

# Denoising Dense Representations with Symbols for Robust Zero-Shot Retrieval

Rodrigo Nogueira



Maritaca AI



# Tutorial Timetable

1. Part 1: Knowledge Graphs and Entities
  1. Welcome & Latent Space Representations (Dietz)
  2. Knowledge Graphs and GPT (Bast)
  3. Entity Linking (Bast)
2. Part 2: Neuro-Symbolic Foundations
  1. Ranking Wikipedia Entities / Aspects (Chatterjee)
  2. Neural Text Representations and Semantic Annotations (Dietz)
  3. Infusion of Symbolic Knowledge into Text Representation (Nie)
3. Part 3: Reasoning, Robustness, and Relevance
  1. Denoising Dense Representations with Symbols (Nogueira) ← **We are here**
  2. Reasoning about Relevance (Dalton)
  3. From PRF to Retrieval Enhanced Generation (Dietz)
4. Part 4: Emerging Topics
  1. Conclusion and Outlook
  2. Panel Discussion

# Agenda

- Motivation: The need for better search engines in the context of LLMs
- 3 ways of using Transformers for search:
  - Dense Retrievers
  - Sparse Retrievers
  - Rerankers
- In-domain vs out-of-domain analysis
- Scaling up model size

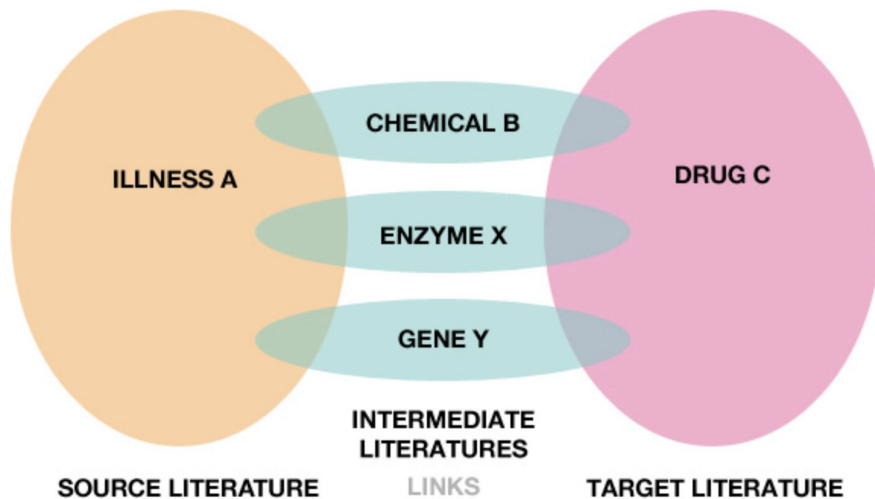
# Why am I interested in information retrieval?

*To connect pieces of knowledge previously thought to be unrelated*

E.g., to build an automated **hypothesis generator** using **literature-based discovery**

Example:

*What are the best drugs for illness A?*



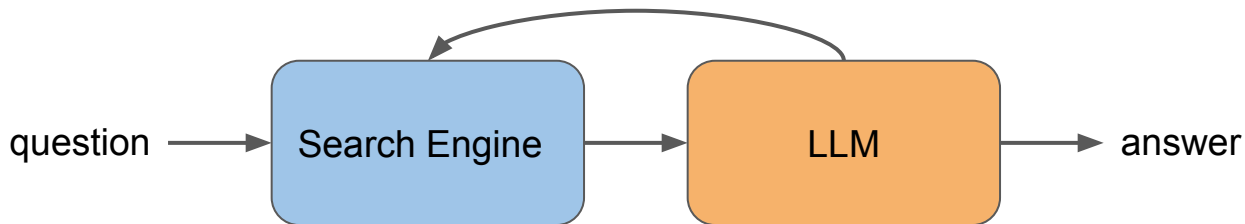
[Swanson linking](#)

# Just ask GPT-4 to generate hypotheses?

Maybe, but it is yet to be shown that this works...

- Current LLM's can hardly remember facts about long-tail topics, let alone connect them

For now, we need to use a search engine to give a LLM the necessary information to link pieces of knowledge



# Information given to LLMs needs to be carefully selected

A example from the IIRC dataset:

Wilhelm Müller was born on 7 October 1794 at **Dessau**, the son of a tailor. In 1813-1814 he took part, as a volunteer in the Prussian army, in the national rising against **Napoleon**. He participated in the battles of **Lützen**, **Bautzen**, **Hanau** and **Kulm**. In 1814 he returned to his studies at Berlin. Müller's son, **Friedrich Max Müller**, was an English orientalist who founded the comparative study of religions.

**Which battle Wilhelm Müller fought in while in the Prussian army had the most casualties?**

## **Battle of Lützen (1813)**

Napoleon lost 19,655 men, while the Prussians lost 8,500 men and the Russians lost 3,500 men

## **Battle of Bautzen**

Losses on both sides totaled around 20,000.

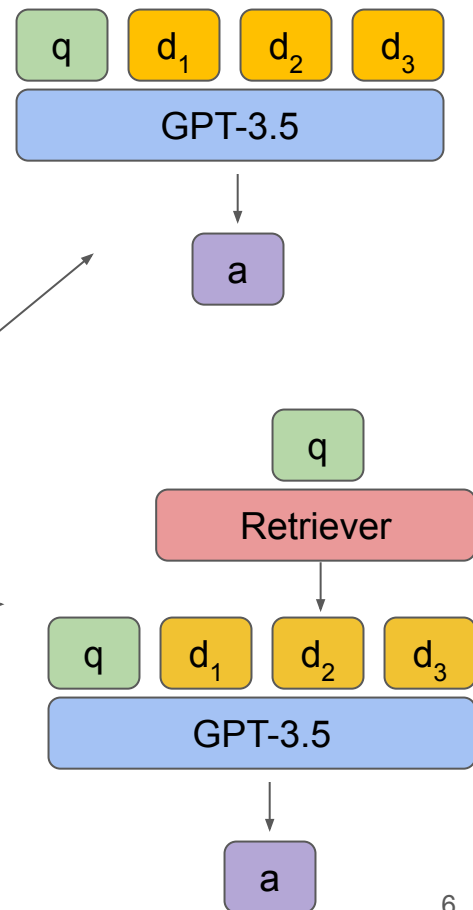
## **Battle of Hanau**

Overall, 4,500 French soldiers and 9,000 allied soldiers were lost in the battle.

## **Battle of Kulm**

The French lost more than half of the pursuing force of 34,000; The allies lost approximately 13,000 soldiers.

IIRC dataset	F1
Human	88.4
GPT-3.5 + Gold Evidence	84.2
GPT-3.5 + SOTA Retriever	50.6

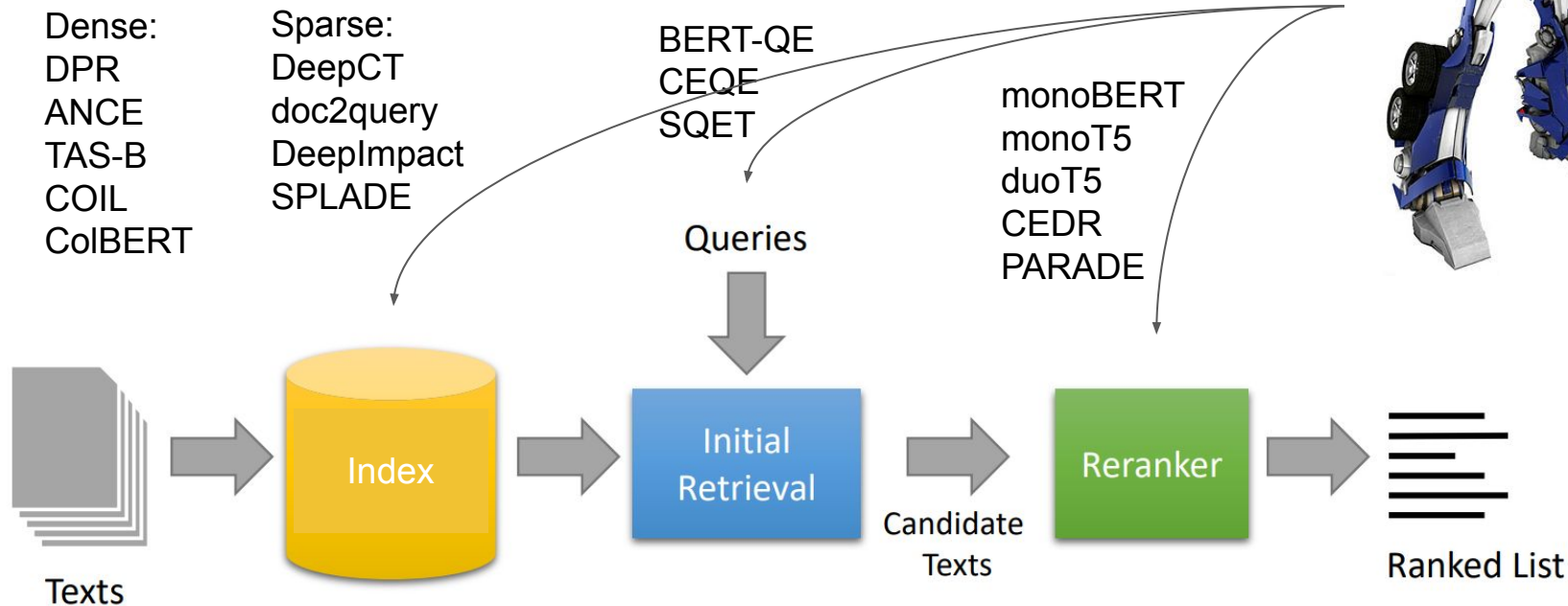


# Just train retrievers on more data?

- Many interesting problems do not have labelled datasets
  - Ex: domain-specific tasks, long-tail knowledge
- We need robust zero-shot retrievers!

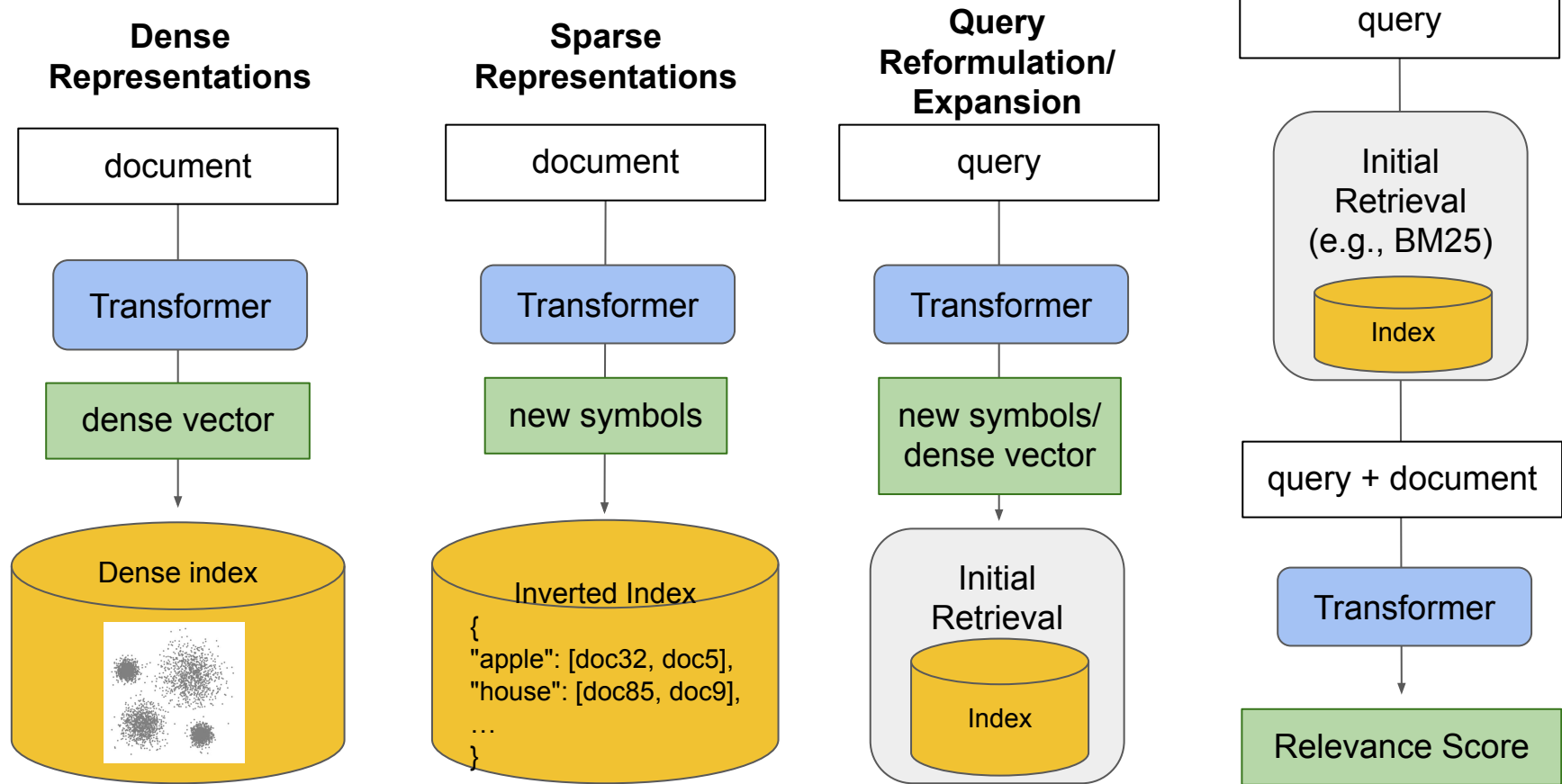
# A Simple Search Engine

Where to use a Pretrained Transformer Model?



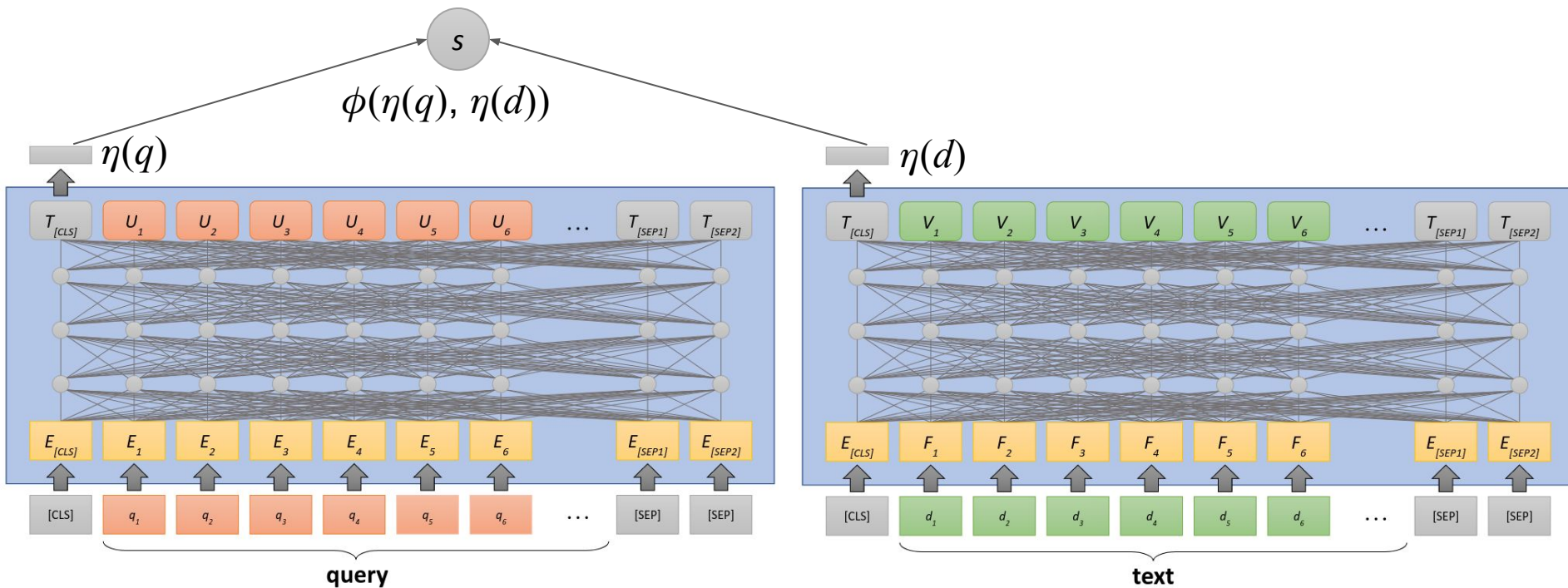


# How Transformers are used in IR?



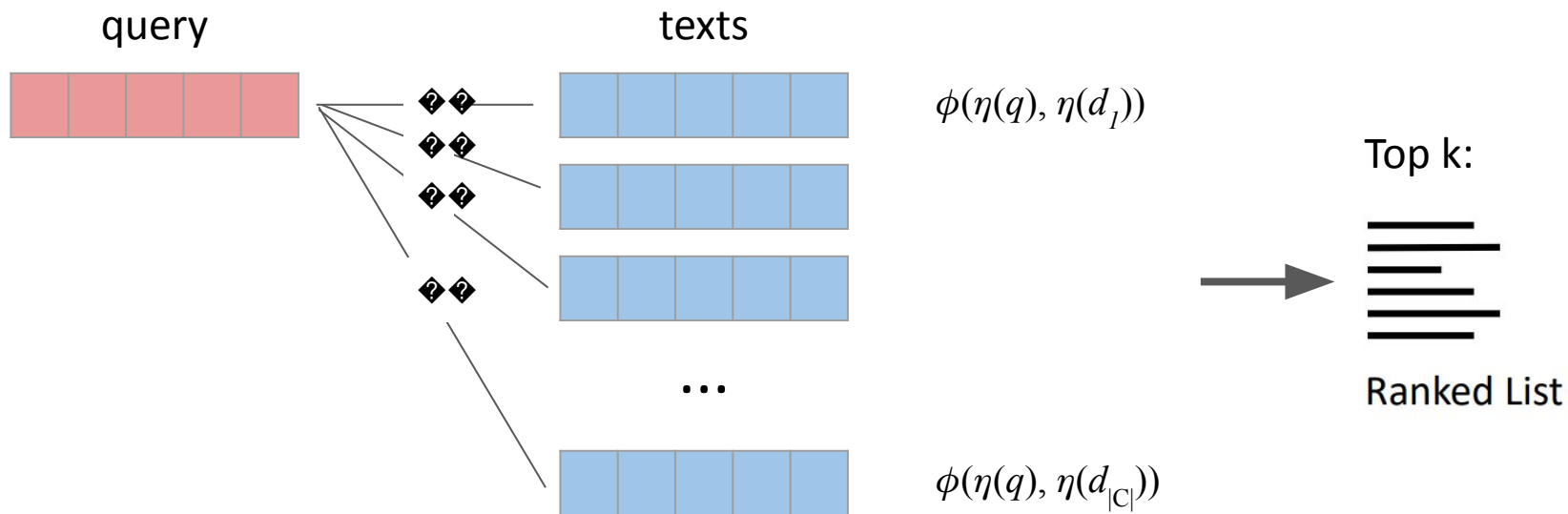
# Dense representations

# Single-vector dense retriever based on BERT



# Retrieval: Find the top k most relevant texts to a query

Brute-force search:

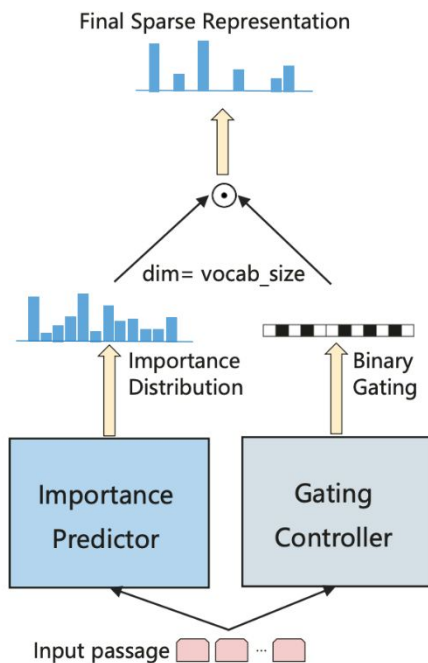


We often need to search many (e.g.: millions) of texts

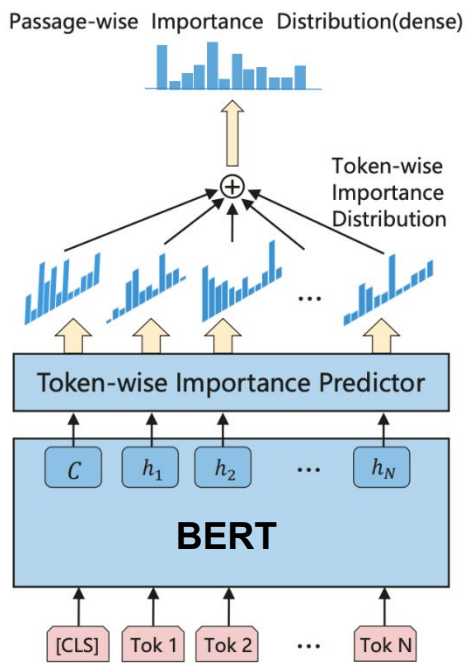
- Brute-force won't scale, Approximate Nearest Neighbor methods are commonly used

# Learned Sparse Representations

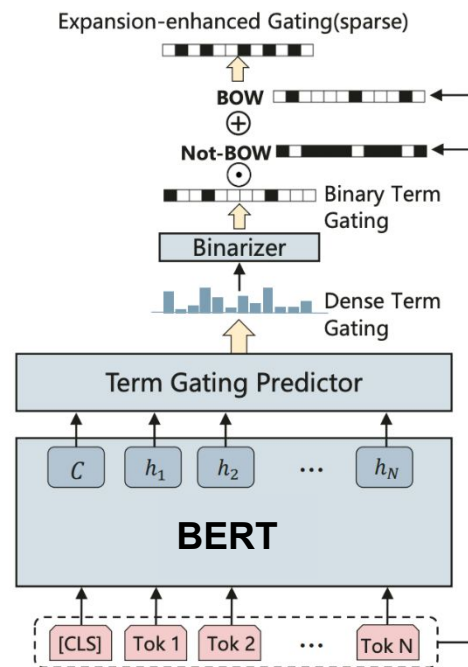
# Learned Sparse Representations: SparTerm



(a) SparTerm Model



(b) Importance Predictor



(c) Gating Controller

# SPLADE: Learned Sparse Representations

$|w|$  = embedding dim = vocabulary size

$w_j = \log(1 + \text{ReLU}(\text{logits}))$  ensures sparsity (i.e., most elements are zeros)

Can be used with existing inverted index infrastructure (e.g., Lucene)

Training loss:

$$\mathcal{L} = \mathcal{L}_{\text{rank-IBN}} + \lambda_q \mathcal{L}_{\text{reg}}^q + \lambda_d \mathcal{L}_{\text{reg}}^d$$

dot product of  $q_i$  and  $d_i$

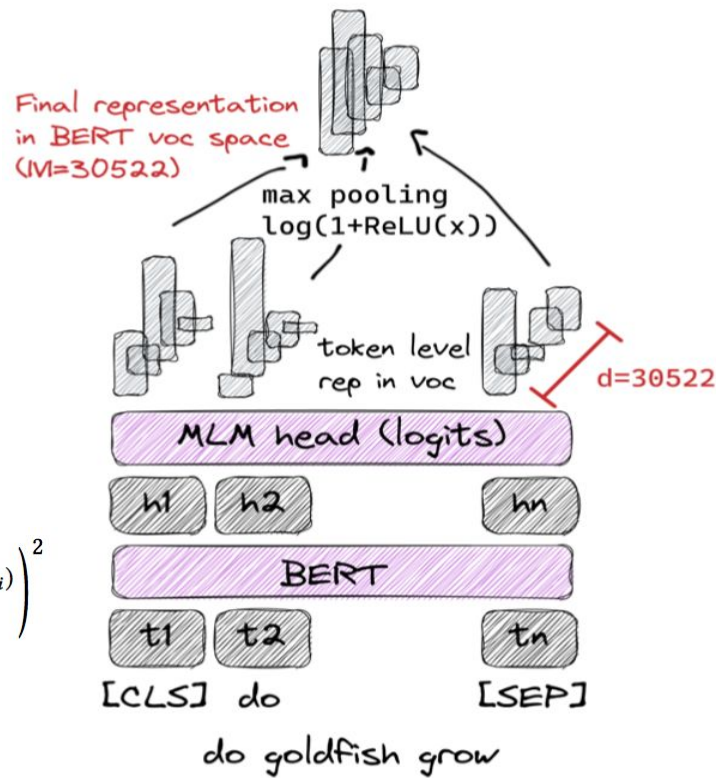
$$\mathcal{L}_{\text{rank-IBN}} = -\log \frac{e^{s(q_i, d_i^+)}}{e^{s(q_i, d_i^+)} + e^{s(q_i, d_i^-)} + \sum_j e^{s(q_i, d_{i,j}^-)}}$$

negative from BM25

in-batch negatives

$$\ell_{\text{FLOPS}} = \sum_{j \in V} \bar{a}_j^2 = \sum_{j \in V} \left( \frac{1}{N} \sum_{i=1}^N w_j^{(d_i)} \right)^2$$

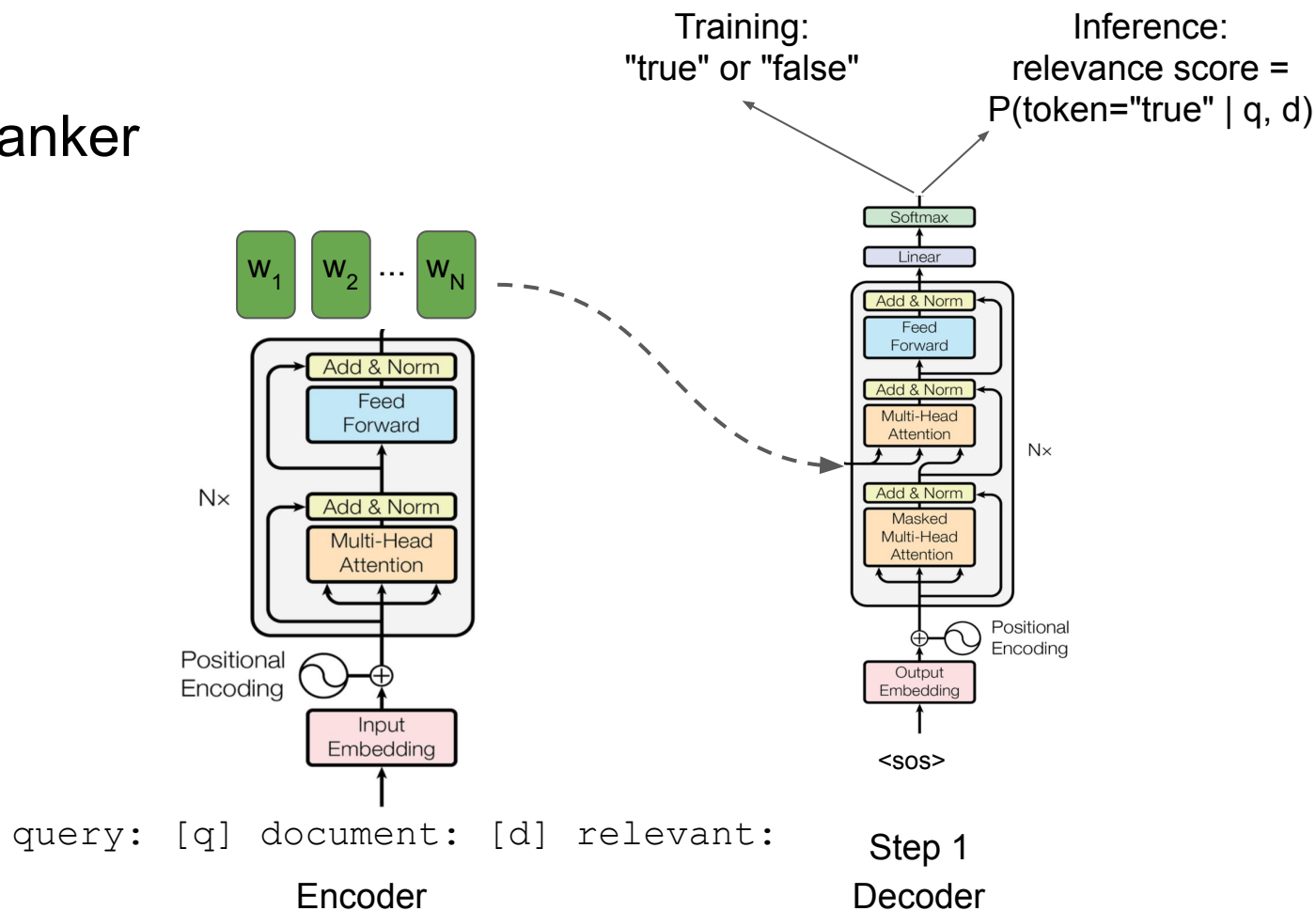
ensures low term weights



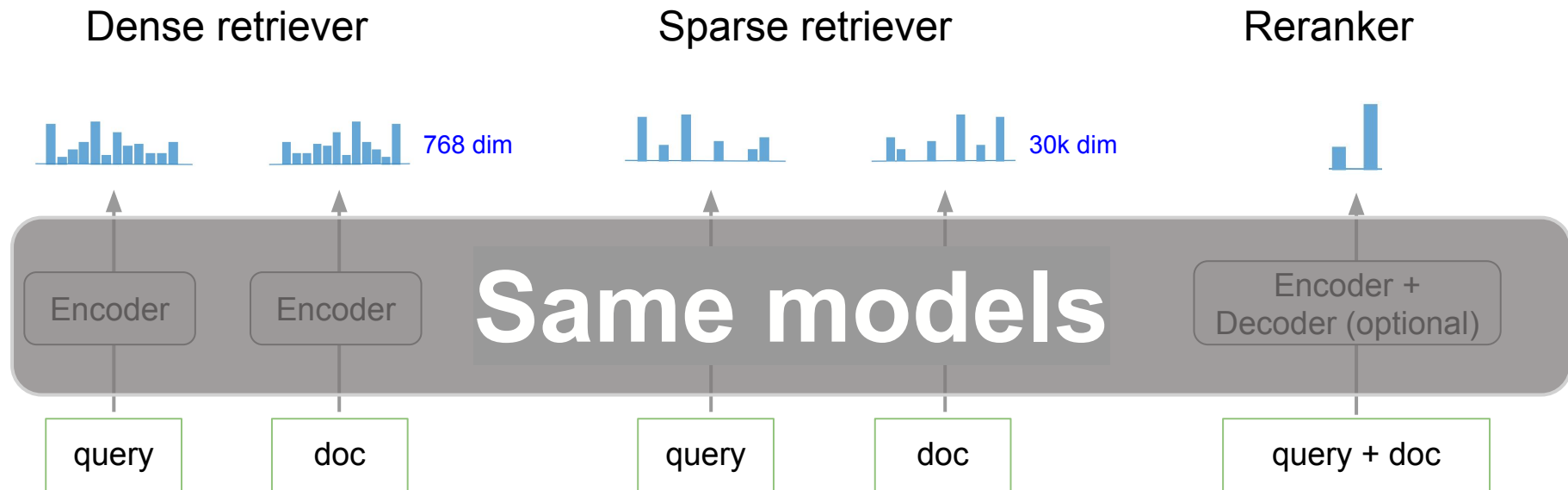
# Rerankers



# monoT5: T5 as a reranker



# In summary



Which is better in in-domain and out-of-domain IR tasks?

# What is in-domain and out-of-domain?

*It is a subjective definition!*

**"In-domain"** refers to queries and documents used in evaluation being from the same domain as training examples

**"Out-of-domain"** refers to training and test examples being from different domains (e.g., train on finance examples and evaluate on biomedicine examples)

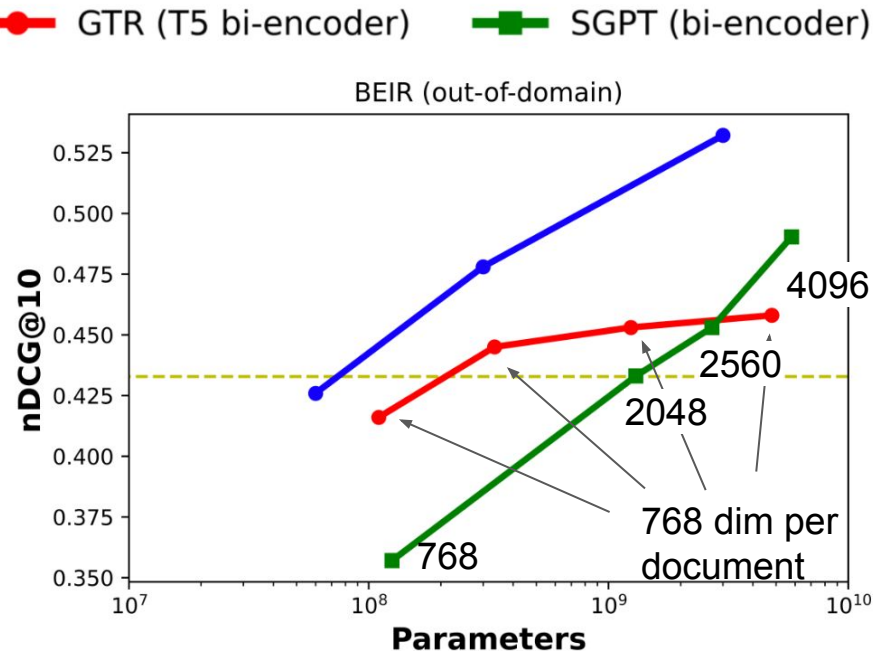
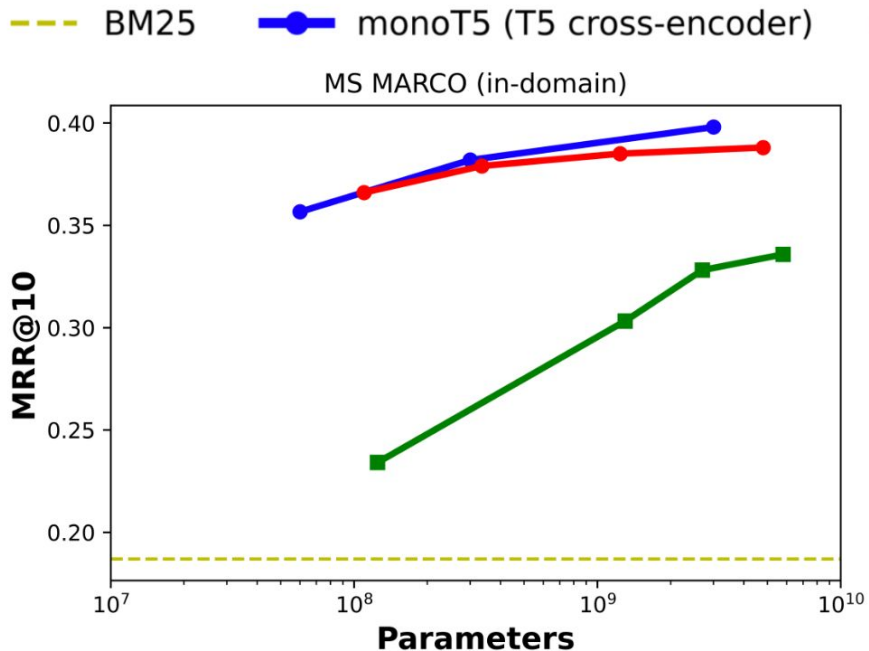
# In-domain vs Out-of-domain

No distillation (e.g., Splade v2, ColBERT v2)  
No IR-specific pretraining (e.g. CoCondenser, Contriever)

Method	nDCG@10	
	In-domain TREC-DL 20	Out-of-domain BEIR
BM25	0.475	0.440
SPLADE (sparse)	0.671	0.458
GTR-base (T5, dense)	0.696	0.426
BM25 + monoT5-base (reranker)	0.701	0.478

Models were trained on MS MARCO  
(same distribution of TREC-DL 2020)

# Scaling up model size



GTR is from Ni et al., "Large Dual Encoders Are Generalizable Retrievers", 2021  
SGPT is from Muennighoff, "SGPT: GPT Sentence Embeddings for Semantic Search", 2022  
Charts from Rosa et al., "In Defense of Cross-Encoders for Zero-Shot Retrieval", 2022

# Conclusions

- Retrieval method: No clear winner if you have lots of query-relevant passages to train on;
- Fine-grained representations (i.e., symbols) are key for out-of-domain effectiveness;
- Cross-encoders in the form of rerankers show better OOD effectiveness than bi-encoders in the form of dense retrievers

# Extras

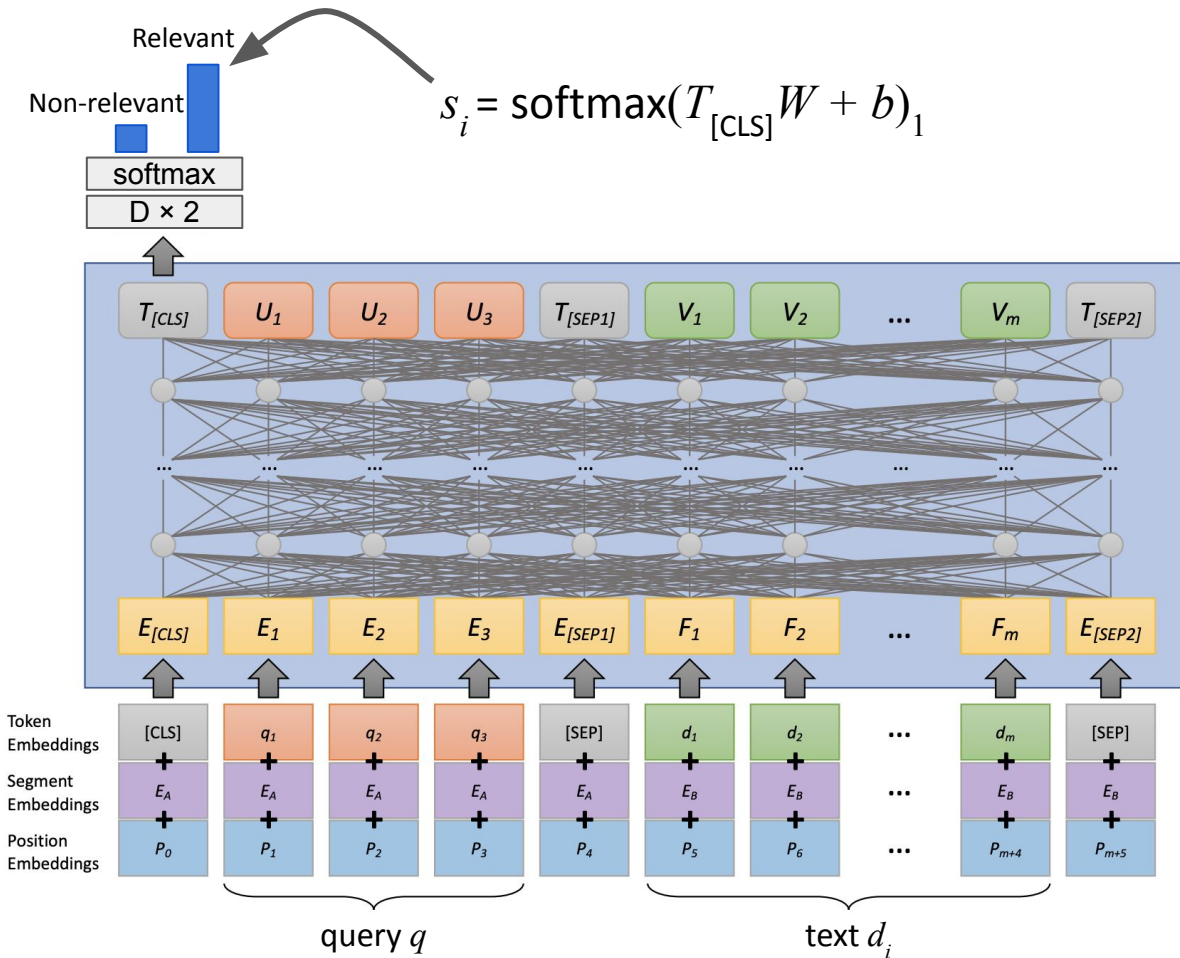


# monoBERT: BERT as a reranker

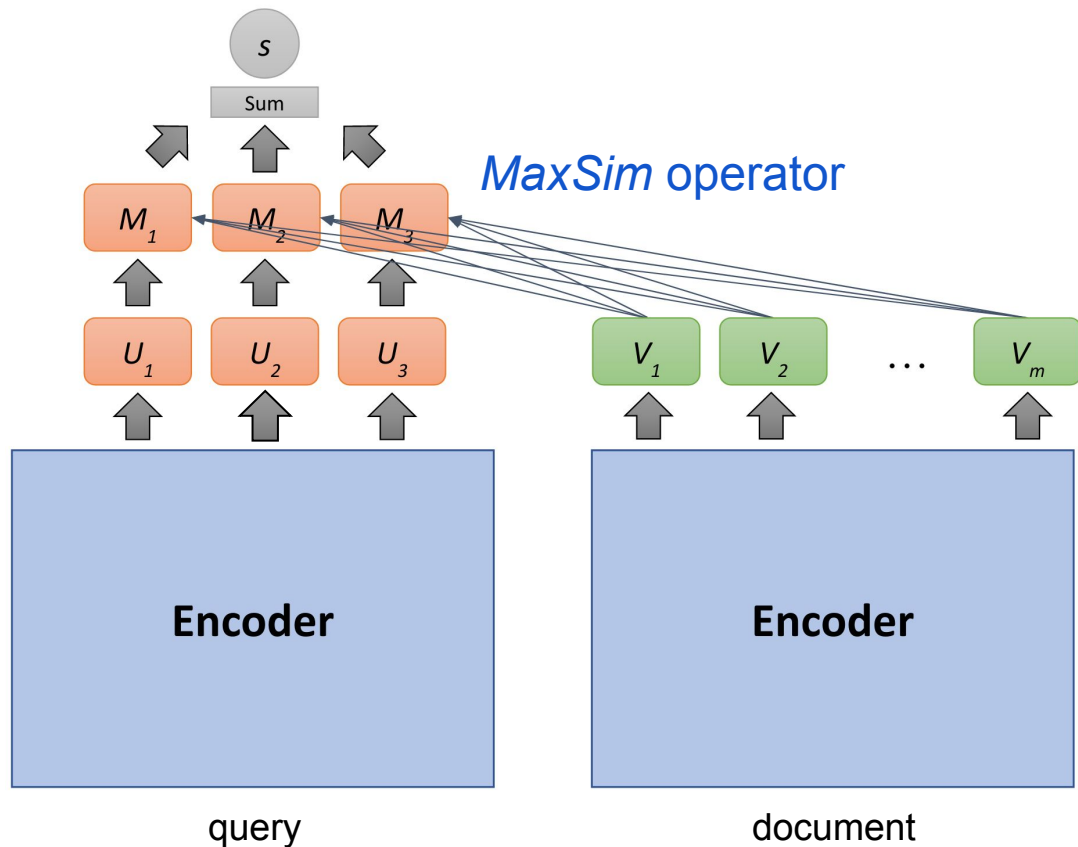
We want:

$$s_i = P(\text{Relevant} = 1 | q, d_i)$$

A binary classifier  
finetuned on pairs of  
<query, relevant text>  
and <query, non-relevant text>



# Multi-vector Dense Retriever: ColBERT



# Augmenting Sparse Representations with doc2query

# doc2query

In practice: 5-40 queries are sampled with top-k or nucleus sampling

Input: Document

TV Mounts 32-in to 55-in Fixed Wall TV Mount Fits TVs Up To 55-in (Hardware Included) Fixed TV wall mount securely holds TVs in place Supports VESA sizes (in mm) from 200 x 200 to 400 x 400 The mount supports TVs from 32-in to 55-in

doc2query

Output: Predicted Query

tv mount 55 inch

+

Concatenate

Expanded Document:

TV Mounts 32-in to 55-in Fixed Wall TV Mount Fits TVs Up To 55-in (Hardware Included) Fixed TV wall mount securely...

tv mount 55 inch

Index

Offline

Online

User's Query

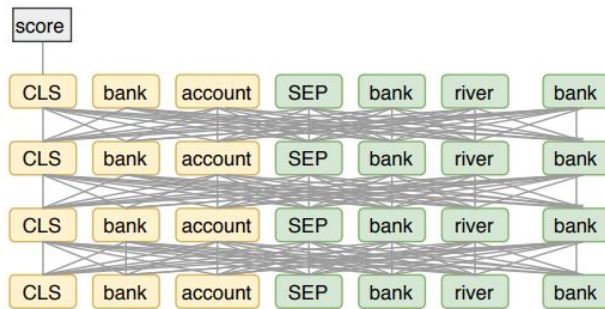
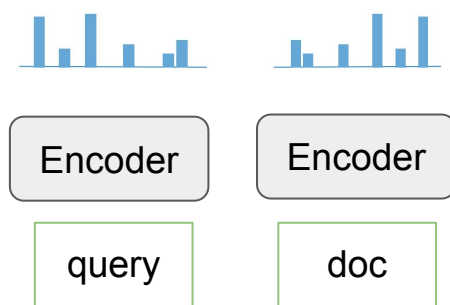
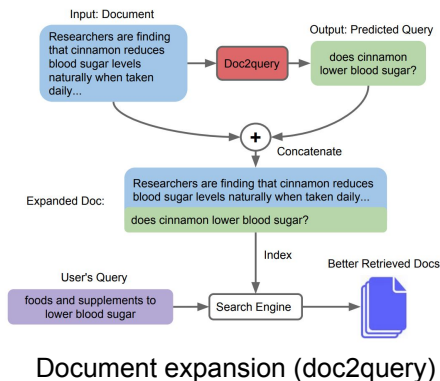
55 inch tv mount

Search Engine

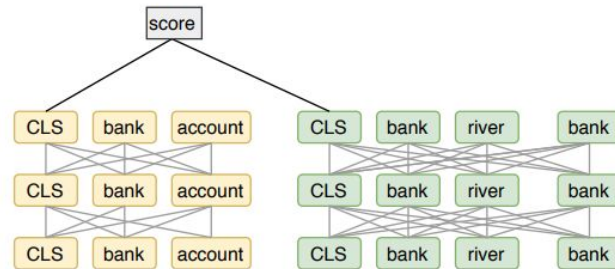
Better Retrieved Docs



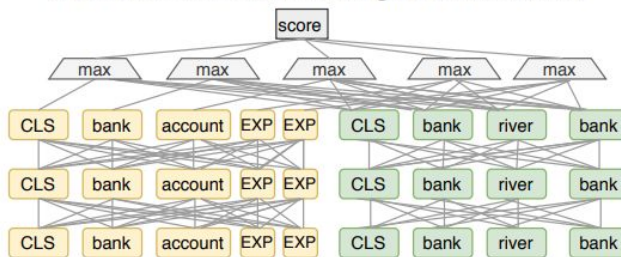
# In summary



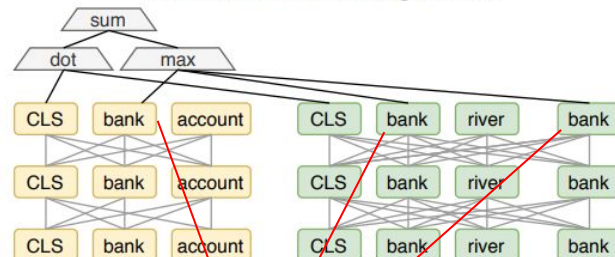
(a) Cross-Attention Model (e.g., BERT reranker)



(b) Dense Retrievers (e.g., DPR)



(c) ColBERT: All-to-All Match



(d) COIL: Contextualized Exact Match

exact match???

Sources: Gao et al., "COIL: Revisit Exact Lexical Match in Information Retrieval with Contextualized Inverted List", 2021  
 Formal et al., "SPLADE: Sparse lexical and expansion model for first stage ranking", 2021  
 Nogueira et al., "Document expansion by query prediction", 2019

# In-domain vs Out-of-domain

No distillation (e.g., Splade v2, ColBERT v2)  
No IR-specific pretraining (e.g. CoCondenser, Contriever)  
No larger training dataset (e.g., GTR, E5)

Method	nDCG@10	
	In-domain TREC-DL 20	Out-of-domain BEIR
BM25	0.475	0.440
doc2query (sparse)	0.619	0.453
SPLADE (sparse)	0.671	0.458
GTR-base (T5, dense)	0.696	0.426
ColBERT (dense multi vector)	0.676	0.453
COIL (dense multi vector)	0.688	0.483
BM25 + monoT5-base (reranker)	0.701	0.490