Token-level representations for retrieval

Itay Levy

- IR background
- ColBERT
  Balancing effectiveness and efficiency
- ColBERTv2
  Apply new tricks to get even better performance and reduced space footprint
- White box analysis
  Get more insights to what ColBERT actually does

ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT

Omar Khattab
Stanford University
okhattab@stanford.edu

Matei Zaharia
Stanford University
matei@cs.stanford.edu

SIGIR 2020

300+ citations

*some slides are from CS224U

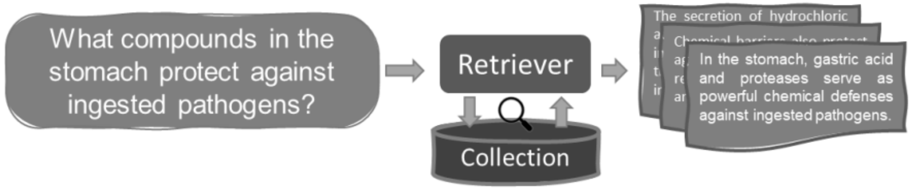
IR Background

IR Setup

- Offline
  - Index the corpus
- Online
  - Given a query, search for top-K relevant documents

Ranked Retrieval

- Scope: A large corpus of text documents (e.g., Wikipedia)
- Input: A textual query (e.g., a natural-language question)
- Output: **Top-K Ranking** of **relevant** documents (e.g., top-100)



IR Evaluation

- A search system must be **efficient** and **effective**
  - If we had infinite resources, we'd just hire experts to look through all the documents one by one!

- Efficiency
  - **Latency (milliseconds; for one query)**
  - Throughput (queries/sec)
  - Space (GBs for the index? TBs?)
  - Hardware required (one CPU core? Many cores? GPUs?)
  - Scaling to various collection sizes, under different loads

Classical IR

Classical IR

- For multi-term queries, classical IR models would tokenize and then treat the tokens independently.

$$RelevanceScore(query, doc) = \sum_{term \in query} Weight_{doc,term}$$

- This reduces a large fraction of classical IR to:
  - How do we best tokenize (and stem) queries and documents
  - **How do we best weight each term–document pair**

Classical IR

Term–Document Weighting: Intuitions

- **Frequency** of occurrence will remain a primary factor
  - If a term t occurs frequently in document d, the document is more likely to be relevant for queries including t
- **Normalization** will remain a primary component too
  - If that term t is rather rare, then document d is even more likely to be relevant for queries including t
  - If that document d is rather short, this also improves its odd

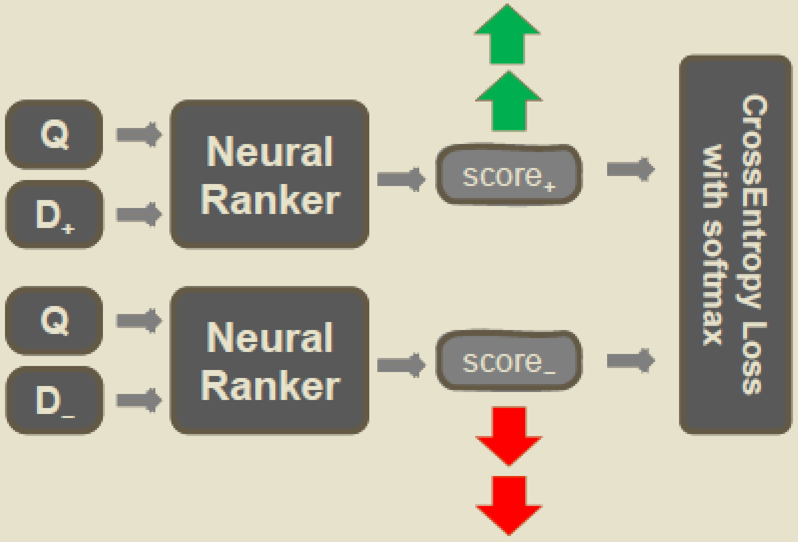
- Amplify the important, the trustworthy, the unusual; deemphasize the mundane and the quirky.

Neural IR

Neural ranking: training

- Many possible choices, but **2-way classification** is often effective!
  - Each training instance is a **triple**

    < query, positive document, negative document >



Recall that we can get positives for each query from our relevance assessments.

Every non-positive can often be treated as an implicit negative.

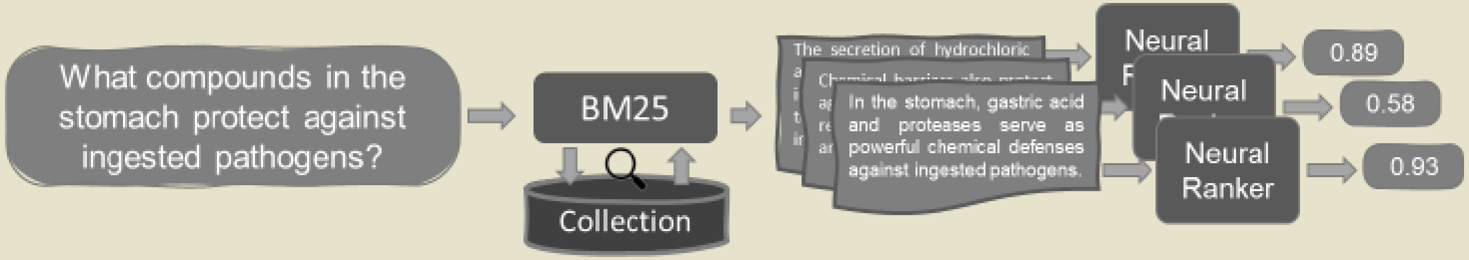
Neural ranking: Inference

- Given a query Q, pick each document d and pass $<Q, d>$ through the network. Sort all by score, returning the top-k results!

- But collections often have many millions of documents
  - MS MARCO has 9M passages
  - Even if you model runs in 1 micro second per passage, that's 9 seconds per query!

Neural re-ranking: Pipelines

- BM25 top-1000 -> Neural IR reranker



- Cuts the work on 10M documents by factor of 10k!
  - But introduces an artificial recall ceiling.

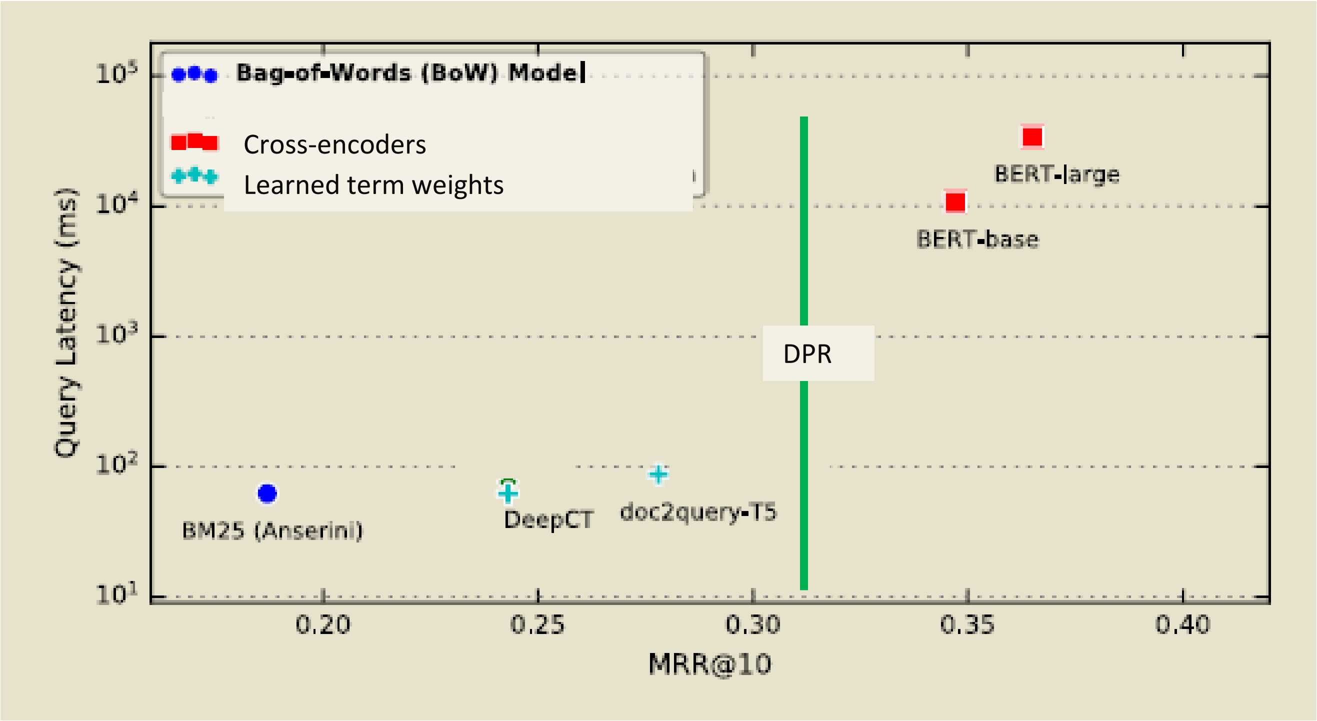
Can we do better?

Yes! Later, we'll discuss

End-to-end retrieval is essential toward improving RECALL.

Neural IR paradigms

1. Cross-encoders
2. Learned term weights
3. Representation similarity
4. Late interaction



Neural IR paradigms: Representation similarity

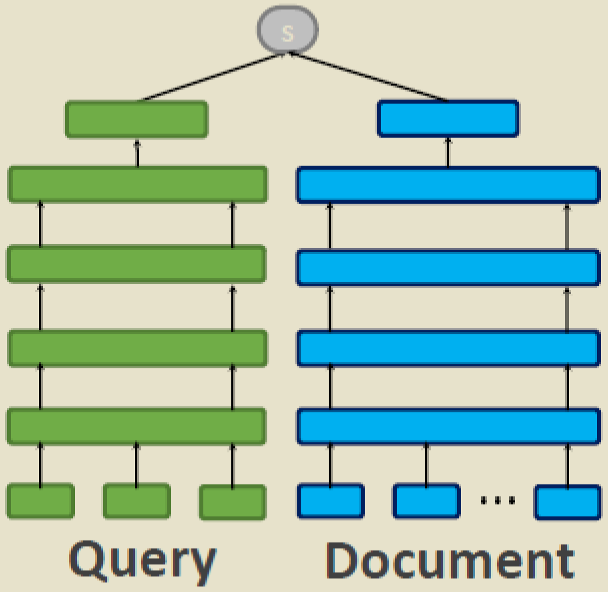
- Tokenize the query and the document
- **Independently** encode the query and the document
- … into a **single-vector** representation each
- Estimate relevance a dot product
  - Or a cosine similarity

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.



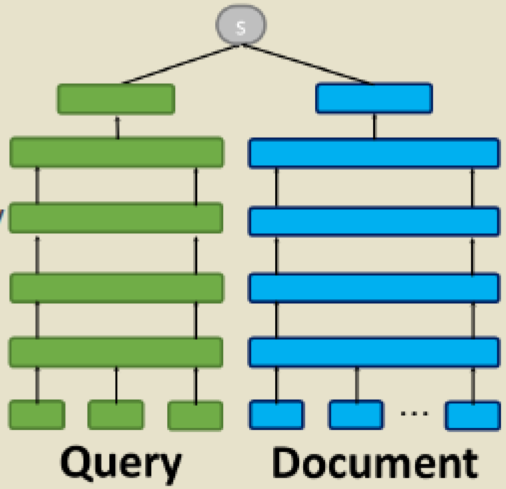
3

SBERT, ORQA, **DPR**, DE-BERT, RepBERT, ANCE

Neural IR paradigms: DPR

Dense Passage Retriever (DPR) by *Karpukhin et al.*

- Encodes each passage into a 768-dimensional vector
- Encodes each query into a 768-dimensional vector
- Trained with N-way cross-entropy loss, over the similarity scores between the query and:
  - A positive passage
  - A negative passage, sampled from BM25 top-100
  - Many in-batch negative passages
    - the positive passages for the *other* queries in the same training batch



Xiong et al. (2020) test a DPR-style retriever on MS MARCO: **31% MRR**. They show that a sophisticated supervision scheme can achieve **33%**.

Both constitute progress over "learned term weights" like DeepCT, but they are still considerably lower than standard BERT's **>36% MRR**.

Neural IR paradigms: Representation similarity downsides

✕ Single-Vector Representations

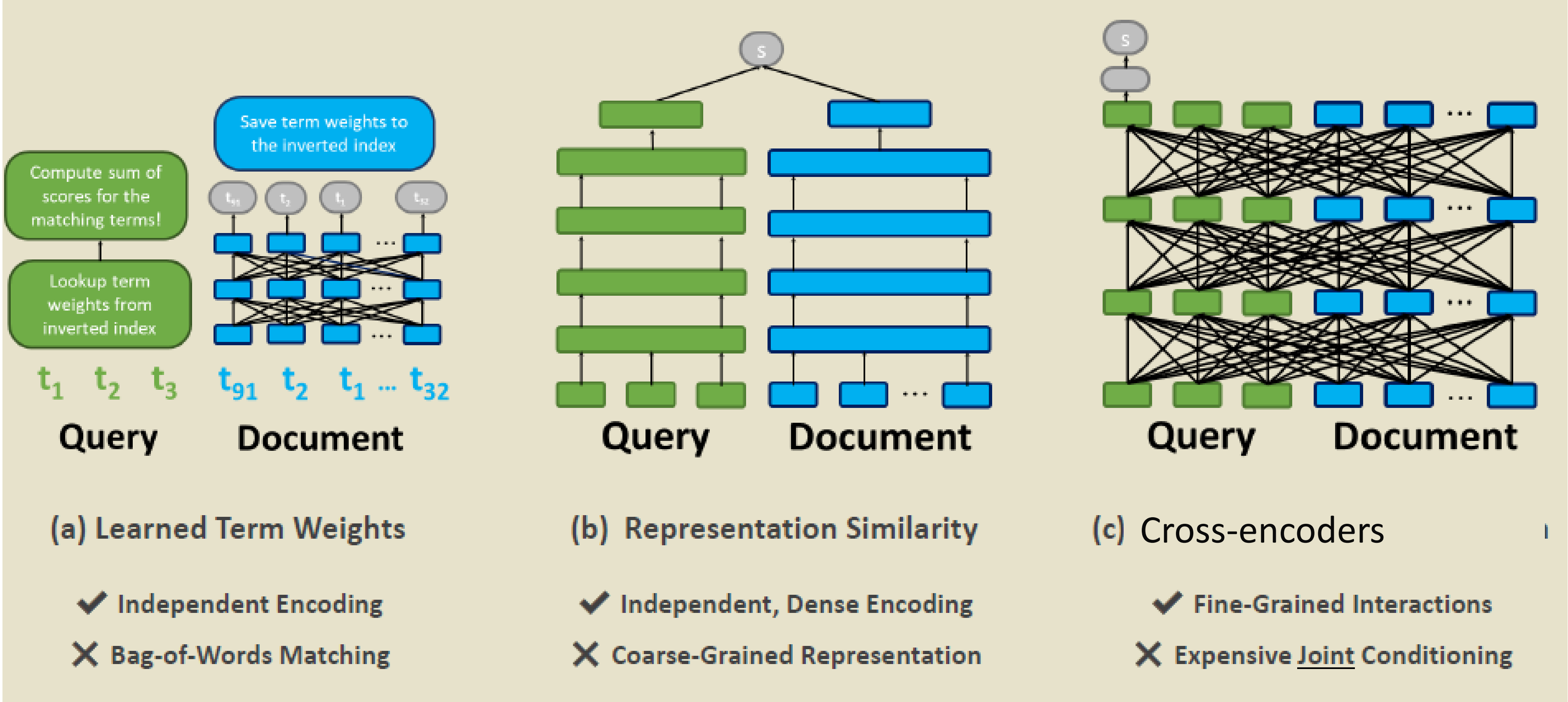
- They "cram" queries and documents into a **coarse-grained** representation!

✕ 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 or Duet)
  - And even term-weighting models (e.g., DeepCT and BM25)

Can we keep precomputation and still have fine-grained interactions?

Neural IR paradigms: Summary



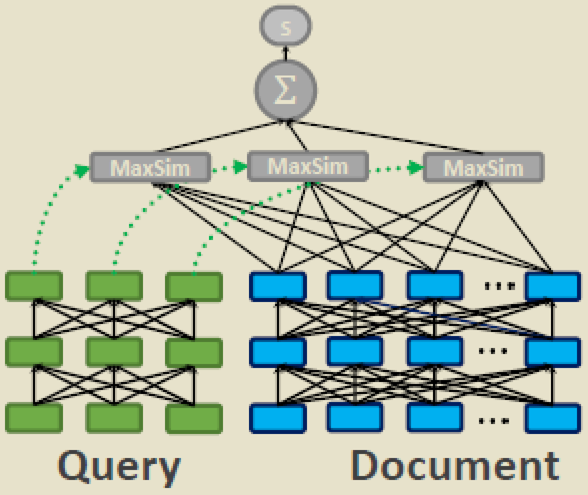
Neural IR paradigms: Summary

- Cross-encoders forced us to use a re-ranking pipeline, where we just re-scored the top-1000 documents retrieved by BM25.

End-to-end retrieval is essential toward improving RECALL.

- **Learning Term Weights** and **Representation Similarity** models alleviate this!
  - They allow us to do end-to-end retrieval: quickly searching over all documents directly.
  - We can save **term weights** in the **inverted index**. This means that we do NOT need a re-ranking pipeline.
  - We can also index **vector representations** for **fast vector-similarity search**, which allows **PRUNING** to find the top-K matches without exhaustive enumeration.
    - Libraries like **FAISS** abstract away the details.

Neural IR paradigms: ColBERT



(d) <u>Late Interaction</u>

(i.e., ColBERT)

Can we keep precomputation and still have fine-grained interactions?

Desired Properties:

- ✔ Independent Encoding
- ✔ Fine-Grained Representations
- ✔ End-to-End Retrieval (pruning!)

Notice that ColBERT represents the document as a MATRIX, not a vector.

Neural IR paradigms: ColBERT

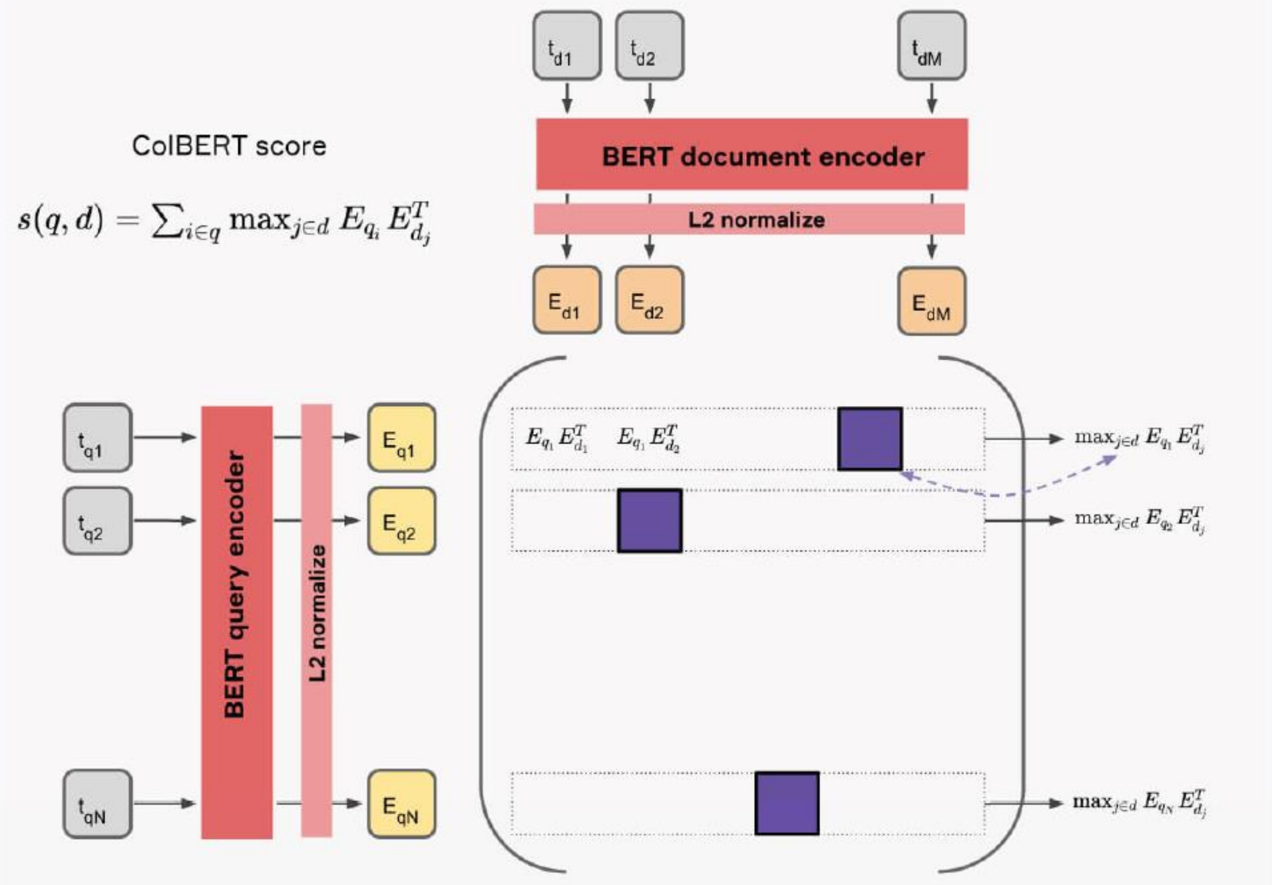


Figure 1. ColBERT model description.

Neural IR paradigms: ColBERT

Late Interaction: Real Example of Matching

when did the transformers cartoon series come out?

[...] the animated [...] The Transformers [...] [...] It was released [...] on August 8, 1986

when did the transformers cartoon series come out?

[...] the animated [...] The Transformers [...] [...] It was released [...] on August 8, 1986

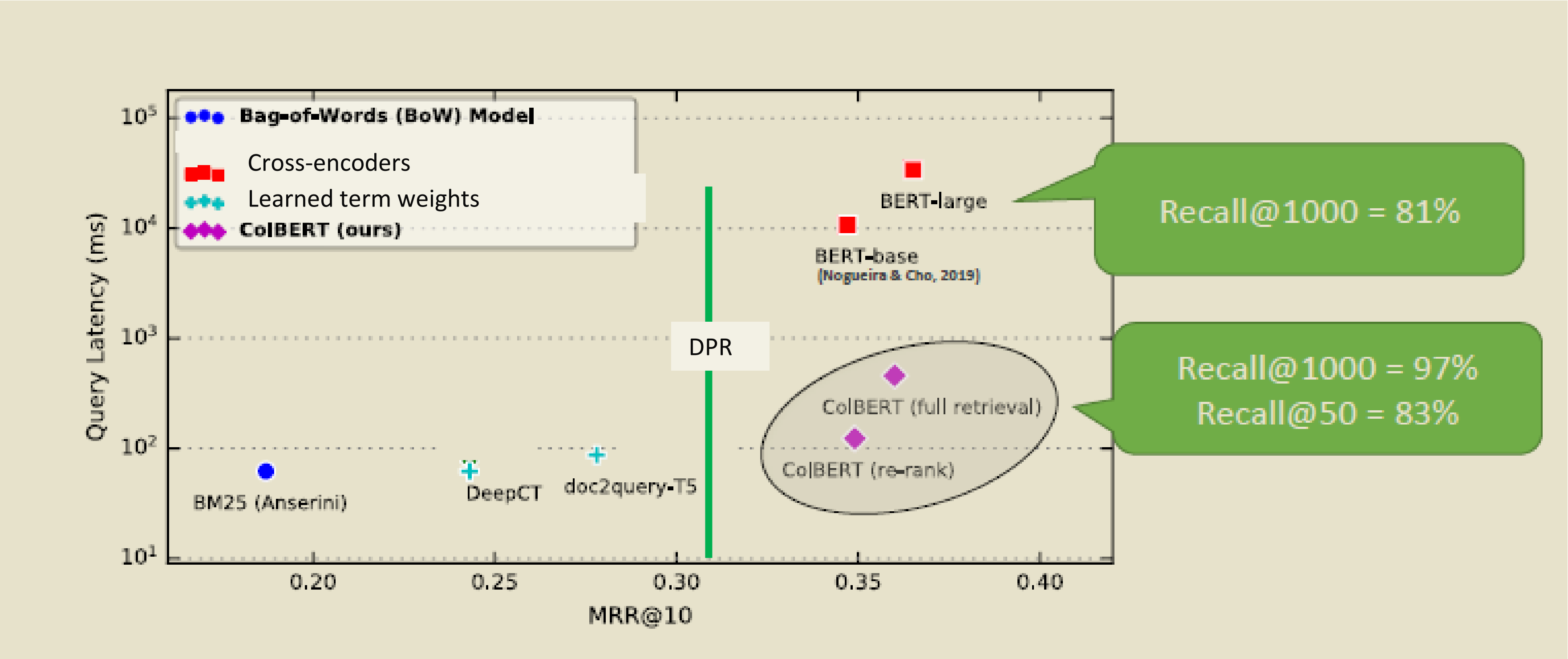
when did the transformers cartoon series come out?

[...] the animated [...] The Transformers [...] [...] It was released [...] on August 8, 1986

when did the transformers cartoon series come out?

[...] the animated [...] The Transformers [...] [...] It was released [...] on August 8, 1986

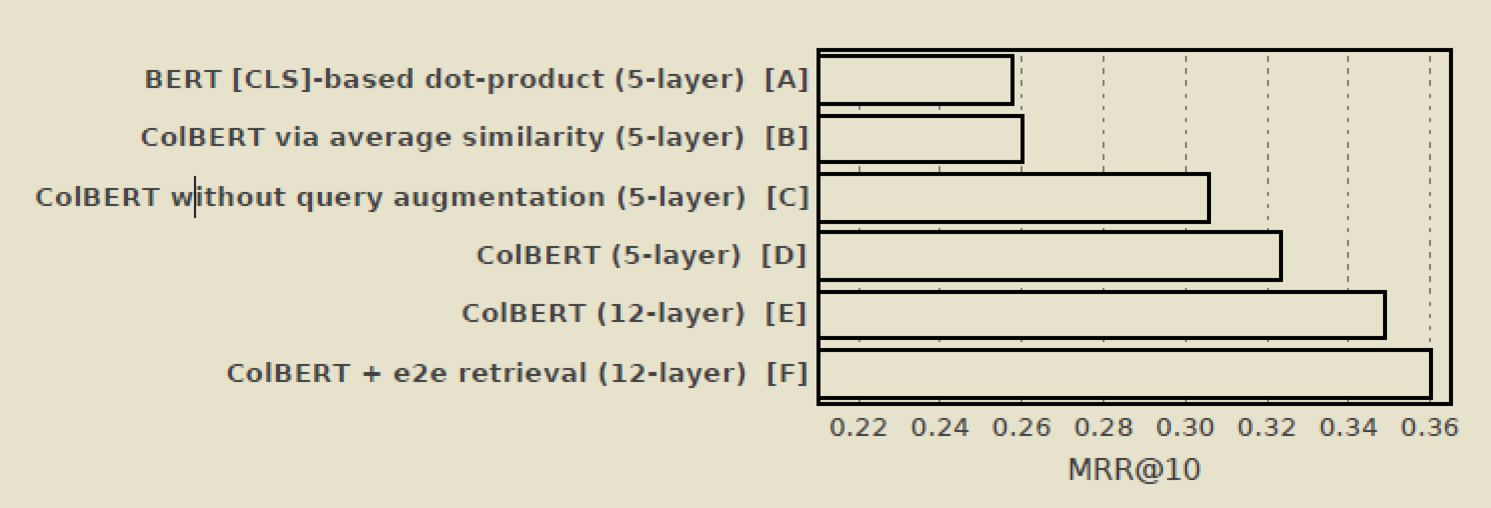
Neural IR paradigms: ColBERT



Colbert- MaxSim operator

1. Cheap yet effective
2. amenable to highly-efficient pruning for top-k retrieval

$$S_{q,d} := \sum_{i \in [|E_q|]} \max_{j \in [|E_d|]} E_{q_i} \cdot E_{d_j}^T$$



Colbert drawbacks

- space footprint and latency

> While ColBERT's embedding dimension has limited impact on the efficiency of query encoding, this step is crucial for controlling the space footprint of documents, as we show in §4.5. In addition, it can have a significant impact on query execution time, particularly the time taken for transferring the document representations onto the GPU from system memory (where they reside before processing a query). In fact, as we show in §4.2, gathering, stacking, and transferring the embeddings from CPU to GPU can be the most expensive step in re-ranking with ColBERT. Finally, the output

Colbert drawbacks

- space footprint and latency

DBPedia [21] (1 Million)			Retrieval Latency		Index
Rank	Model	Dim.	GPU	CPU	Size
(1)	BM25+CE	–	450ms	6100ms	0.4GB
(2)	ColBERT	128	350ms	–	20GB
(3)	docT5query	–	–	30ms	0.4GB
(4)	BM25	–	–	20ms	0.4GB
(5)	TAS-B	768	14ms	125ms	3GB
(6)	GenQ	768	14ms	125ms	3GB
(7)	ANCE	768	20ms	275ms	3GB
(8)	SPARTA	2000	–	20ms	12GB
(9)	DeepCT	–	–	25ms	0.4GB
(10)	DPR	768	19ms	230ms	3GB

Table 3: Estimated average retrieval latency and index sizes for a single query in DBPedia [21]. Ranked from best to worst on zero-shot BEIR. Lower the latency or memory is desired.

Mitigating Colbert drawbacks

- Compression

Setting	Dimension(m)	Bytes/Dim	Space(GiBs)	MRR@10
Re-rank Cosine	128	4	286	34.9
End-to-end L2	128	2	154	36.0
Re-rank L2	128	2	143	34.8
Re-rank Cosine	48	4	54	34.4
Re-rank Cosine	24	2	27	33.9

Table 4: Space Footprint vs MRR@10 (Dev) on MS MARCO.

- Clustering
  - search the nearest 10 centroids to each query embedding

Takeaways

- Late Fine-grained interactions in ColBERT balance the effectiveness – efficiency tradeoff well
- Need further work on improving space footprint

ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction

Keshav Santhanam* Stanford University

Omar Khattab* Stanford University

Jon Saad-Falcon Georgia Institute of Technology

Christopher Potts Stanford University

Matei Zaharia Stanford University

NAACL 2022

Colbertv2 – Improving Efficiency

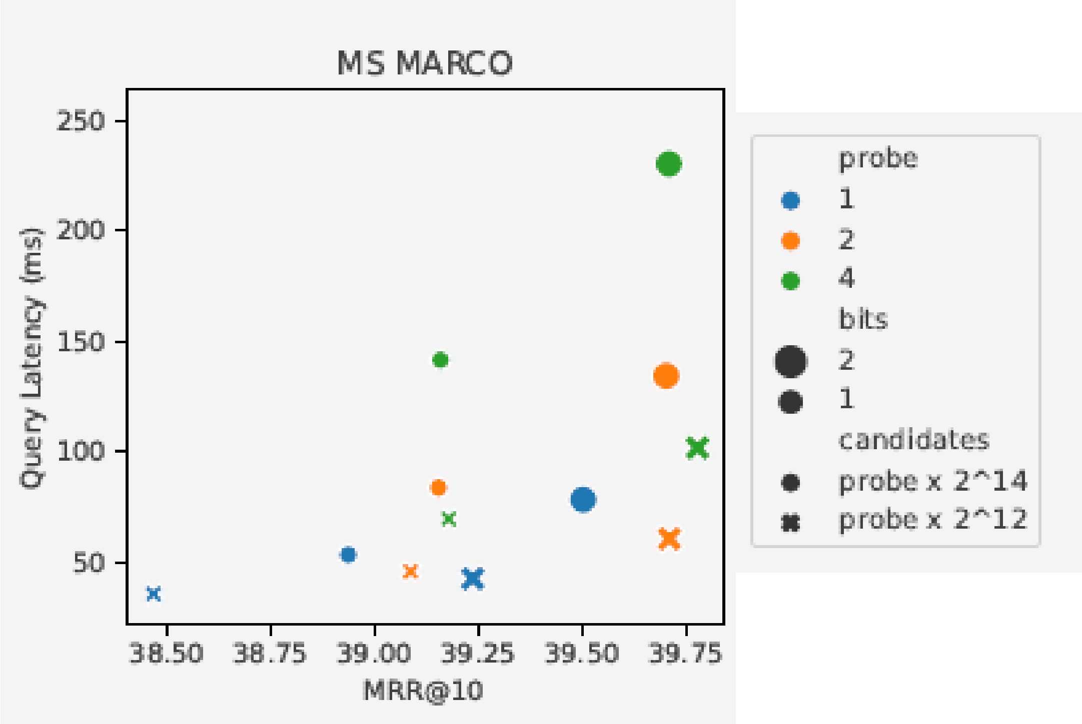
Residual compression

centroids C, ColBERTv2 encodes each vector v as the index of its closest centroid C_t and a *quantized* vector $\tilde{r}$ that approximates the residual $r = v - C_t$.

ColBERT. Whereas ColBERT requires 154 GiB to store the index for MS MARCO, ColBERTv2 only requires 16 GiB or 25 GiB when compressing embeddings to 1 or 2 bit(s) per dimension, respectively, resulting in compression ratios of 6–10×.

Colbertv2 – Improving Efficiency

The figure varies three settings of ColBERTv2. In particular, we evaluate indexing with 1-bit and 2-bit encoding (§3.4) and searching by probing the nearest 1, 2, or 4 centroids to each query vector (§3.5). When probing `probe` centroids per vector, we score either `probe` $\times 2^{12}$ or `probe` $\times 2^{14}$ candidates per query.[8]



Colbertv2 – better supervision

Distillation from cross-encoder and hard-negative mining

- 63 hard negatives instead of 1
- In batch negatives
- Refreshing index

Colbertv2 – in-domain results

Method	Official Dev (7k) MRR@10	Official Dev (7k) R@50	Official Dev (7k) R@1k	Local Eval (5k) MRR@10	Local Eval (5k) R@50	Local Eval (5k) R@1k
Models without Distillation or Special Pretraining						
RepBERT	30.4	-	94.3	-	-	-
DPR	31.1	-	95.2	-	-	-
ANCE	33.0	-	95.9	-	-	-
LTRe	34.1	-	96.2	-	-	-
ColBERT	36.0	82.9	96.8	36.7	-	-
Models with Distillation or Special Pretraining						
TAS-B	34.7	-	97.8	-	-	-
SPLADEv2	36.8	-	97.9	37.9	84.9	98.0
PAIR	37.9	86.4	98.2	-	-	-
coCondenser	38.2	-	**98.4**	-	-	-
RocketQAv2	38.8	86.2	98.1	39.8	85.8	97.9
ColBERTv2	**39.7**	**86.8**	**98.4**	**40.8**	**86.3**	**98.3**

Table 4: In-domain performance on the development set of MS MARCO Passage Ranking as well the "Local Eval" test set described by Khattab and Zaharia (2020).

BEIR

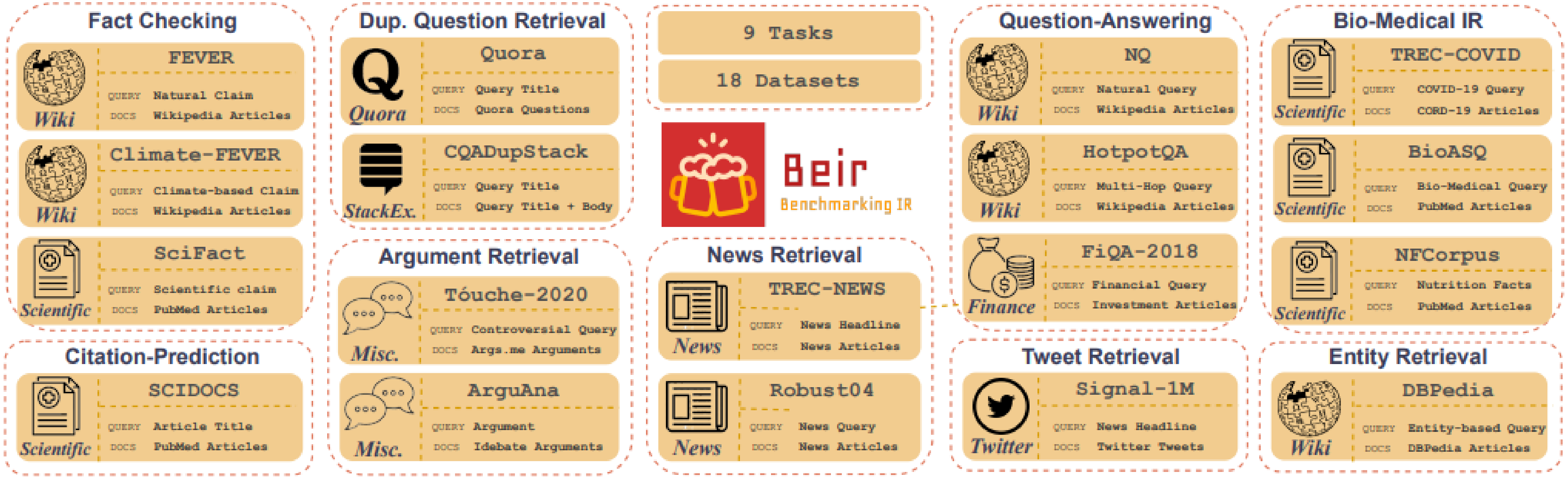


Figure 1: An overview of the diverse tasks and datasets in BEIR benchmark.

Colbertv2 – OOD results

The models that decompose scoring into term level interactions, ColBERTv2 and SPLADEv2, are almost always the strongest

Corpus	Models without Distillation				Models with Distillation			
	ColBERT	DPR-M	ANCE	MoDIR	TAS-B	RocketQAv2	SPLADEv2	ColBERTv2
BEIR Search Tasks (nDCG@10)								
DBPedia	39.2	23.6	28.1	28.4	38.4	35.6	43.5	**44.6**
FiQA	31.7	27.5	29.5	29.6	30.0	30.2	33.6	**35.6**
NQ	52.4	39.8	44.6	44.2	46.3	50.5	52.1	**56.2**
HotpotQA	59.3	37.1	45.6	46.2	58.4	53.3	**68.4**	66.7
NFCorpus	30.5	20.8	23.7	24.4	31.9	29.3	33.4	**33.8**
T-COVID	67.7	56.1	65.4	67.6	48.1	67.5	71.0	**73.8**
Touché (v2)	-	-	-	-	-	24.7	**27.2**	26.3
BEIR Semantic Relatedness Tasks (nDCG@10)								
ArguAna	23.3	41.4	41.5	41.8	42.7	45.1	**47.9**	46.3
C-FEVER	18.4	17.6	19.8	20.6	22.8	18.0	**23.5**	17.6
FEVER	77.1	58.9	66.9	68.0	70.0	67.6	**78.6**	**78.5**
Quora	85.4	84.2	85.2	**85.6**	83.5	74.9	83.8	85.2
SCIDOCS	14.5	10.8	12.2	12.4	14.9	13.1	**15.8**	15.4
SciFact	67.1	47.8	50.7	50.2	64.3	56.8	**69.3**	**69.3**

(a)

Table 5: Zero-shot evaluation results. Sub-table (a) reports resu

Takeaways

- Applying new tricks to ColBERT show:
  - SOTA performance both in and **out of domain**
  - Comparable space footprint
- All at the cost of added complexity

A White Box Analysis of ColBERT

Thibault Formal[1,2], Benjamin Piwowarski[1], and Stéphane Clinchant[2]

[1] Sorbonne Université, LIP6, F-75005 Paris, France
benjamin.piwowarski@lip6.fr
[2] Naver Labs Europe, Meylan, France
firstname.name@naverlabs.com

ECIR 2021
Best Short Paper Award

Setting

- Reranking 1000 BM25 candidates
- MSMARCO
- Simplified ColBERTv1
- From subwords to words → sum of contributions

ColBERT term importance

Term-importance is a well known heuristic in IR (IDF)

Does ColBERT (implicitly) capture a notion of term importance ?

Term importance ~ *contribution* of a term in the ranking == difference between original ranking and the ranking given when we drop the contribution of a term (τ-AP [3])

Values close to 1 == rankings are the same == the term is pretty much useless for the final decision

ColBERT implicitly captures IDF

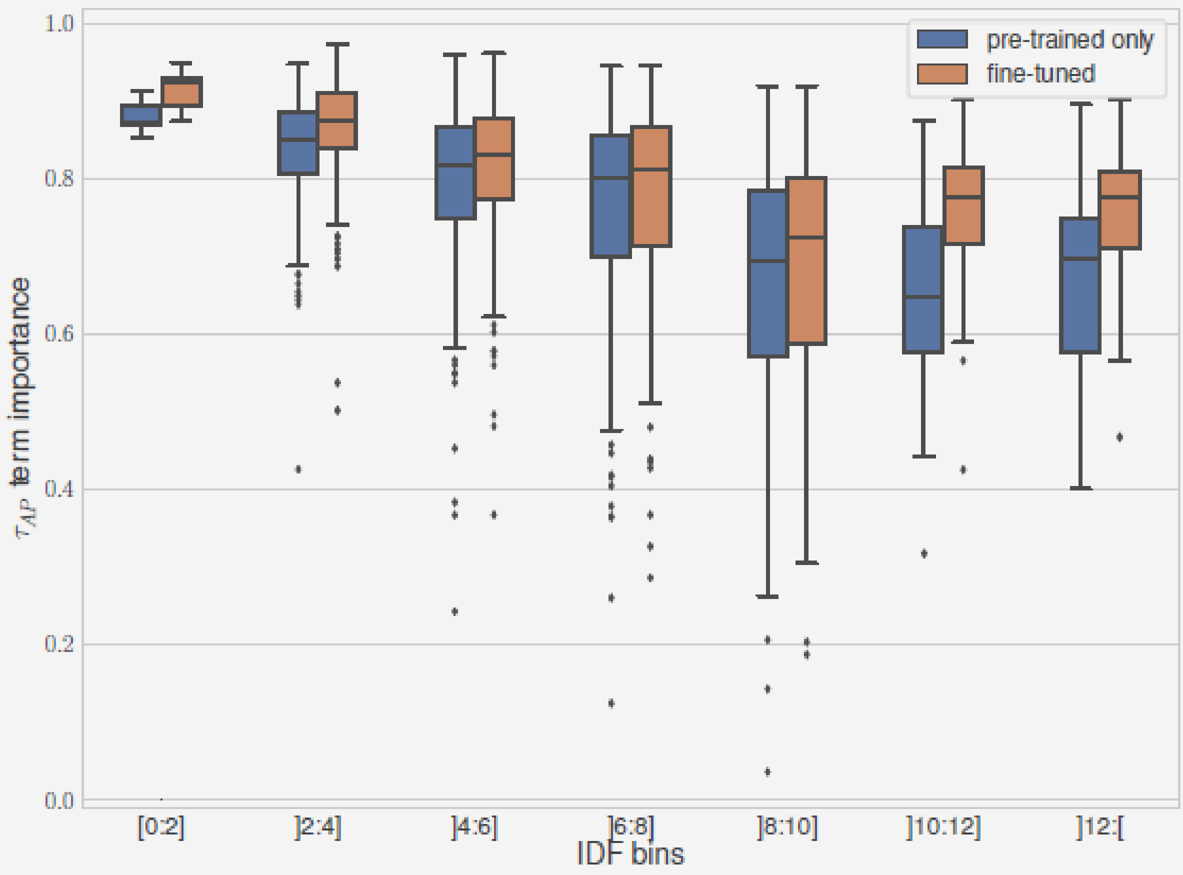


Fig. 1. ColBERT term importance (as computed using τ_{AP}) with respect to IDF (standard term importance).

Pearson r = -0.4

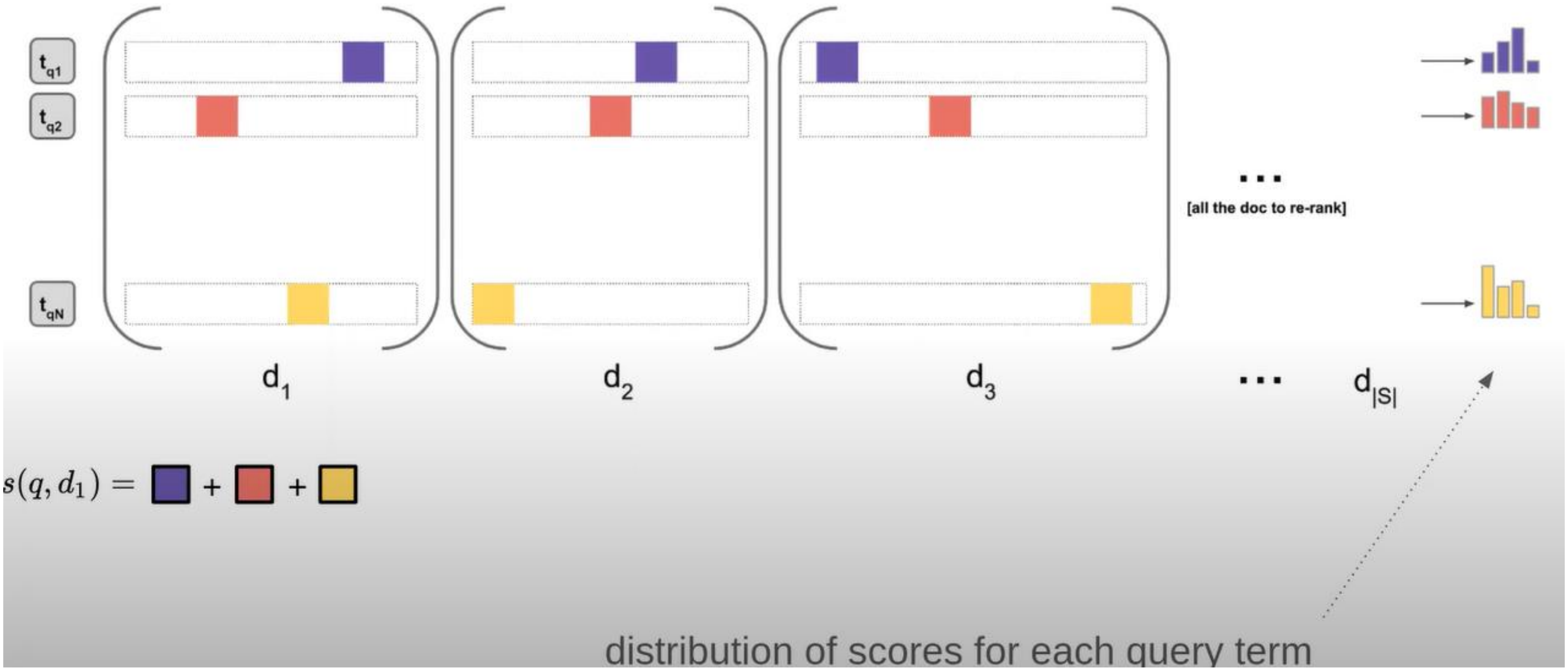
Exact/Soft match patterns

Neural models ⤳ soft-matching

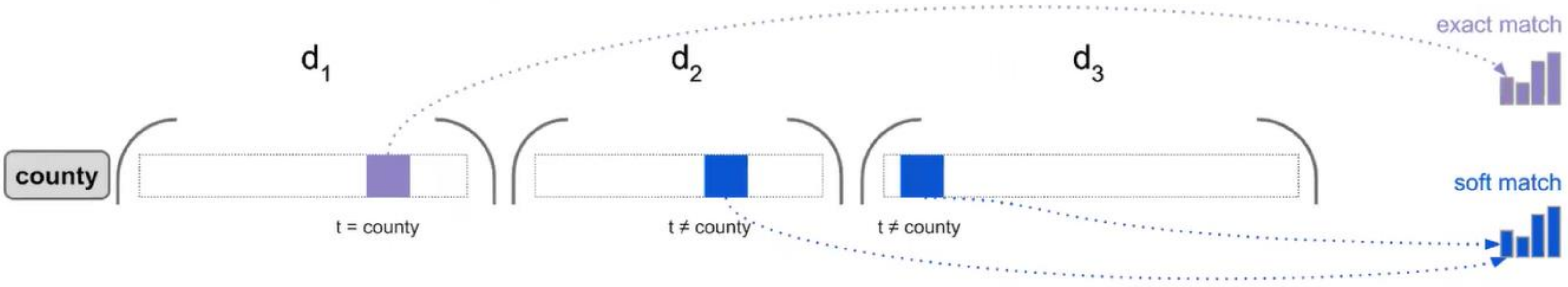
Exact matching is still a critical component of IR systems !

Does ColBERT capture exact match ?

Exact/Soft match patterns



Exact/Soft match patterns

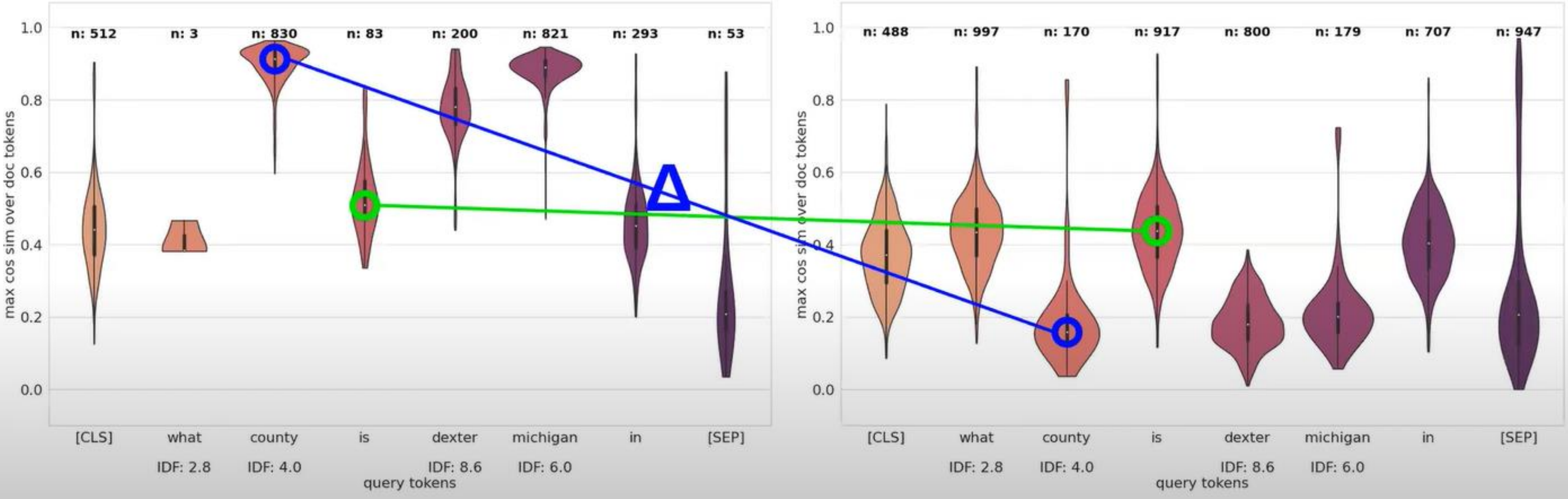


2 distributions of scores for each query term

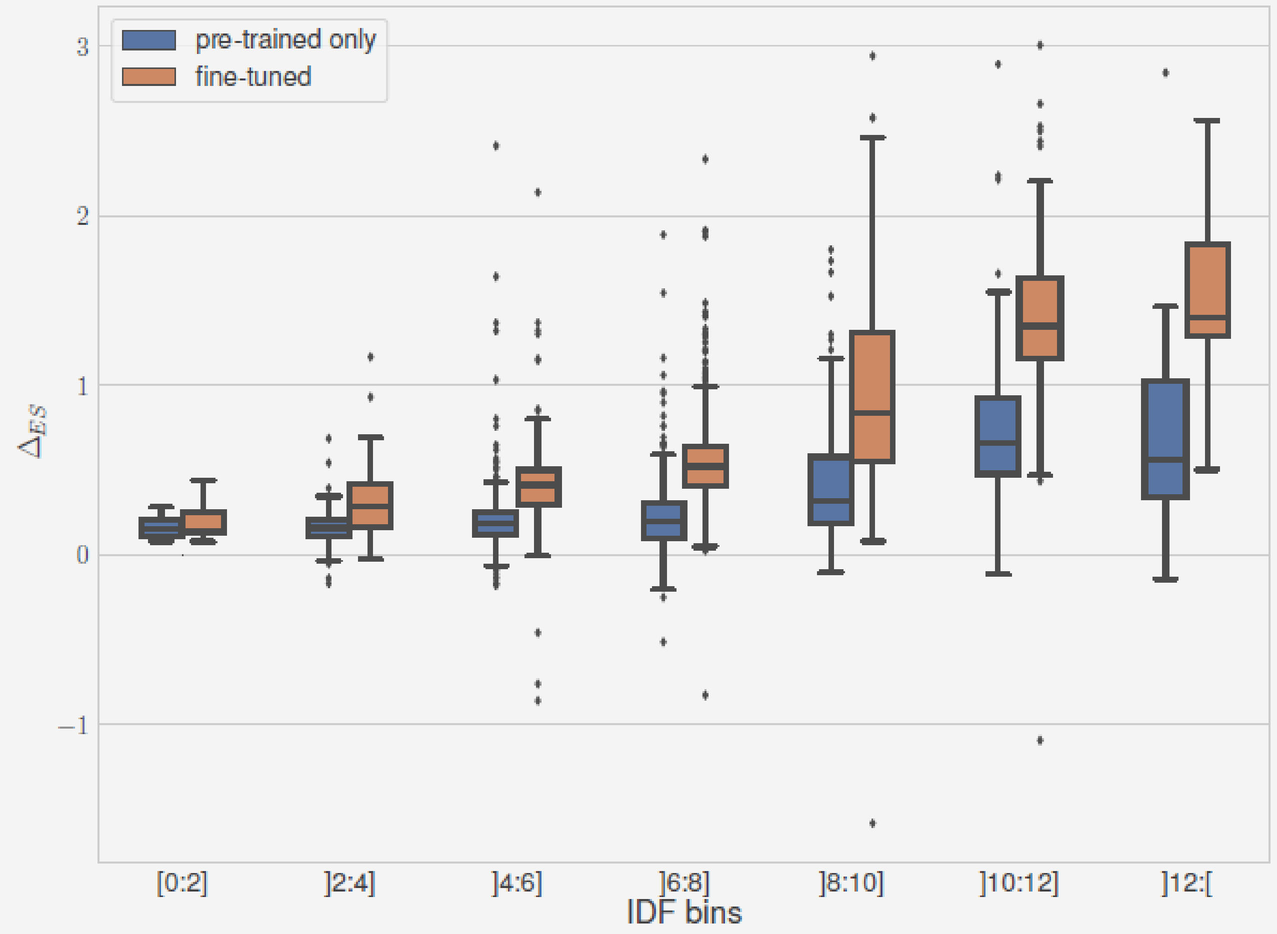
- exact case
- soft case

Exact/Soft match patterns

Δ exact-soft



Exact/Soft match patterns



Pearson r = 0.667

Fig. 2. Δ_{ES} with respect to IDF: we observe a moderate correlation (0.667) between Δ_{ES} and IDF, showing that the less frequent a term is, the more it is likely to be matched exactly.

Contextual embeddings variation

Colbert can distinguish terms for which exact match is important !

But how is it able to promote exact match from the contextualized embeddings ?

Contextual embeddings variation

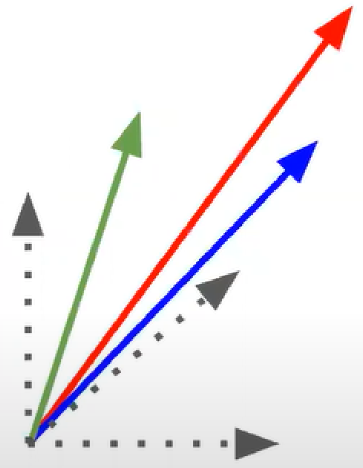
Hypothesis: **content** words have contextualized embeddings pointing to the **same** direction

[...] mango is an exotic fruit [...]

[...] mango is now cultivated in most frost-free tropical [...]

...

bla bli blo is mango



Contextual embeddings variation

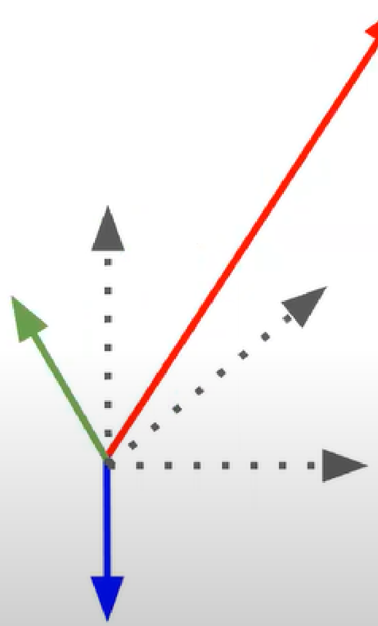
Hypothesis: **frequent** words have contextualized embedding pointing to **different** directions

[...] mango is an exotic fruit [...]

[...] mango is now cultivated in most frost-free tropical [...]

...

bla bli blo is mango

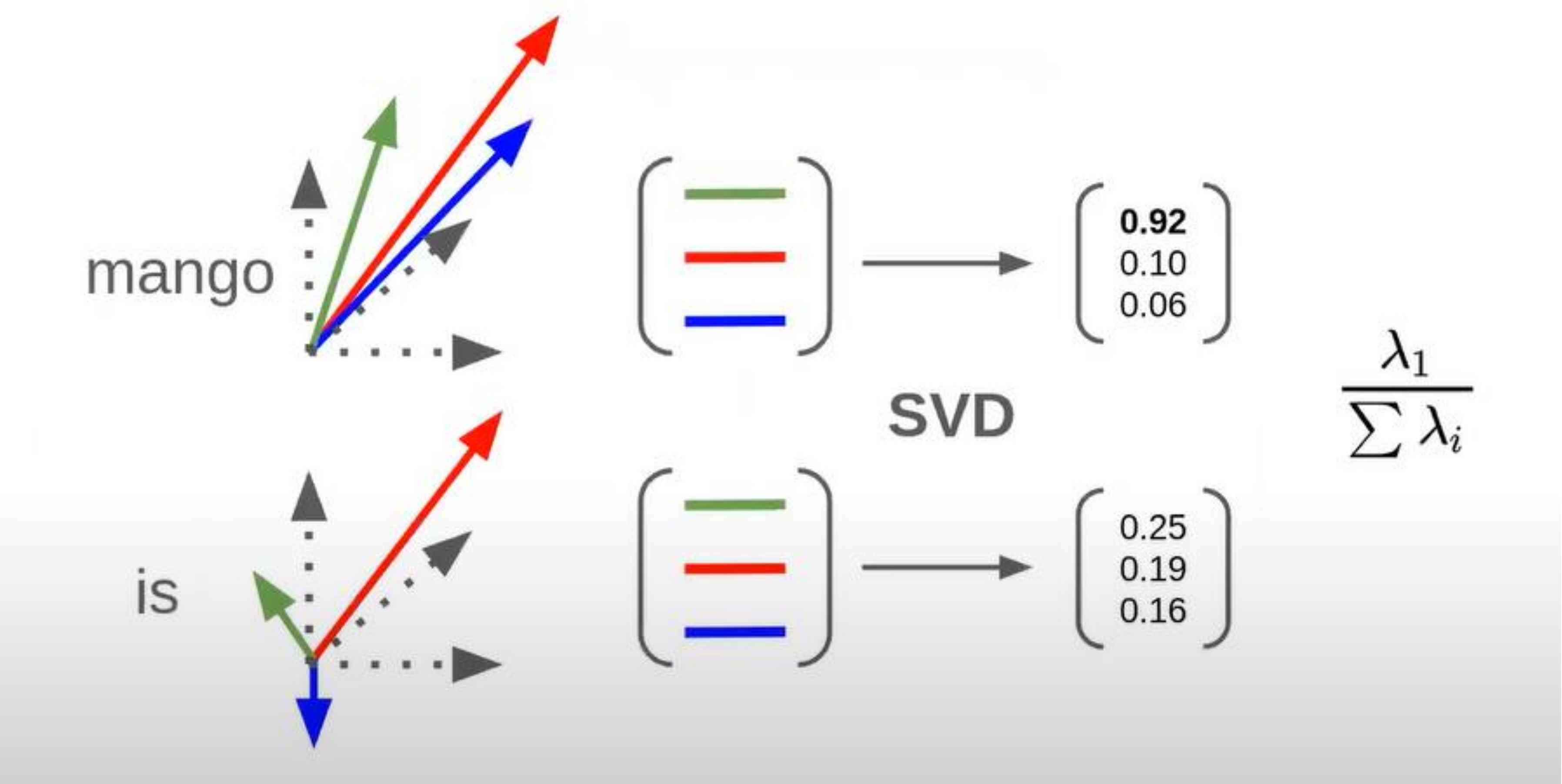


Contextual embeddings variation

Hypothesis

- for important terms, contextual embeddings vary less, hence ColBERT will tend to select the same term in documents (*cosine sim close to 1*)
- terms carrying less information tend to absorb more the context in sequences, hence their embeddings vary more

Spectral analysis of contextual term embeddings



$$\frac{\lambda_1}{\sum \lambda_i}$$

Contextual embeddings variation

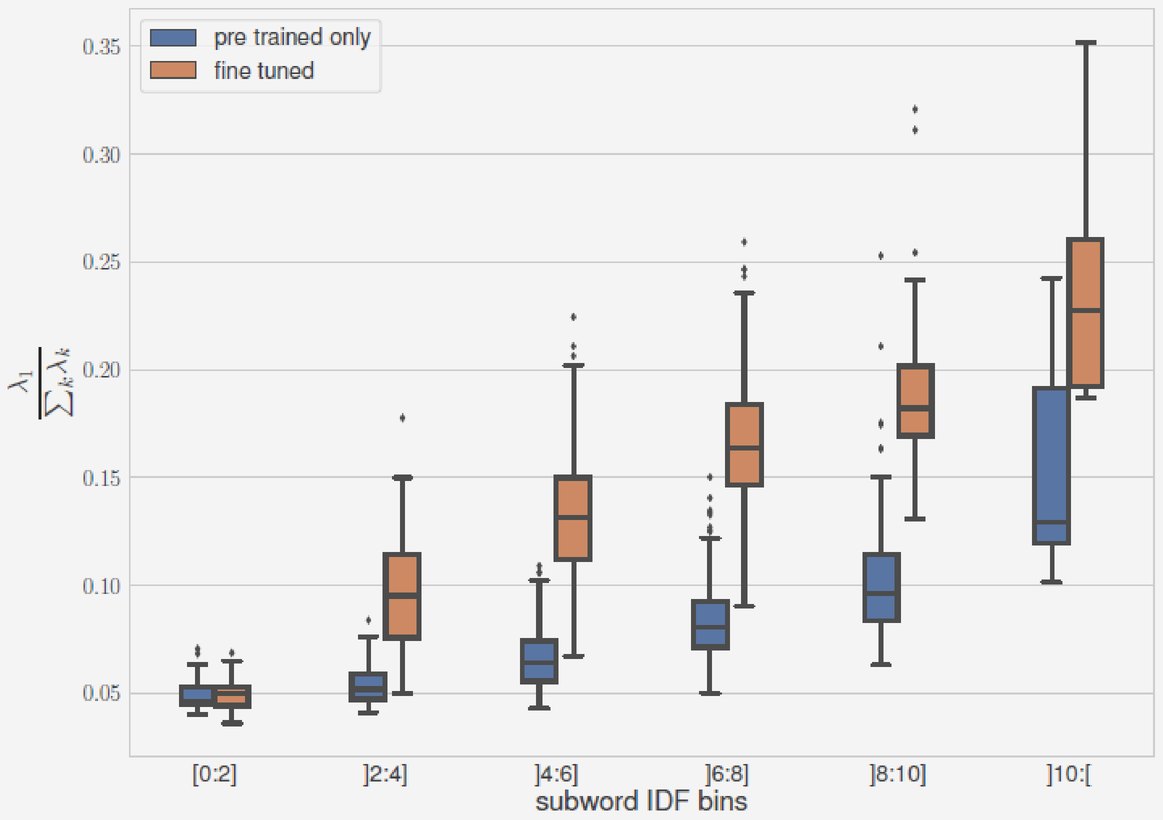


Fig. 3. Ratio of the first eigenvalue to the sum of the eigenvalues with respect to IDF (subword level). The less frequent the term is, the higher the ratio is, *showing that all contextualized embedding for a rare term are concentrated in the same direction.*

Pearson r = 0.77

Contextual embeddings variation

Query	
when did family feud come out ?	feud (IDF=9) → feud, feud, feud, feud, feud, feud, trait, feud, feud, feud, feud, name, ##bers, feud, feud come (IDF=3.6) → happen, item, landing, released, name, en, going, it, episode, game, reactions, goes, released, red, came
what is the botanical name for mango	mango (IDF=8.1) → mango, mango, mango, mango, mango, mango, mango, mango, mango, mango, mango, ge, mango, garden, mango name (IDF=3.1) → phrase, variety, a, them, the, for, skin, term, is, mango, top, rooms, name, on, known
who formed the commonwealth of independent states	commonwealth (IDF=7.4) → commonwealth, commonwealth, us, commonwealth, commonwealth, commonwealth, commonwealth, commonwealth, services, issued, commonwealth, commonwealth, commonwealth, commonwealth, six formed (IDF=3.2) → formed, became, by, as, became, independent, a, established, established, issued, states, in, of, lending, a

Figure 9. Sample of matched terms, for query terms with different IDF values.

Takeaways

(1) ColBERT captures a notion of term importance

(2) Exact match remains a key component

(3) and is promoted for terms with high IDF