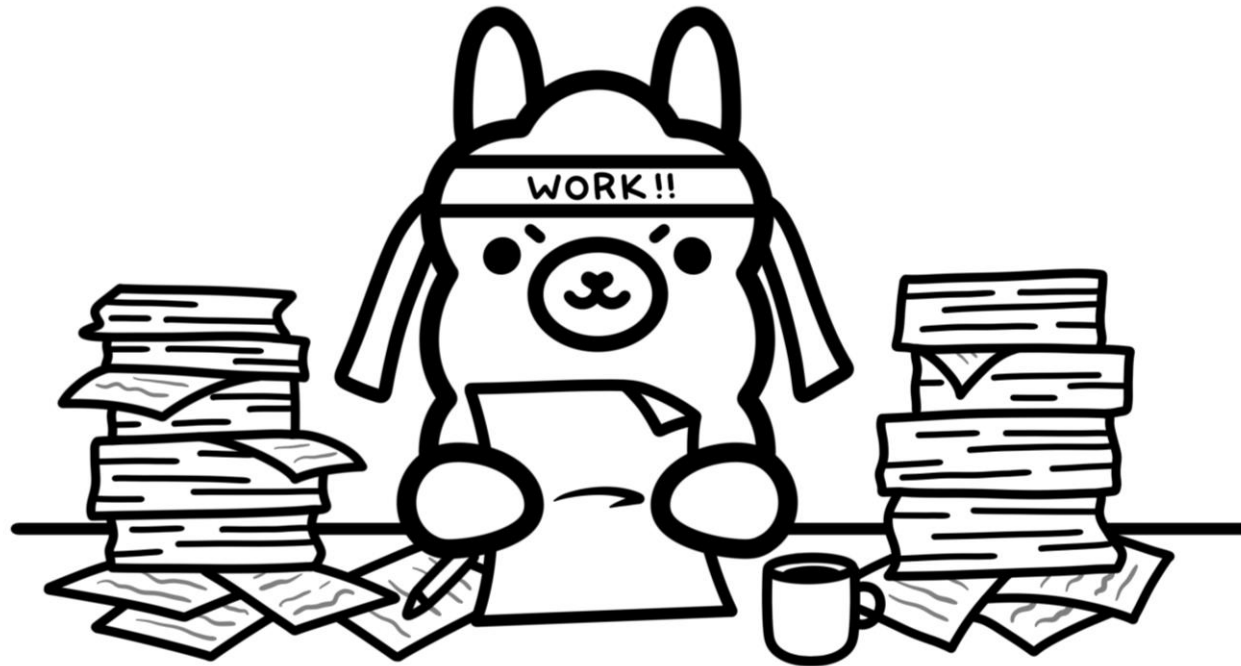


# Neural information retrieval



Prof Dr Marko Robnik-Šikonja

Natural Language Processing, Edition 2024

# Obtaining relevant context for a query

- a part of traditional information retrieval
- But still relevant even for LLMs
- The context can constitute a part of the prompt to LLM
- Well-known approaches
  - BM25 (Best match 25)
  - DPR (Dense Passage Retrieval)
  - Dot product on sentence encoders, e.g., LaBSE
  - CovBERT

# Ranking documents with BM25

- Okapi BM25 (Best match 25)
- uses bag-of-words document representation, works similarly to tf-idf weighting
- Given a query  $Q$ , with words  $q_1, \dots, q_n$  the BM25 score of a document  $D$  is:

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

- $f(q_i, D)$  is the number of times that  $q_i$  occurs in  $D$ ,
- avgdl is the average document length in the text collection
- $k_1$  and  $b$  are parameters, usually chosen from  $k_1 \in [1.2, 2.0]$  and  $b = 0.75$

## IDF variant

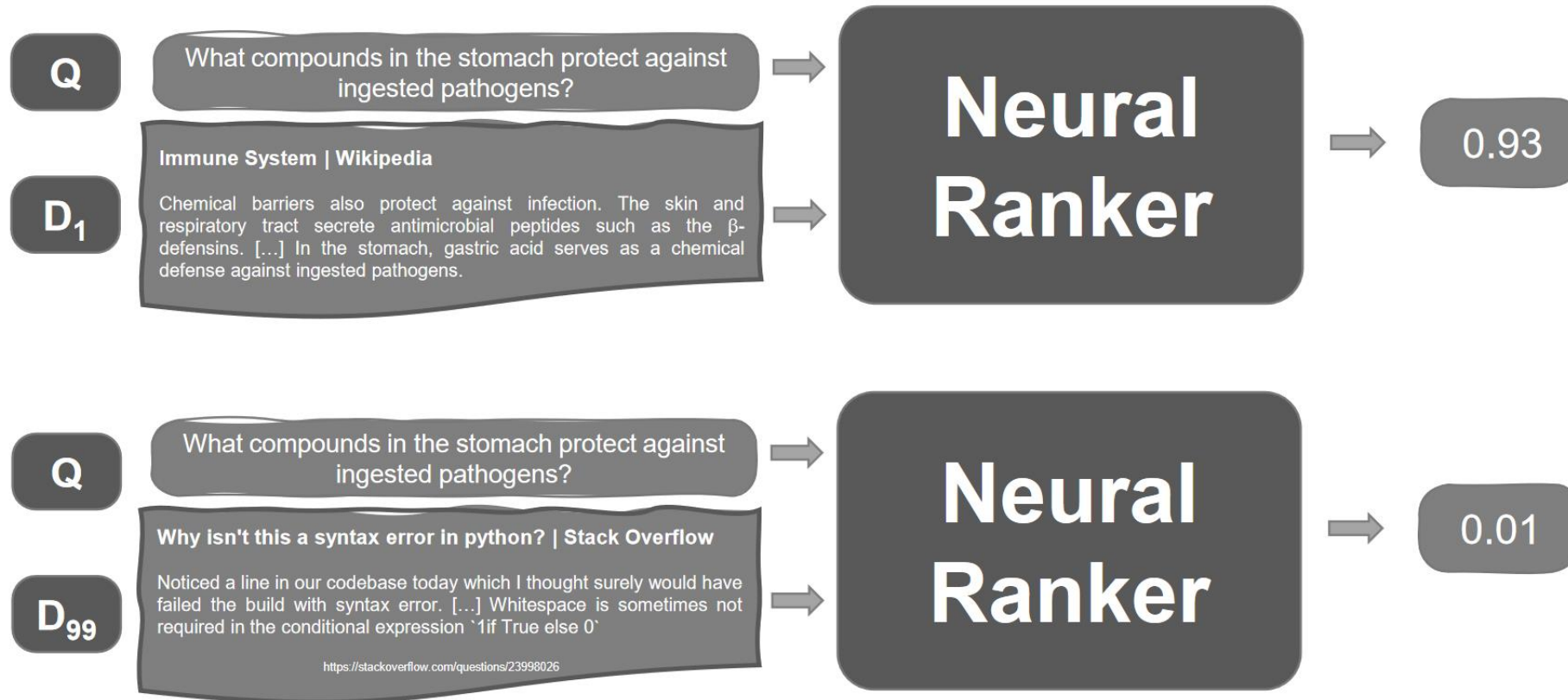
- IDF (inverse document frequency) weights the query term  $q_i$

$$\text{IDF}(q_i) = \ln \left( \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1 \right)$$

- where  $N$  is the total number of documents in the collection, and  $n(q_i)$  is the number of documents containing  $q_i$

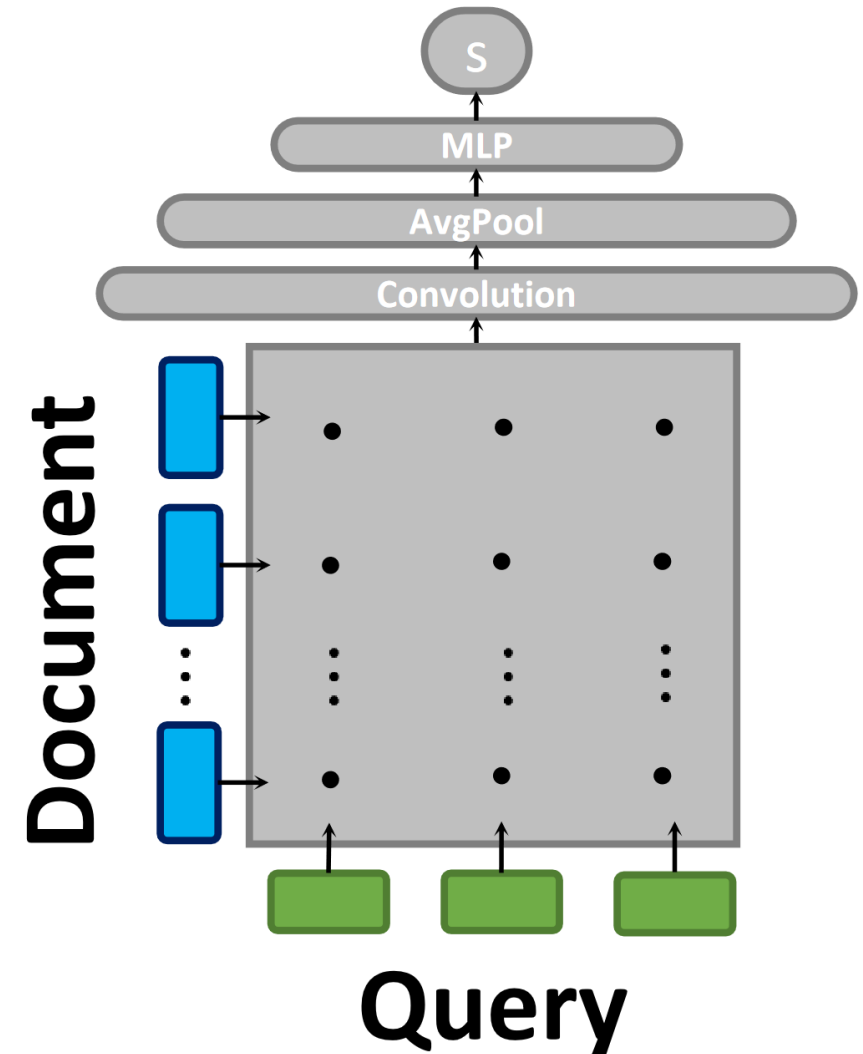
# Neural Ranking

- Use neural representations



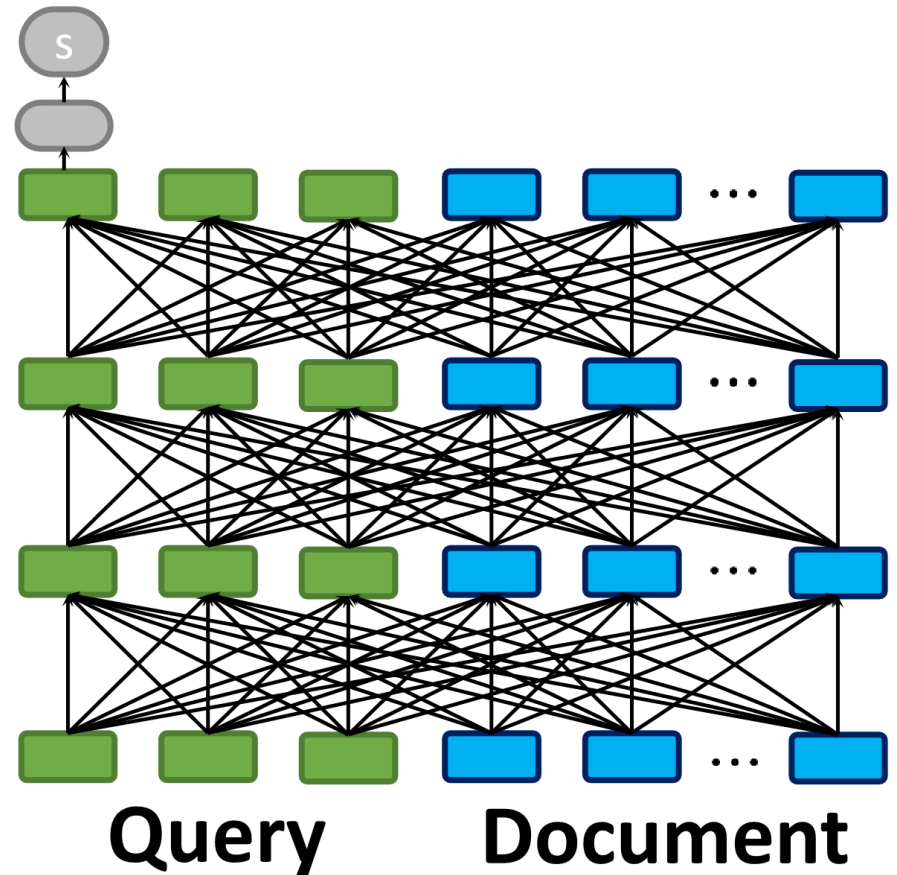
# Query-Document Interaction Approach

- IR Ranking refers to scoring query-document pairs, sorting them in descending order, and then getting the top K results:
  - Tokenize query and documents
  - Embed tokens to vector
  - Make query-document interaction matrix and compute cosine similarity for each pair of words.
  - Compress the matrix into a score. Use a neural layer (convolution, linear layers)
- considerably better than non-neural methods but computationally expensive
- why?



# All-to-all Interaction with BERT

- 1. Feed BERT “[CLS] Query [SEP] Document [SEP]”
  - 2. Run this through all the BERT layers
  - 3. Extract the final [CLS] output embedding
  - 4. Reduce to a single score through a linear layer
- 
- This is essentially a standard BERT classifier, used for ranking passages.
  - We must fine-tune BERT for this task with positives and negatives to be effective
  - Much better quality—but also a dramatic increase in computational cost
  - How to get a better query latency?



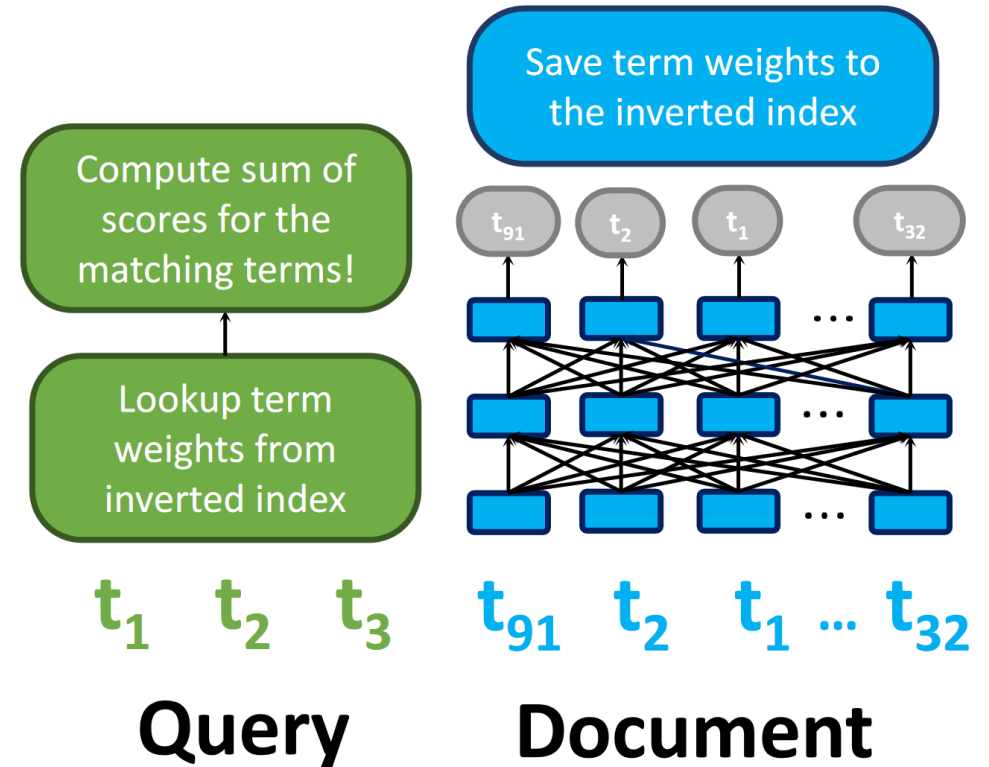
# Faster IR: precomputing

- Is there a value in jointly representing queries and documents?
- BERT rankers are slow because their computations can be redundant:
  - Represent the query (1000 times for 1000 documents)
  - Represent the document (once for every query!)
  - Conduct matching between the query and the document
- We have the documents in advance.
  - Can we pre-compute the document representations?
  - And “cache” these representations for use across queries



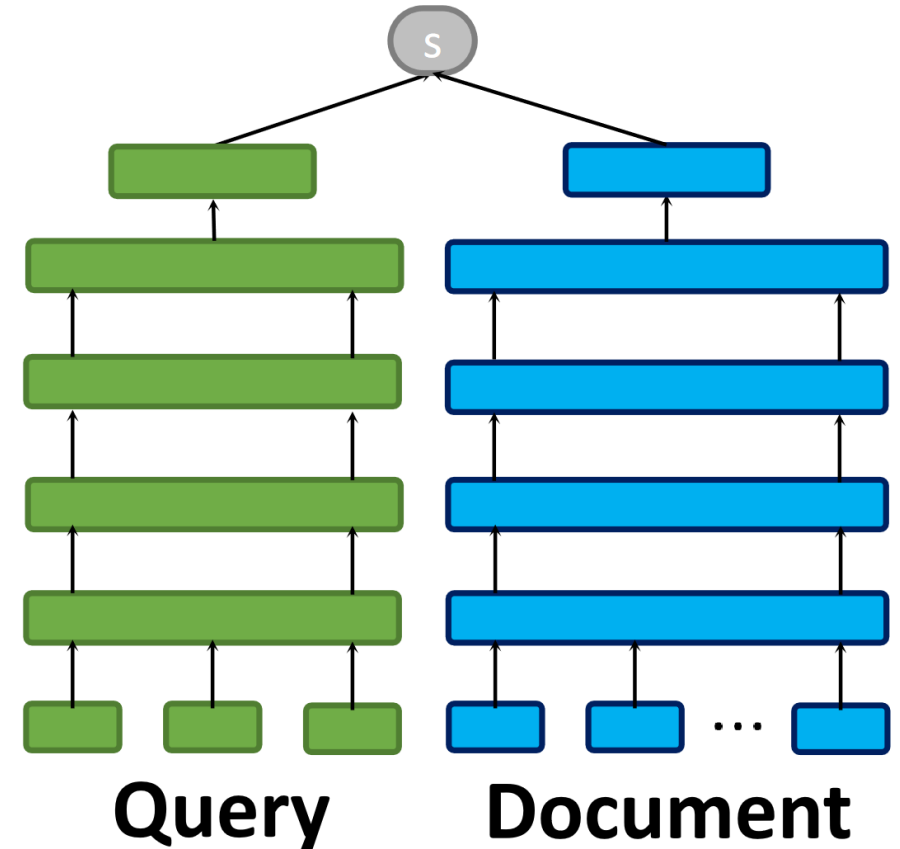
# A bad solution: Neural bag-of-words

- BM25 decomposed a document's score into a summation over term-document weights. Can we learn term weights with BERT?
- Tokenize the query/document
- Use BERT to produce a score for each token in the document
- Add the scores of the tokens that also appear in the query



# Neural IR: Representation similarity

- Tokenize the query and the document
- Independently encode the query and the document into a single-vector representation each
- Estimate relevance as a dot product or a cosine similarity
- Like learning term weights, this paradigm offers strong efficiency advantages:
  - Document representations can be pre-computed!
  - Query computations can be amortized.
  - Similarity computations are very cheap.



# Example of representation similarity: Dense Passage Retrieval (DPR)

- BERT based passage retrieval
- Encodes each passage and each query into a 768-dimensional vector
- ranks passages in the document collection relative to query  $q$  using dot product similarity
- BERT is additionally pretrained to maximize the similarity between  $q$  and correct passages and minimize the similarity between  $q$  and wrong passages using the loss:

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

- A negative passage is sampled from BM25 top-100
- passages and query are encoded with modified BERT (using the CLS token representation) then ranked based on the dot product similarity

Karpukhin et al (2020) Dense passage retrieval for open-domain question answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6769–6781.

# LaBSE sentence encoder

- LaBSE (Language-agnostic BERT Sentence Encoder)
- dual-encoder architecture, where source and target sentences (in different languages) are encoded separately using a shared BERT-based encoder
- pre-trained on masked language modeling and translated language modeling
- supports 109 languages
- allows finding similar sentences across different languages.
- loss

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\phi(x_i, y_i)}}{e^{\phi(x_i, y_i)} + \sum_{n=1, n \neq i}^N e^{\phi(x_i, y_n)}}$$

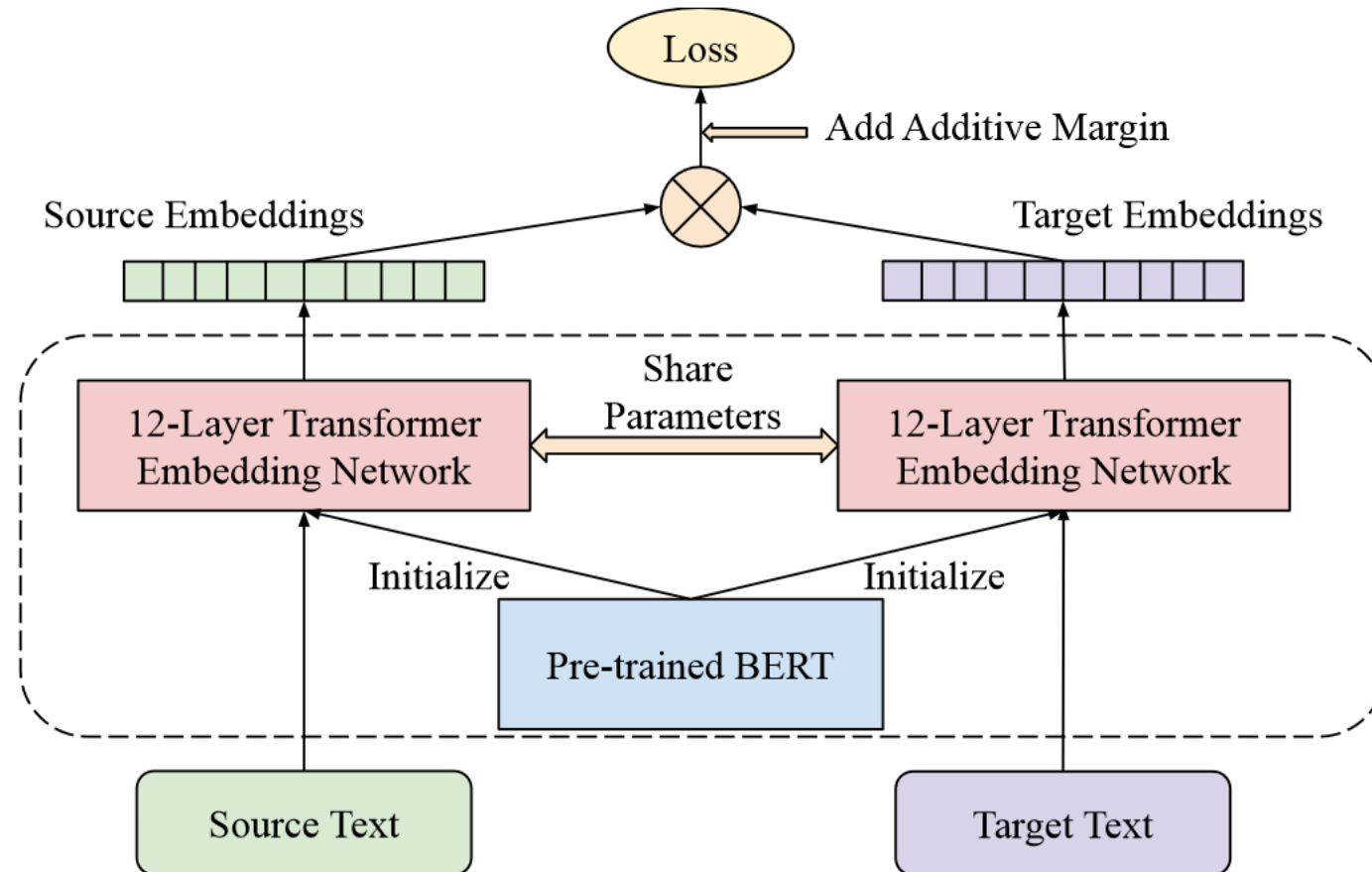
Feng, F., Yang, Y., Cer, D., Arivazhagan, N. and Wang, W., 2022. Language-agnostic BERT Sentence Embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 878-891).

<https://arxiv.org/abs/2007.01852>

<https://tfhub.dev/google/LaBSE>

# LaBSE architecture

- Dual encoder model with BERT based encoding modules.

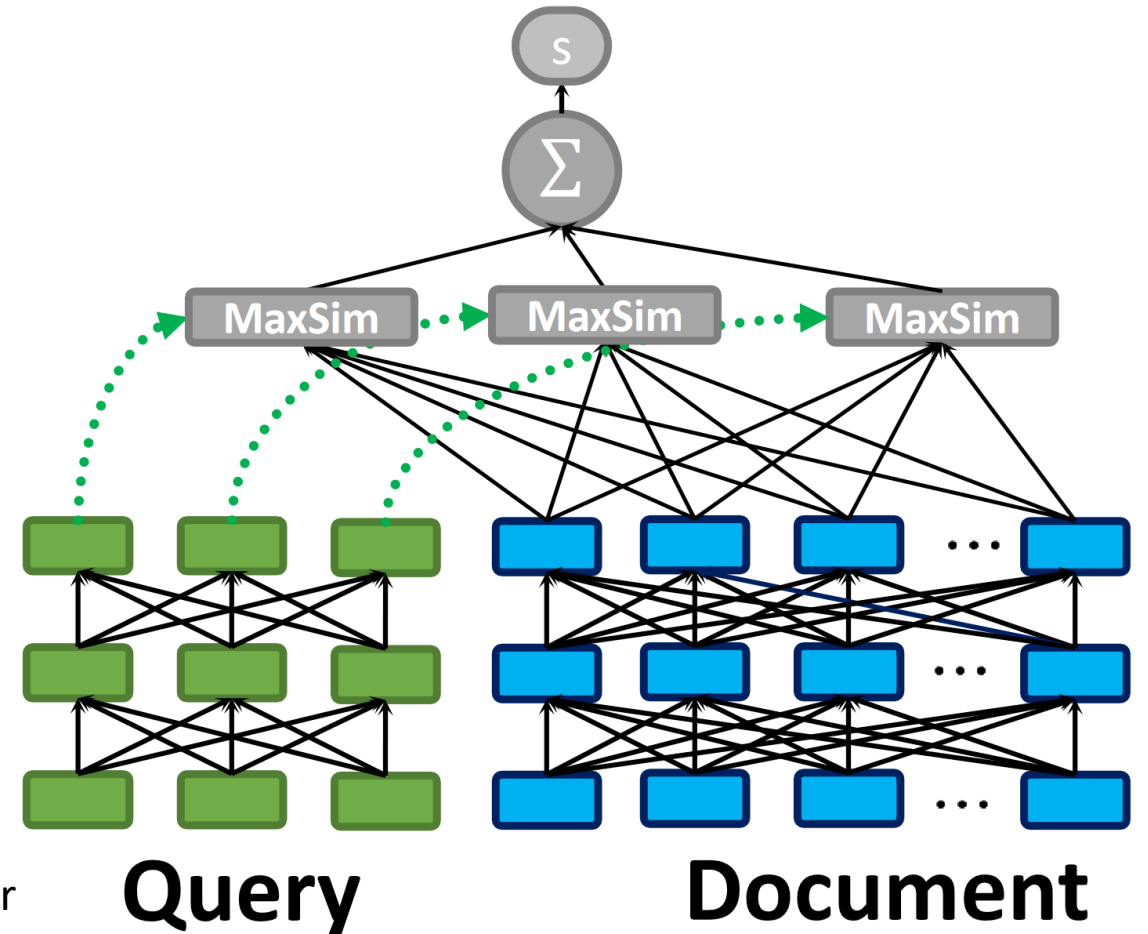


# Representation Similarity: Downsides

- Single-Vector Representations “cram” queries and documents into a coarse-grained representation!
- No fine-grained interactions
- They estimate relevance as single dot product!
- We lose term-level interactions, which we had in query–document interaction models (e.g., BERT) and even term-weighting models (e.g., BM25)
- Can we keep precomputation and still have fine-grained interactions?

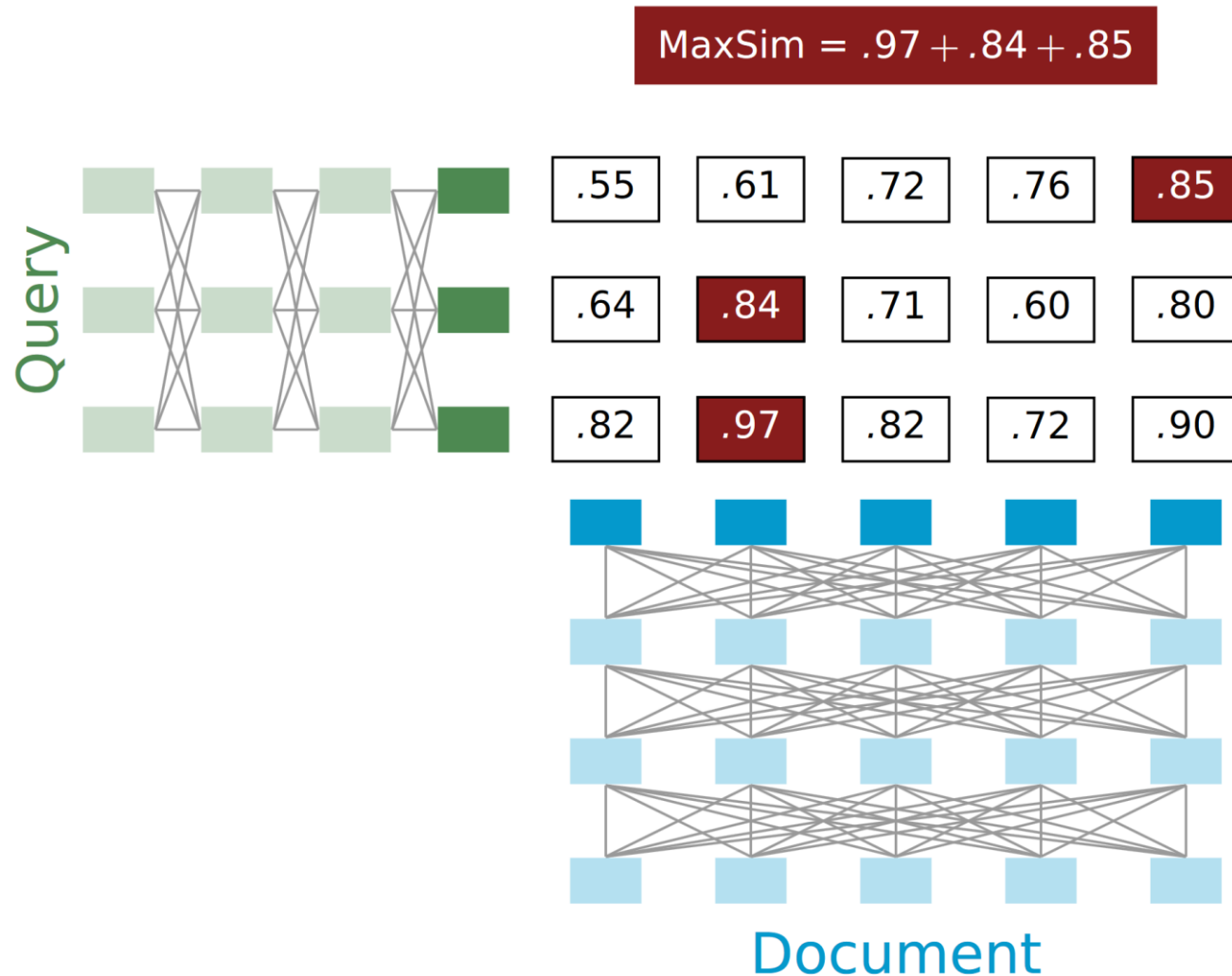
# Neural IR: Late interactions

- Independent Encoding
- Fine-Grained Representations
- End-to-End Retrieval
- ColBERT represents the document as a MATRIX, not a vector



Omar Khattab and Matei Zaharia. "ColBERT: Efficient and effective passage search via contextualized late interaction over BERT." SIGIR'20

# ColBERT: MaxSim



1. Examples:  $\langle q_i, doc_i^+, \{doc_{i,k}^-\} \rangle$
2. Loss: negative log-likelihood of the positive passage, with **MaxSim** as the basis.

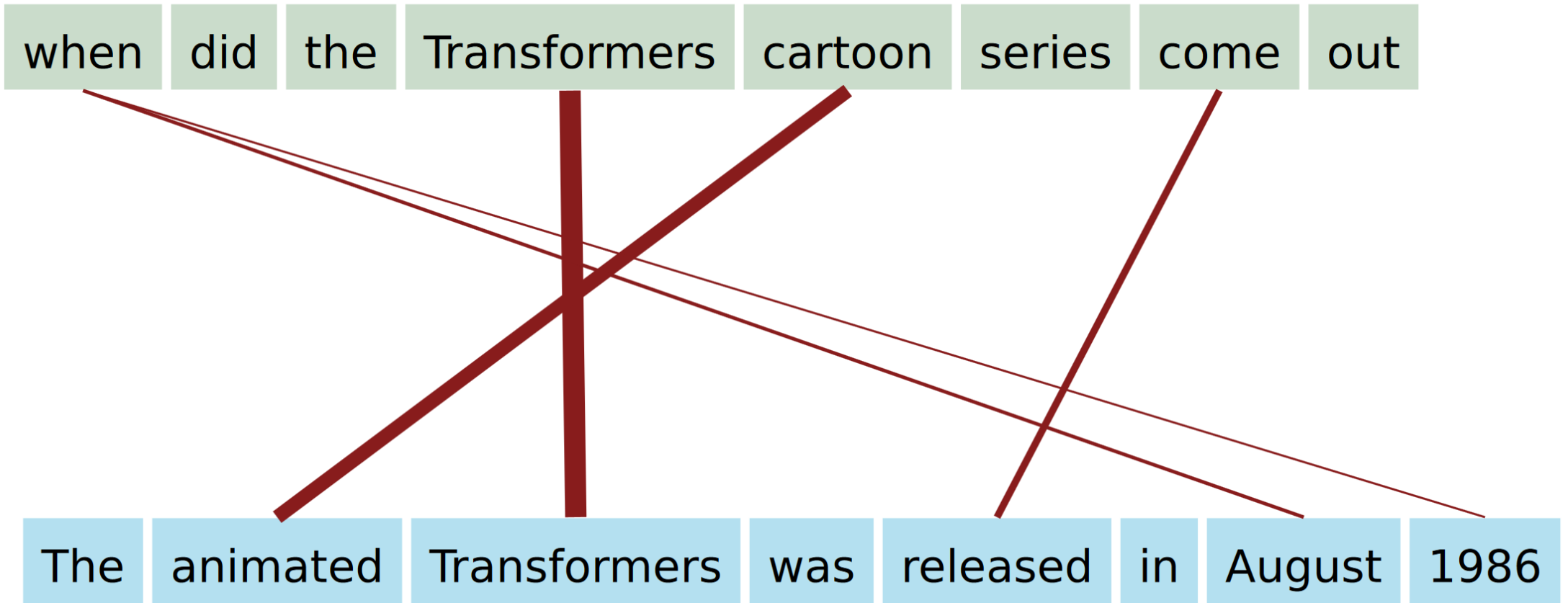
For a BERT-style encoder with  $N$  layers:

$$\mathbf{MaxSim}(q, doc) = \sum_i^L \max_j^M \mathbf{Enc}(q)_{N,i}^T \mathbf{Enc}(doc)_{N,j}$$

with  $L$  is the length of  $q$ ,  $M$  the length of  $doc$ .



# Soft alignment with CoBERT



# Common Evaluation Metrics

1. *Accuracy* (does answer match gold-labeled answer?)

2. *Mean Reciprocal Rank*

– For each query return a ranked list of  $M$  candidate answers.

– Query score is  $1/\text{Rank}$  of the first correct answer

- *If first answer is correct: 1*
- *else if second answer is correct:  $\frac{1}{2}$*
- *else if third answer is correct:  $\frac{1}{3}$ , etc.*
- *Score is 0 if none of the  $M$  answers are correct*

– Take the mean over all  $N$  queries

$$MRR = \frac{\sum_{i=1}^N \frac{1}{rank_i}}{N}$$

# IR evaluation datasets

- Text REtrieval Conference (TREC) has annual competitions for comparing IR systems.
- MS MARCO Ranking is the largest public IR benchmark.
  - It is adapted from a Question Answering dataset
  - It consists of more than 500k Bing search queries
  - Passage Ranking: 9M short passages; sparse labels
  - Document Ranking: 3M long documents; sparse labels
- Many others