论文工作

1. 开领域 Text-to-SQL 任务探索
   - 和闭领域 Text-to-SQL 相比，增加了表格检索阶段
   - 将执行的结果作为 QA 任务的答案



4.1 论文工作

2. 基于执行引导的表格检索

- 执行引导（Execution Guidance,EG）：以执行是否出现 runtime error 来指引 SQL 生成
- 执行引导能否能扩展至开领域 Text-to-SQL 的表格检索，作为准确性的衡量指标之一？

在Open-domain WikiSQL数据集中抽取了1000问题，将其对应的候选表以随机的一个表填充。

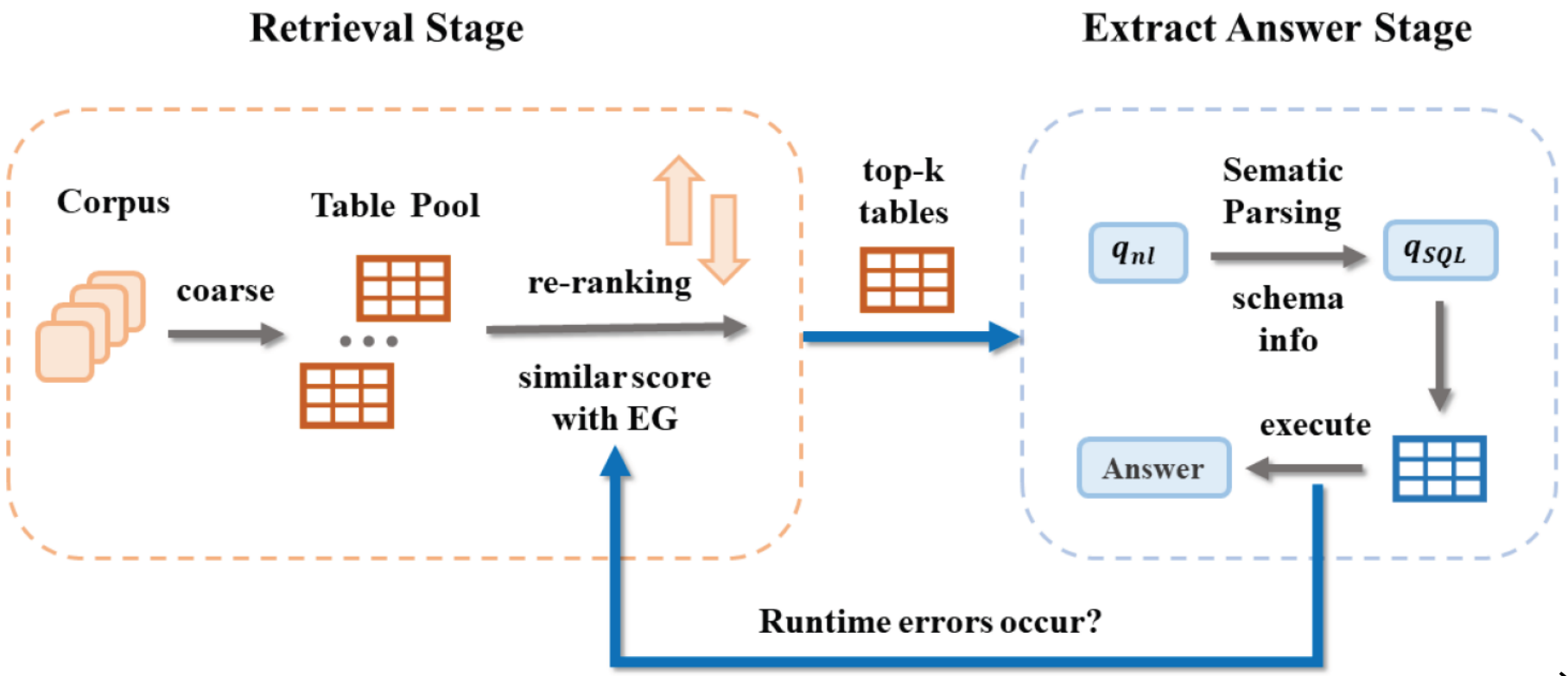
如果 SQL 在一个候选表格上发生了执行错误，则有较大的可能检索出的表是错误。

候选表正确	候选表错误
961（正常执行）	206（正常执行）
39（执行错误）	794（执行错误）

4.1 论文工作

2．基于执行引导的表格检索

- 检索中的执行引导：以执行是否出现 runtime error 作为检索 re-ranking 的一个指标



$$sim_{withEG} = (1-\alpha)\cdot sim_{origin}/maxsim_{origin} + \alpha \cdot res_{EG}$$

$res_{EG} = 1$：没有执行错误

$res_{EG} = 0$：发生执行错误

4.1 论文工作

3. Open-WikiSQL

- 对原来的 WikiSQL 做修改，使其符合开领域设置
- 关键：去语境化 (Decontextualization)
  - 代词替换
  - 名称补全
  - 范围限定
  - 添加背景信息

Table name: Central States Collegiate Hockey League

Question: How many of the schools listed are in Ames, IA?

+ 范围限定

Question: For all the schools in Central States Collegiate Hockey League(CSCHL), How many of the schools are in Ames, IA?

4.2 实验结果

实验结果

Table 2. Experimental Results of Table Retrieval

Method	*hit*@1	*hit*@5	*hit*@10	*hit*@20	*hit*@50	*hit*@100
bm25 (baseline)	43.62	62.21	68.45	74.36	81.98	87.29
bm25 (with EG)	60.03	77.99	82.66	86.54	89.44	90.71
DPR (delimiter, none EG)	58.84	79.05	85.33	89.77	94.59	96.91
DPR (delimiter, with EG)	77.97	90.50	93.40	95.37	97.22	97.97
DPR (none delimiter, none EG)	57.24	78.80	84.94	89.75	94.79	97.01
DPR (none delimiter, with EG)	76.89	90.21	93.22	95.48	97.14	97.72

Table 3. Experimental Results of Different α Weight

α	*hit*@1	*hit*@5	*hit*@10	*hit*@20	*hit*@50	*hit*@100
0.9	77.97	90.50	93.40	95.37	97.22	97.97
0.5	77.97	90.50	93.40	95.37	97.22	97.97
0.3	77.97	90.50	93.40	95.37	97.22	97.97
0.1	76.97	90.46	93.47	95.68	97.57	98.20
0 (none EG)	58.84	79.05	85.33	89.77	94.59	96.91

Table 4. Experimental Results of End-to-end Open-domain Text-to-SQL

Method	acc_{lf}	acc_{ex}
bm25 (baseline)	47.40	51.55
bm25 (with EG)	59.51	64.11
DPR (delimiter, none EG)	62.52	66.98
DPR (delimiter, with EG)	75.28	80.13

4.3 局限性

局限性

- 对表格schema 信息的完整性有较高的要求。一些 web 表格没有包含详细的模式信息，此时执行引导无法充分发挥出效果。
- 模型在转换和执行SQL时需要一定的时间成本和计算资源。粗检索步骤中的 table pool size 会受到限制。
- 只适用于简单SQL的问句和单表检索情况。

总结

1. 问题背景、两阶段检索框架
2. 检索部分
   - 经典方法：TF-IDF，BM25
   - 深度学习：DPR
3. 表格开领域问答
   - DTR
   - 对 DTR 的反思
4. 论文工作

参考文献

2017. Reading Wikipedia to Answer Open-Domain Questions.

2021. Decontextualization: Making sentences stand-alone.

2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

2021. Open Domain Question Answering over Tables via Dense Retrieval.

2020. TaPas: Weakly Supervised Table Parsing via Pre-training.

2020. Dense Passage Retrieval for Open-Domain Question Answering.

2020. Hybrid Ranking Network for Text-to-SQL.

2009. The probabilistic relevance framework: BM25 and beyond.

2018. Robust text-to-sql generation with execution-guided decoding

2022. Table Retrieval May Not Necessitate Table-specific Model Design.

谢谢