FNLP
Information Retrieval

Yansong Feng
fengyansong@pku.edu.cn

Wangxuan Institute of Computer Technology
Peking University

May 21, 2025

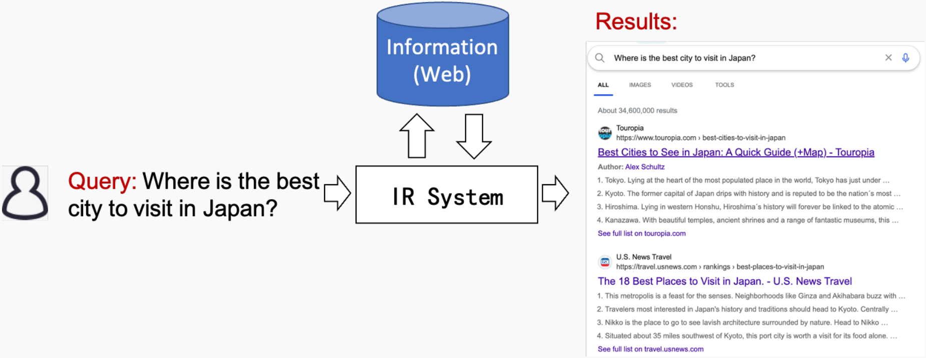
Outline

Outline

Information Retreival

Information Retreival (IR): to retrieve all types of data based on users' information needs.

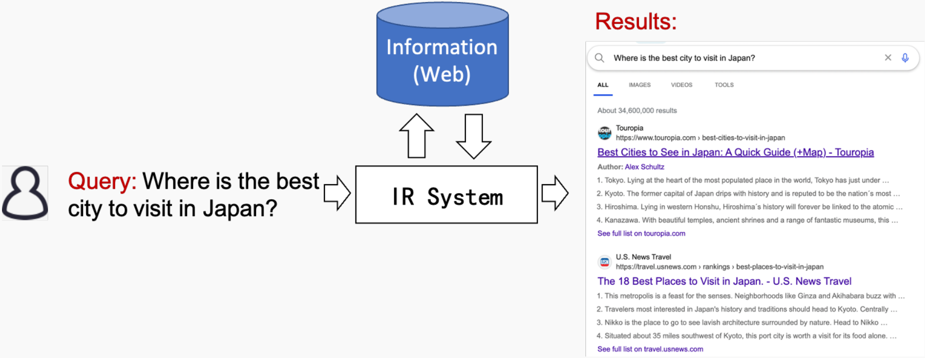
- A query from users
- An Information Retrieval system and a collection of data
- Output results



Information Retreival

Information Retreival (IR): to retrieve all types of data based on users' information needs. Question Answering can be seen as a type of IR.

- A query from users
- An Information Retrieval system and a collection of data
- Output results



Information Retreival

We expect an IR system to provide results that are:

- Accurate
- On time
- Comprehensive
- Diversity
- ...

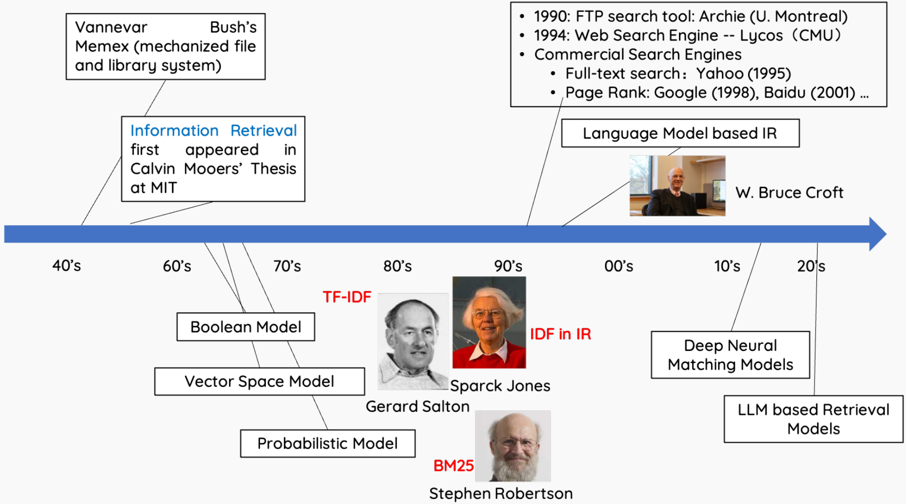
Information Retreival

We expect an IR system to provide results that are:

- Accurate
- On time
- Comprehensive
- Diversity
- ...

This field covers a wide range of topics across Natural Language Processing, Machine Learning, Data Mining, Database, System Architecture, Parallel Computing, etc.

A Brief Timeline of IR



Wide Applications

- Search Engines
- Recommender Systems
- Electronic Commerce
- Online Advertisement
- Social Media
- Intelligence Analysis

The Key in IR

To compute the similarity or relevance between a query and candidates

- Search Engines : query ⇒ web pages
- Recommender Systems : user ⇒ goods
- Electronic Commerce : user ⇒ goods/clients
- Online Advertisement : user ⇒ adverstisements
- Social Media : user ⇒ friends, tweets, videos, ...
- Intelligence Analysis : claims ⇒ evidence, clues, ...

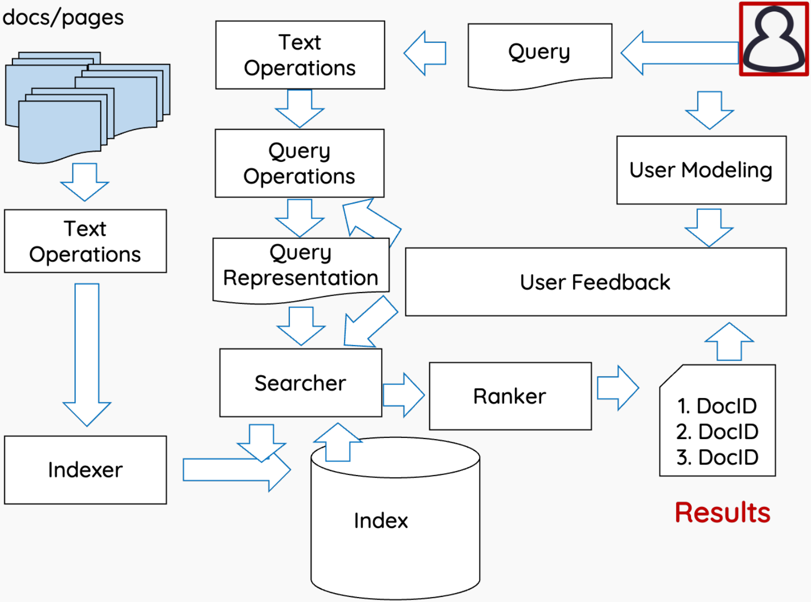
The Key in IR

To compute the similarity or relevance between a query and candidates

- Search Engines : query $\Rightarrow$ web pages
- Recommender Systems : user $\Rightarrow$ goods
- Electronic Commerce : user $\Rightarrow$ goods/clients
- Online Advertisement : user $\Rightarrow$ adverstisements
- Social Media : user $\Rightarrow$ friends, tweets, videos, ...
- Intelligence Analysis : claims $\Rightarrow$ evidence, clues, ...

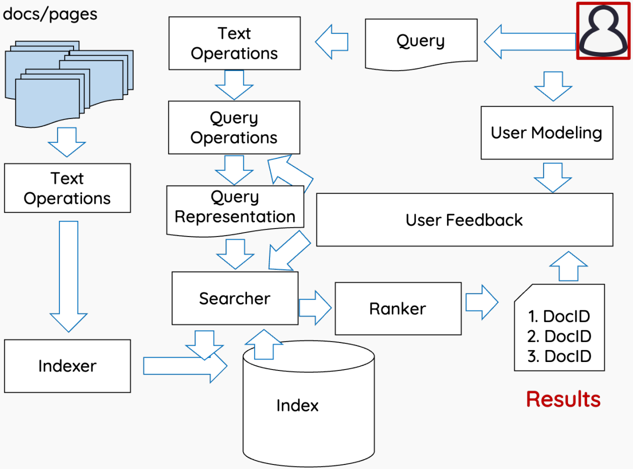
The computation is not limited in text but, really, as we expect, in a multimodal or crossmodal way.

The Main Architectures



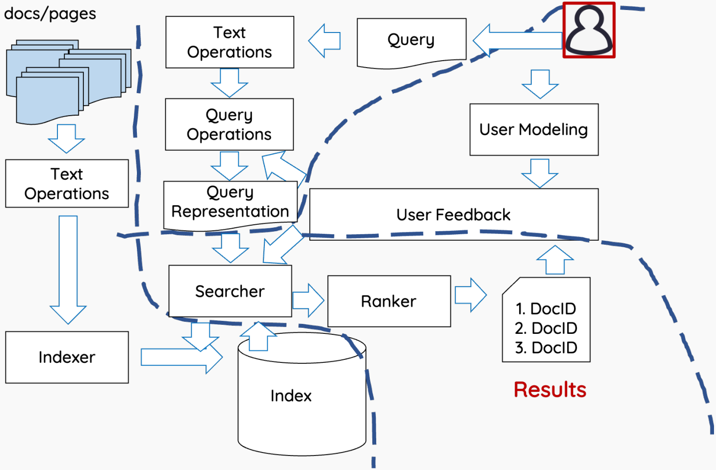
The Main Architectures

- Document Analysis and Indexer
- Query Analysis
- Searching and Ranking
- User Modeling



The Main Architectures

- Document Analysis and Indexer
- Query Analysis
- Searching and Ranking
- User Modeling



Outline

Definition

Given a vocabulary, $V = \{w_1, w_2, ...w_N\}$, a query $\mathbf{q} = w_{q1}, w_{q2}, ...w_{q3}$, a collection of documents, $D = \{d_1, d_2, ...d_K\}$, the task is to find a set of ds that are relevant to query q.

Definition

Given a vocabulary, $V = \{w_1, w_2, ...w_N\}$, a query $\mathbf{q} = w_{q1}, w_{q2}, ...w_{q3}$, a collection of documents, $D = \{d_1, d_2, ...d_K\}$, the task is to find a set of ds that are relevant to query q.

- assume $R(d)$ containing all ds relevant to query q
- the task will be to find a set $\tilde{R}(d)$ approaching $R(d)$

Definition

Given a vocabulary, $V = \{w_1, w_2, ...w_N\}$, a query $\mathbf{q} = w_{q1}, w_{q2}, ...w_{q3}$, a collection of documents, $D = \{d_1, d_2, ...d_K\}$, the task is to find a set of ds that are relevant to query q.

- assume $R(d)$ containing all ds relevant to query q
- the task will be to find a set $\tilde{R}(d)$ approaching $R(d)$
  - Solution 1: Select all relevant docs, $\tilde{R}(d) = \{d \in D | f(d, q) = 1\}$
    - Absolute relevance: really difficult to decide

Definition

Given a vocabulary, $V = \{w_1, w_2, ...w_N\}$, a query $\mathbf{q} = w_{q1}, w_{q2}, ...w_{q3}$, a collection of documents, $D = \{d_1, d_2, ...d_K\}$, the task is to find a set of ds that are relevant to query q.

- assume $R(d)$ containing all ds relevant to query q
- the task will be to find a set $\tilde{R}(d)$ approaching $R(d)$
  - Solution 1: Select all relevant docs, $\tilde{R}(d) = \{d \in D | f(d, q) = 1\}$
    - Absolute relevance: really difficult to decide
  - Solution 2: Rank all relevant docs in D, $\tilde{R}(d) = \{d \in D | f(d, q) > \theta\}$
    - Relative relevance: make sure to rank more relevant ds higher

Definition

Given a vocabulary, $V = \{w_1, w_2, ...w_N\}$, a query $\mathbf{q} = w_{q1}, w_{q2}, ...w_{q3}$, a collection of documents, $D = \{d_1, d_2, ...d_K\}$, the task is to find a set of ds that are relevant to query q.

- assume $R(d)$ containing all ds relevant to query q
- the task will be to find a set $\tilde{R}(d)$ approaching $R(d)$
  - Solution 1: Select all relevant docs, $\tilde{R}(d) = \{d \in D | f(d, q) = 1\}$
    - Absolute relevance: really difficult to decide
  - Solution 2: Rank all relevant docs in D, $\tilde{R}(d) = \{d \in D | f(d, q) > \theta\}$
    - Relative relevance: make sure to rank more relevant ds higher
- Often
  for q, d_i, d_j, to find a f so that $f(q, d_i) > f(q, d_j)$ if
  $p(\text{Relevance}|q, d_i) > p(\text{Relevance}|q, d_j)$

Main Steps

Conventionally, there will be 2 steps: retrieval and ranking

- Retrieval: whether a doc d is relevant to q
  - Try to recall as many relevant candidates as possible
- Ranking: which doc d is more relevant to q
  - Rank more relevant ds higher

Main Steps

Conventionally, there will be 2 steps: retrieval and ranking

- Retrieval: whether a doc d is relevant to q
  - Try to recall as many relevant candidates as possible
- Ranking: which doc d is more relevant to q
  - Rank more relevant ds higher

The key is to compute the relevance between q and d.

Main Steps

Conventionally, there will be 2 steps: retrieval and ranking

- Retrieval: whether a doc d is relevant to q
  - Try to recall as many relevant candidates as possible
- Ranking: which doc d is more relevant to q
  - Rank more relevant ds higher

The key is to compute the relevance between q and d.

- How to model the relevance computation
- How to represent query q and doc d

Main Steps

Conventionally, there will be 2 steps: retrieval and ranking

- Retrieval: whether a doc d is relevant to q
  - Try to recall as many relevant candidates as possible
- Ranking: which doc d is more relevant to q
  - Rank more relevant ds higher

The key is to compute the relevance between q and d.

- How to model the relevance computation
- How to represent query q and doc d

Why are these two aspects challenging?

In Literature

- Boolean Model
- Vector Space Model
- Probabilistic Model
- Statistical Language Model
- Neural Model

Outline

Outline

Boolean Model

Bool query: represent query q by connecting each w in q using predefined operators, such as AND, OR and NOT

Example

Query: like AND information AND retrieval

- DocID:1 I like machine learning
- DocID:2 Machine learning is different with deep learning
- DocID:3 Information retrieval is important for many applications
- DocID:4 I like information retrieval
- DocID:5 Machine learning is useful for ranking in search engine

Boolean Model

Bool query: represent query q by connecting each w in q using predefined operators, such as `AND`, `OR` and `NOT`

Example

Query: like `AND` information `AND` retrieval

- DocID:1 I like machine learning
- DocID:2 Machine learning is different with deep learning
- DocID:3 Information retrieval is important for many applications
- DocID:4 I like information retrieval
- DocID:5 Machine learning is useful for ranking in search engine

- We can not use brute-force search to go over all docs
- very SLOW, and hard to deal with `NOT`

Bool Model

Query： like AND information AND retrieval

like 1 → 4
information 3 → 4
retrieval 3 → 4

⇨ 4
Intersection

Query： like OR retrieval

like 1 → 4
retrieval 3 → 4

⇨ 1、3、4
Union

Query： like NOT retrieval

like 1 → 4
retrieval 3 → 4

⇨ 1
Subtraction

Vocab	Doc-Freq	DocID List
application	1	3
engine	1	5
deep	1	2
different	1	2
for	2	3 → 5
I	2	1 → 4
important	1	3
in	1	5
information	2	3 → 4
is	3	2 → 3 → 5
like	2	1 → 4
learning	3	1 → 2 → 5
machine	3	1 → 2 → 5
many	1	3
ranking	1	5
retrieval	1	3 → 4
search	1	5
useful	1	5
with	1	2

Outline

Vector Space Model

Problems with Bool Model

- not easy for users to build Bool queries
- all words are treated equally
- not easy to rank the candidates

Vector Space Model

Problems with Bool Model

- not easy for users to build Bool queries
- all words are treated equally
- not easy to rank the candidates

Vector Space Model: represent queries and docs using vectors, and treat the distance between vectors as relevance.

Vector Space Model

Problems with Bool Model

- not easy for users to build Bool queries
- all words are treated equally
- not easy to rank the candidates

Vector Space Model: represent queries and docs using vectors, and treat the distance between vectors as relevance.

- how to obtain the vector representations
- how to compute the distance between vectors

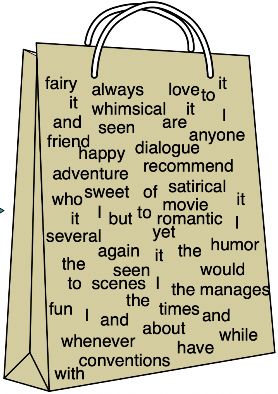
Vector Space Model

We did talk about this before!

The Bag-of-Words Representation

The so-called Bag-of-Words format

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

[Jurafsky and Martin, SLP3]

Weighting a Term

The weighted value or importance of a word t in a document d can be considered by taking both t's term frequency and inverse document frequency:

$$TF_{t,d} = count(t, d)$$

Weighting a Term

The weighted value or importance of a word t in a document d can be considered by taking both t's term frequency and inverse document frequency:

$$TF_{t,d} = \begin{cases} 1 + \log_{10} count(t,d) & \text{if } count(t,d) > 0 \\ 0 & \text{otherwise} \end{cases}$$

Weighting a Term

The weighted value or importance of a word t in a document d can be considered by taking both t's term frequency and inverse document frequency:

$$TF_{t,d} = \begin{cases} 1 + \log_{10} count(t,d) & \text{if } count(t,d) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$IDF_t = \log_{10} \frac{N}{DF_t}$$

Weighting a Term

The weighted value or importance of a word t in a document d can be considered by taking both t's term frequency and inverse document frequency:

$$TF_{t,d} = \begin{cases} 1 + \log_{10} count(t,d) & \text{if } count(t,d) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$IDF_t = \log_{10} \frac{N}{DF_t}$$

$$TF - IDF = TF_{t,d} \times IDF_t$$

Beyond TF-IDF

Any problems you can imagine with TF-IDF style ?

Beyond TF-IDF

Any problems you can imagine with TF-IDF style ?

Make the vector representations with more *semantics*

- Topic Models
  - LSI, PLSI (Hoffman, 1999), ...
  - LDA (Blei et al., 2003), ...
- Neural Models
  - Word Embeddings (Word2vec, Mikolov et al., 2013, Glove, Pennington et al., 2014, ...)
  - Sentence Embeddings (Le and Mikolov, 2014, ...)
  - Document Embeddings (Le and Mikolov, 2014, ...)

The Relevance Function: $f()$

How to compute the relevance between query q and doc d, of dimention $|\text{dim}|$

- Euclidean Distance: (sensitive to length)

$$f(q,d) = 1/\sqrt{\sum_{i=1}^{|\text{dim}|}(d_i - q_i)^2}$$

- Dot Product:

$$f(q,d) = \cos(q,d) = \frac{q \cdot d}{|q||d|}$$

- Dice:

$$f(q,d) = \text{dice}(q,d) = \frac{2 \times q \cdot d}{||q||^2 + ||d||^2}$$

- Jaccard:

$$f(q,d) = \text{Jaccard}(q,d) = \frac{q \cdot d}{||q||^2 + ||d||^2 - q \cdot d}$$

Outline

1. Background
2. The Information Retrieval Task
3. **Retrieval Methods**
   - Boolean Model
   - Vector Space Model
   - **Probabilistic Model**
   - Neural Model
4. Take-away

Probabilistic Model

Use probabilistic model to characterize the probablity of q and d are relevant

- Binary Independence Model (Robertson and Jones, 1970's)
- Okapi BM25 (Robertson, 1994)

$$\text{BM25}(q,d) = \sum_{w \in q} \log(\frac{N}{df_w}) \frac{tf_{w,d}}{k(1 - b + b(\frac{|d|}{avgdl})) + tf_{w,d}}$$

  where N is the total number of docs, $|d|$ is the length of d, parameter k balances between term frequency and IDF, b controls the importance of document length normalization.

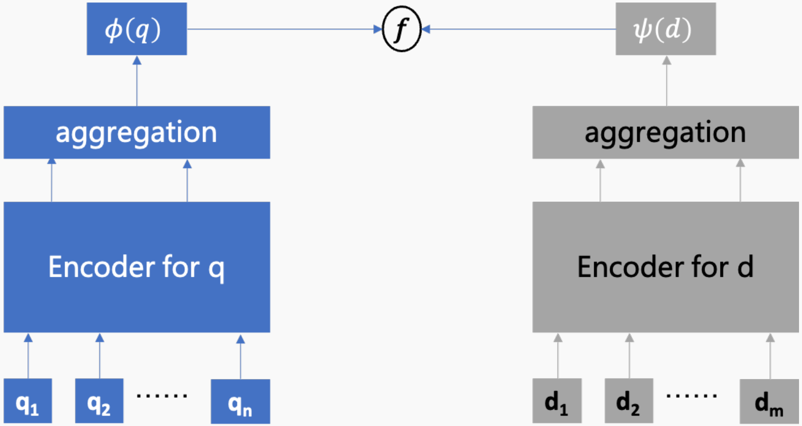
- very powerful, still widely used

Outline

Neural Model

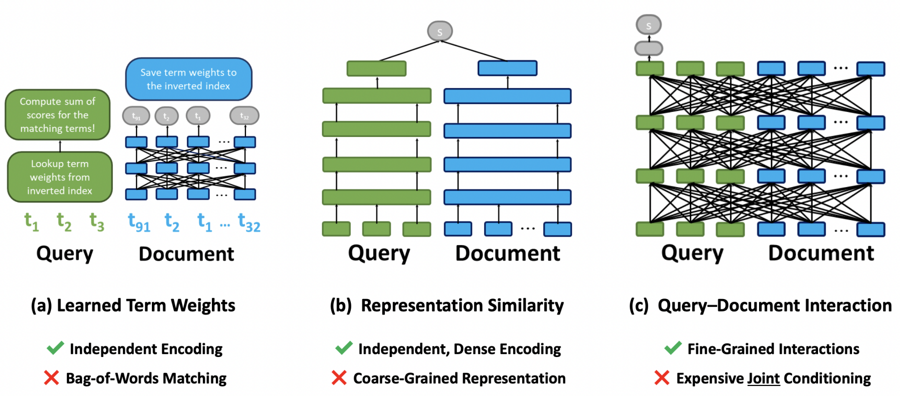
Compute the relevance through neural models.

- can use many different neural architectures
- even with pre-trained models
- but need training data!



Neural Choices

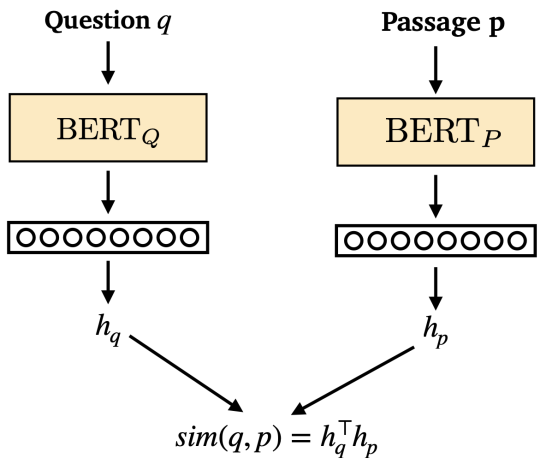
There could be many neural choices:



[Omar Khattab]

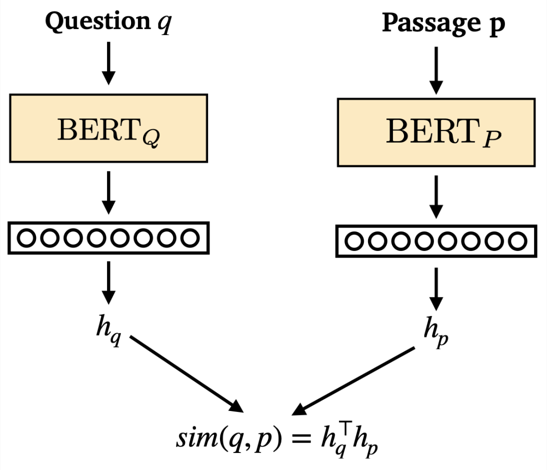
Dense Passage Retrieval (DPR)

Can we train a dense retrieval model from a small number of Q/A pairs only, without pre-training! [Karpukhin et al. 2020]



Dense Passage Retrieval (DPR)

Can we train a dense retrieval model from a small number of Q/A pairs only, without pre-training! [Karpukhin et al. 2020]



But, it is still hard to collect training data, why?

DPR: Training Examples

Positive examples

- Provided in the reading comprehension datasets
- Passages of high BM25 scores that contain the answer string

Negative examples

- Random passages from the corpus
- Passages of high BM25 scores that DO NOT contain the answer string
- Positive passages of OTHER questions

DPR: Training Examples

Positive examples

- Provided in the reading comprehension datasets
- Passages of high BM25 scores that contain the answer string

Negative examples

- Random passages from the corpus
- Passages of high BM25 scores that DO NOT contain the answer string
- Positive passages of OTHER questions

The best model uses OTHERs from the same mini-batch (in-batch negatives) and one passages from hard negatives.

DPR: In-Batch Negatives

- A small trick to effective generate more training pairs
- Suppose we have n pairs of relevant questions and passages. Let $Q_{d\times n}$ and $P_{d\times n}$ be the question and passage embeddings.
- $S = Q^T P$ is a $n \times n$ matrix of the similarity scores. Scores of n^2 pairs of questions and passages. For each question, 1 positive passage and $n-1$ negative passages

Dense Passage Retrieval

- No need of expensive large scale training?
  - At least for modest-sized QA datasets, but limited language coverage
- Using Reading comprehension vs QA datasets as training
  - Doesn’t make a much difference (41.5 vs 41.0 on NQ).
  - We can train the system using only Q/A pairs!
- BM25 + DPR is slightly better than DPR but the difference is small.

Outline

Take-away

- BM 25: Best Matching 25
- Neural Models: Dense passage retrieval
- Many Matching Models available now, but none of them is perfect

Readings

- Chapter 14. Speech and Language Processing: `https://web.stanford.edu/~jurafsky/slp3/14.pdf`
- Vladimir Karpukhin, Barlas Ouz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, Wen-tau Yih, Dense Passage Retrieval for Open-Domain Question Answering, EMNLP 2020