

Dense Retrieval ♥ Knowledge Distillation

Sebastian Hofstätter

sebastian.hofstaetter@tuwien.ac.at

 [/s_hofstaetter](https://twitter.com/s_hofstaetter)

Today

Dense Retrieval Knowledge Distillation

1

Dense Retrieval

- The BERT_{DOT} model
- (Approximate) nearest neighbor search

2

Knowledge Distillation

- Cross-architecture: Margin-MSE
- TAS-Balanced with dual supervision

3

Analyzing Dense Retrieval

- Zero-shot transfer
- Exploring new failure types

Neural Methods for IR Beyond Re-ranking

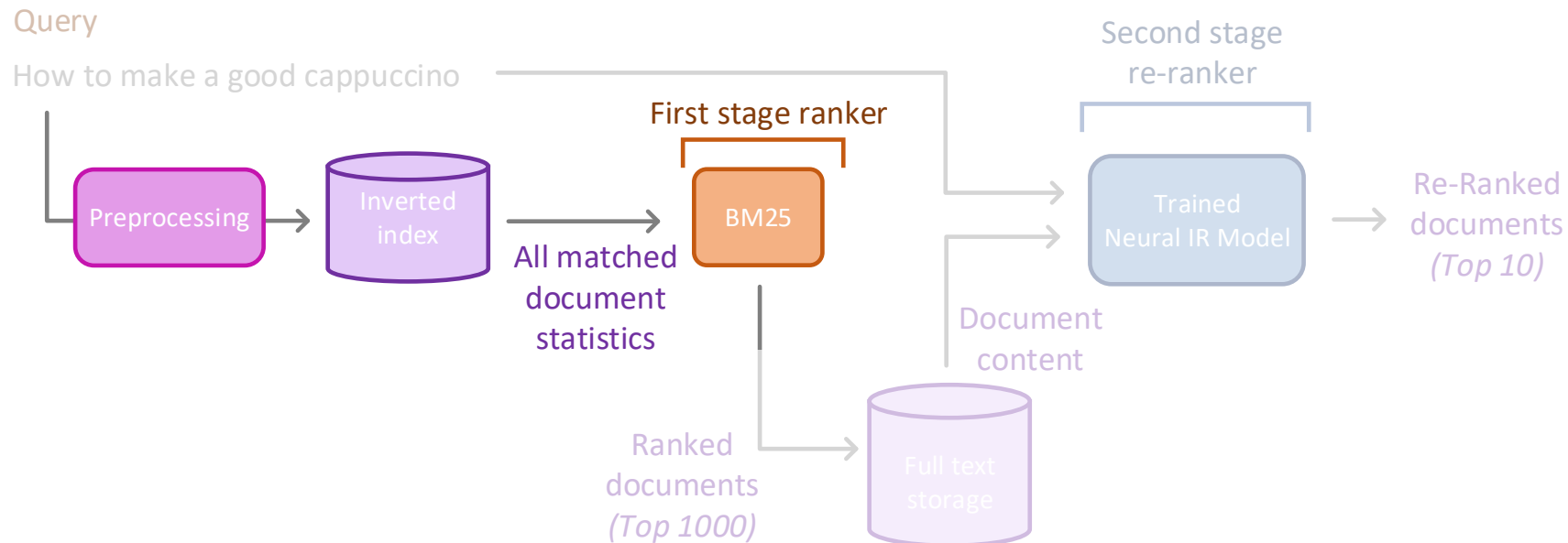
- Re-Ranking depends on a candidate selection (bottleneck)
 - How to bring neural advances in this first-stage phase
- Today we look at dense retrieval as (inverted index) BM25 alternative
- Many other neural approaches to improve first-stage retrieval:
 - **Doc2query**: Document expansion with query text that would semantically match the document. Exists in both BERT and T5 variants. Then index the expanded documents with BM25
 - **DeepCT**: Assign term weights based on BERT output during indexing -> retrieval with inverted index & BM25
 - **COIL**: Fuses contextual vectors into an inverted index structure, for faster lookup of semantic matches

Dense Retrieval

The future of search?

Recall: Neural Re-Ranking Models

- Re-rankers: They change the ranking of a pre-selected list of results
 - Same interface as classical ranking methods: $score(q, d)$
- Query workflow:

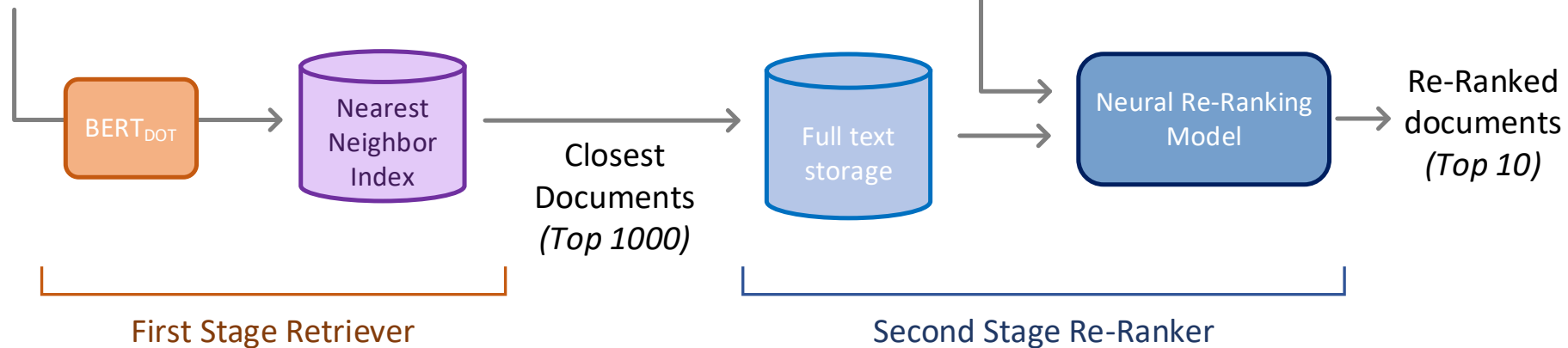


Dense Retrieval & Re-Ranking

- Dense retrieval replaces the traditional first stage
 - Using a neural encoder & nearest neighbor vector index
 - Can be used as part of a larger pipeline

Query

How to make a good cappuccino

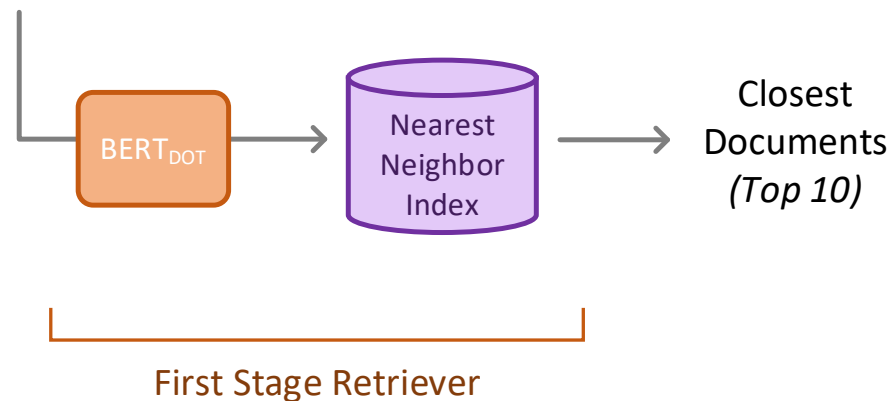


Standalone Dense Retrieval

- If dense retrieval is effective enough for our goals:
 - We can also use it as a standalone solution
 - Much faster + less complexity if we remove re-ranking stage

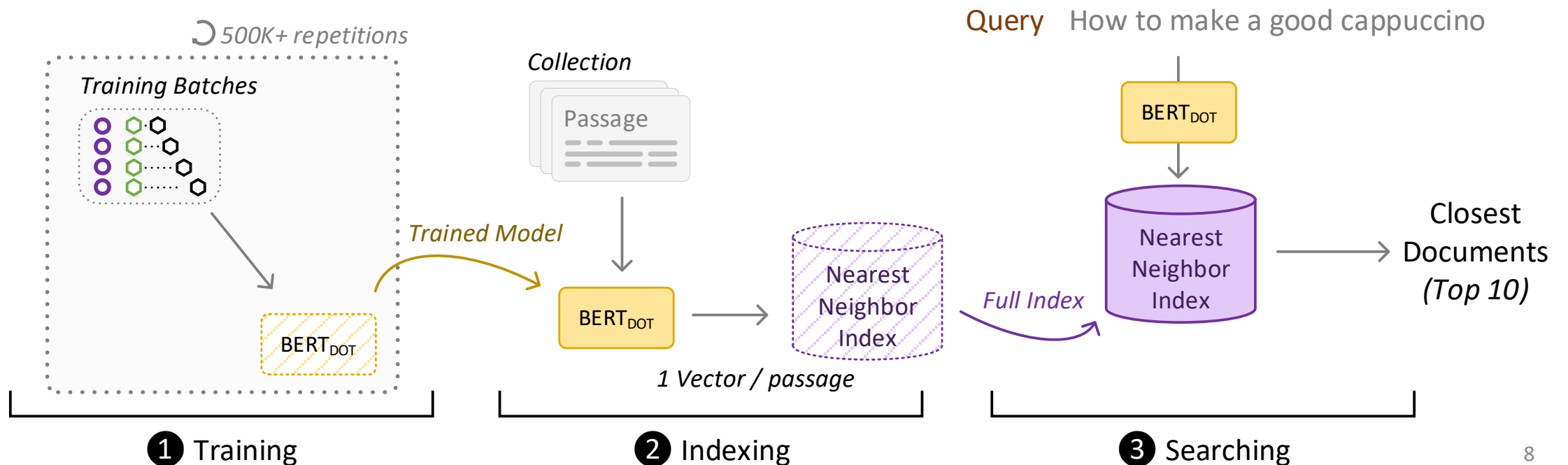
Query

How to make a good cappuccino

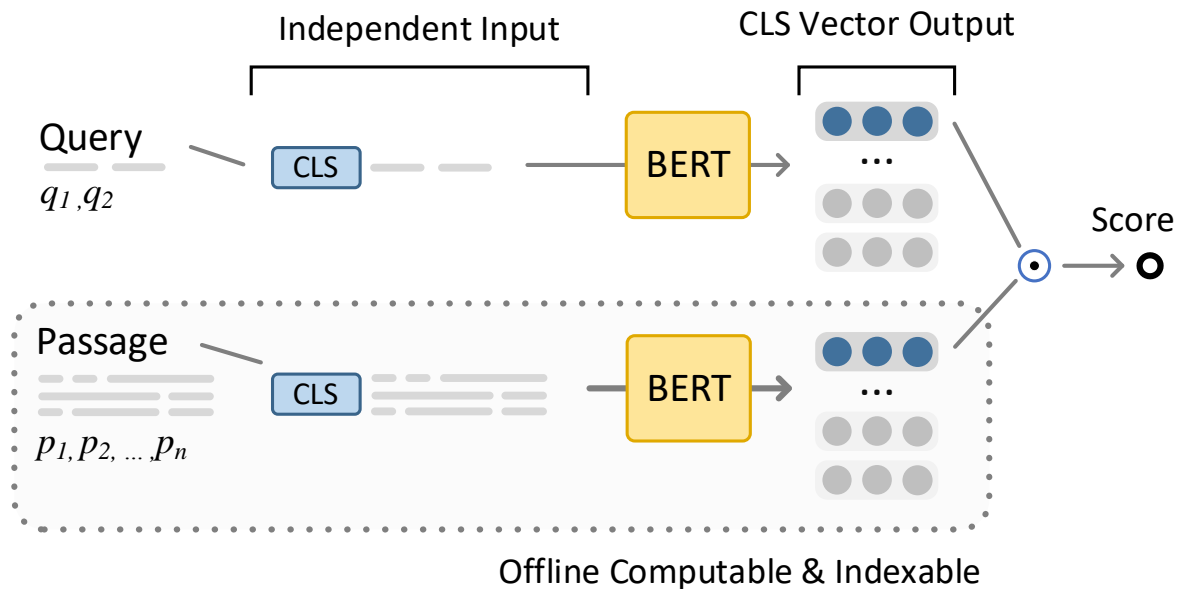


Dense Retrieval Lifecycle

- 3 major phases in the dense retrieval lifecycle
 - Each comes with several complex choices and required techniques
 - Could skip ❶ if we use a pre-trained model



BERT_{DOT} Model



- Passages and queries are compressed into a single vector
 - Passages are completely independent -> moves most computation into the indexing phase
 - Only need query encoding at runtime
- Relevance is scored with a dot-product
 - Cosine-sim variants also exist
 - This allows easy use of an (approximate) nearest neighbor index

BERT_{DOT}

- Simple formula (as long as we abstract BERT):

Encoding

$$\begin{aligned}\hat{q} &= \text{BERT}([CLS]; q_{1..n})_{CLS} \\ \hat{p} &= \text{BERT}([CLS]; p_{1..m})_{CLS}\end{aligned}$$

Independent computation

Matching

$$s = \hat{q} \cdot \hat{p}$$

Can be done “outside” the model
(with a nearest neighbor library)

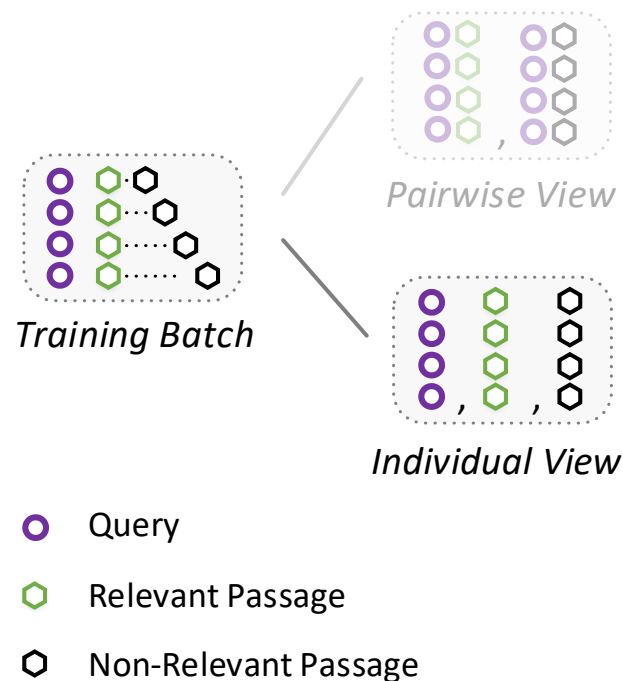
- Optional compression of \hat{q} , \hat{p} with a single linear layer

$q_{1..n}$	Query tokens
$p_{1..m}$	Passage tokens
BERT	Pre-trained BERT model
[CLS]	Special tokens
x_{CLS}	Pool the CLS vector
s	Output score

Training BERT_{DOT}

- We can train BERT_{DOT} like a re-ranking model
 - It works ok, but we can do more
- Considering that we don't want to use it as a re-ranker (more requirement on recall & not returning some random results)
- Paths for improved training:
 - ① No need to compute the score “inside” the model
 - Allows us to operate on output vectors instead (re-use encoding work)
 - ② Use the indexing capabilities during training
 - With repeated indexing we can use index results to sample passages

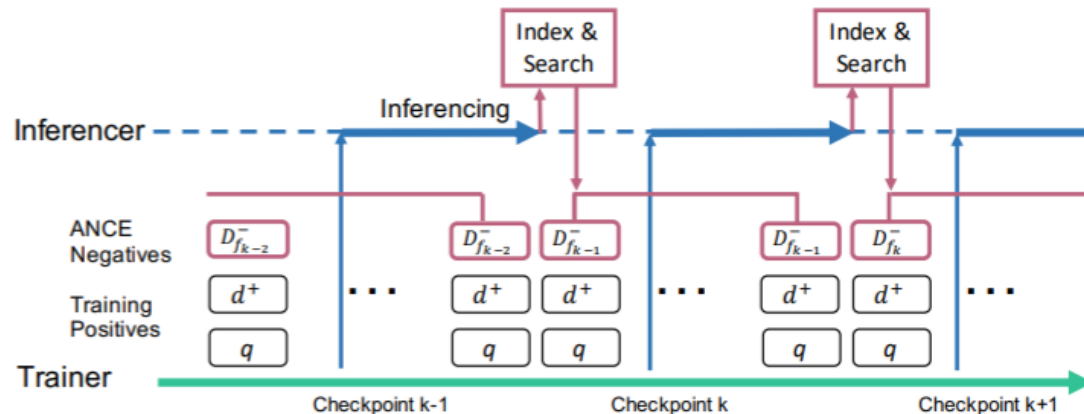
Sampling Strategies: In-Batch Negatives



- Dense retrieval does not need pairwise concatenation + aggregation is only 1 dot-product (virtually no overhead)
- Training workflow:
 - Forward pass can be done separately for query, 2x passages
 - Pairwise aggregation can be row-wise (as before) or crisscrossing between other samples in the batch
 - In-batch negatives: Because we anchor the relevant passage to its query and use every other passage in the batch as additional negative sample

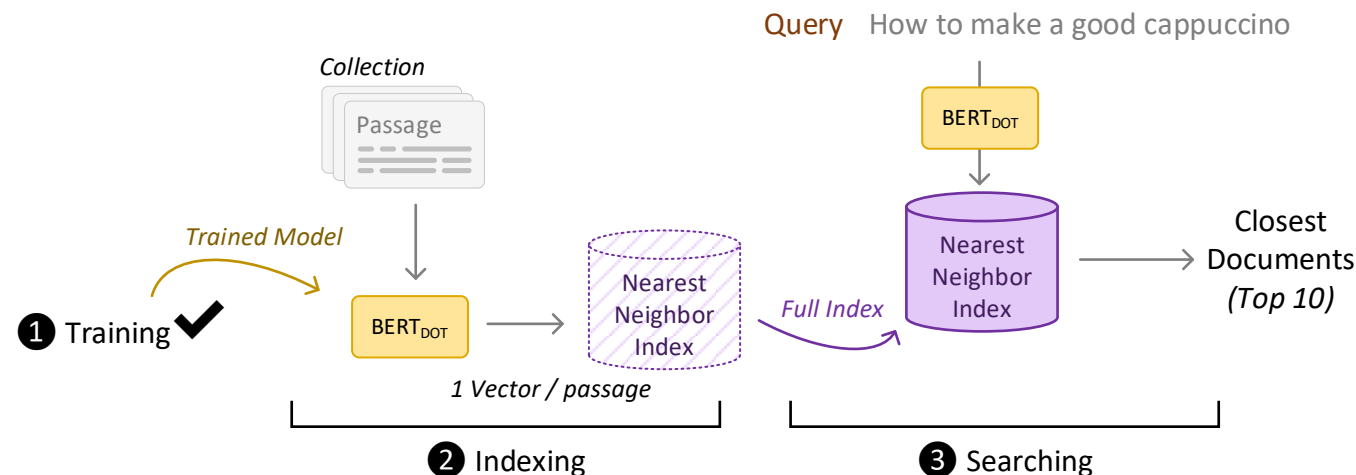
Sampling Strategies: ANCE

- We can train BERT_{DOT} with BM25 negative passages samples (like our re-ranking models)
- Or we may use the model itself to create new negative samples
 - ANCE: Continuous indexing & inference for new negative samples
 - Indexing costs multiple GPU hours every 10k batches (for MSMARCO)
 - Does not scale well, but works well if you have enough GPUs



Nearest Neighbor Search

- Once we have a trained DR model, we encode every passage in our collection
 - We save passages in an (approximate) nearest neighbor index
- During search we encode the query on the fly and search for nearest neighbor vectors in the passage index



NN Search: GPU Brute-Force

- Retrieving the top-1K from 9 million vectors is fast
 - We need to do 9M dot-products (a very big matrix multiplication) with 768 dim. vectors
- GPUs are made for this
 - Vectors must fit in GPU memory
 - 70ms latency / query
 - Incredible scale when increasing the batch size
 - Using a CPU this takes ~1 sec. / q

Table 1: Latency analysis of Top-1000 retrieval using our BERT_{DOT} retrieval setup for all MSMARCO passages using DistilBERT and Faiss (FlatIP) on a single Titan RTX GPU

Batch Size	Q. Encoding		Faiss Retrieval		Total	
	Avg.	99 th Per.	Avg.	99 th Per.	Avg.	99 th Per.
1	8 ms	11 ms	54 ms	55 ms	64 ms	68 ms
10	8 ms	9 ms	141 ms	144 ms	162 ms	176 ms
2,000	273 ms	329 ms	2,515 ms	2,524 ms	4,780 ms	4,877 ms

Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling; Hofstätter et al. SIGIR 2021
<https://arxiv.org/abs/2104.06967>

Approximate NN Search

- Brute-force search does not scale well beyond a couple of million vectors
- Fortunately, nearest neighbor search of vectors is a very common and broadly used technique in ML
 - We have many techniques and libraries to speed up search for sub-linear results
- Popular library: FAISS
 - Offering many algorithms (brute-force, inverted lists on clusters, HNSW, ...)
 - CPU and GPU supported
- Approximate search is another tradeoff between latency-effectiveness
 - We add a lot of complexity to the search system, but necessary for low-latency CPU serving

Production Support

- Dense retrieval is gaining more and more support in production systems
 - HuggingFace model hub gives us a common format to share models
 - Search engine must incorporate indexing & query encoding + provide nearest neighbor search
- Projects include Vespa.ai & Pyserini (integrates with Lucene)
 - Vespa provides deep integration of dense retrieval in common search features, such as filtering on properties
 - Important to filter during search, not after as to avoid empty result lists
 - Pyserini is a project focused on reproducing as many dense retrieval models as possible
 - Including easy hybrid search options between BM25 and DR

Other Uses for the BERT_{DOT} Model

- Semantic comparisons of all sorts:
 - Sentence, passage, document similarity -> all compressed into 1 vector
 - Recommendation models
- S-BERT (Sentence transformers) library provides many models & scenarios
 - Based on the HuggingFace transformer library
 - Offers many scenarios and built models out of the box
- Adaptions based on dot-product similarity also allow for multi-modal comparisons
 - For example: Encoding images and text in the same vector-space

Knowledge Distillation

An (almost) free lunch.

The Idea of Knowledge Distillation

- Most training data is noisy & not very fine-granular
 - In the case of MSMARCO we only have one labeled relevant per query
 - Might have a lot of false-negatives (positives that are not labeled)
- **Teacher:** Powerful – but slow – models can provide labels
 - Re-run inference on all training samples after the model is fully trained
 - Now we have fine-grained scores for all examples
- **Student:** We can use those new labels to train more efficient models
 - Try to find better weights, that produce a higher quality output

Different Levels of Supervision

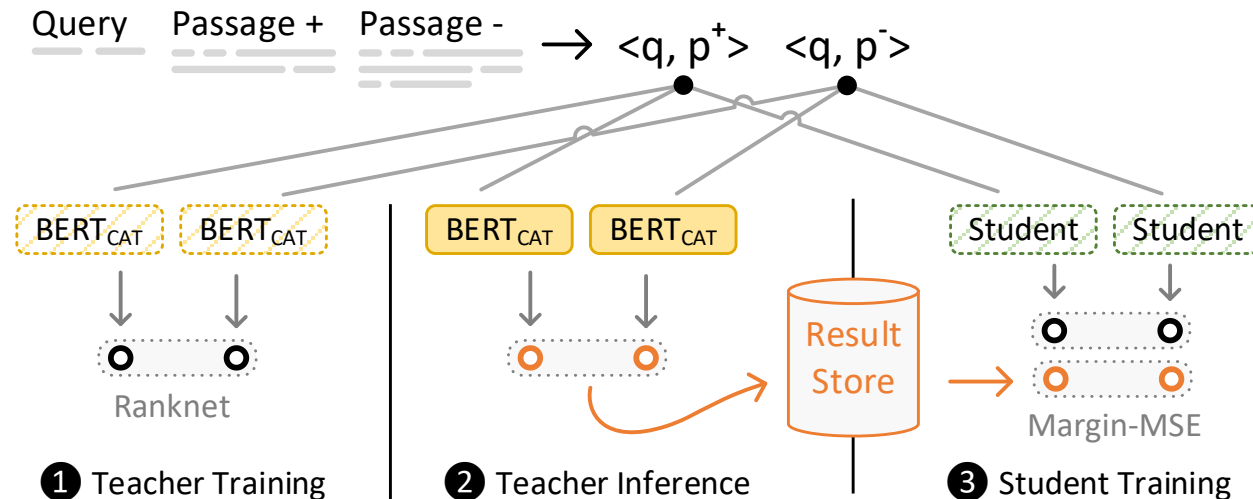
- We may use the final output scores (or class distribution, etc..) as supervision signal
 - This makes it possible to operate architecture independent
 - Easy to use an ensemble of teachers
- We also may use intermediate results in some or all layers as signal (f.e. activations, attention distribution)
 - This locks us into a certain architecture
 - Potentially also similar parameter settings
 - Much more supervision signals, than just using final score

DistilBERT

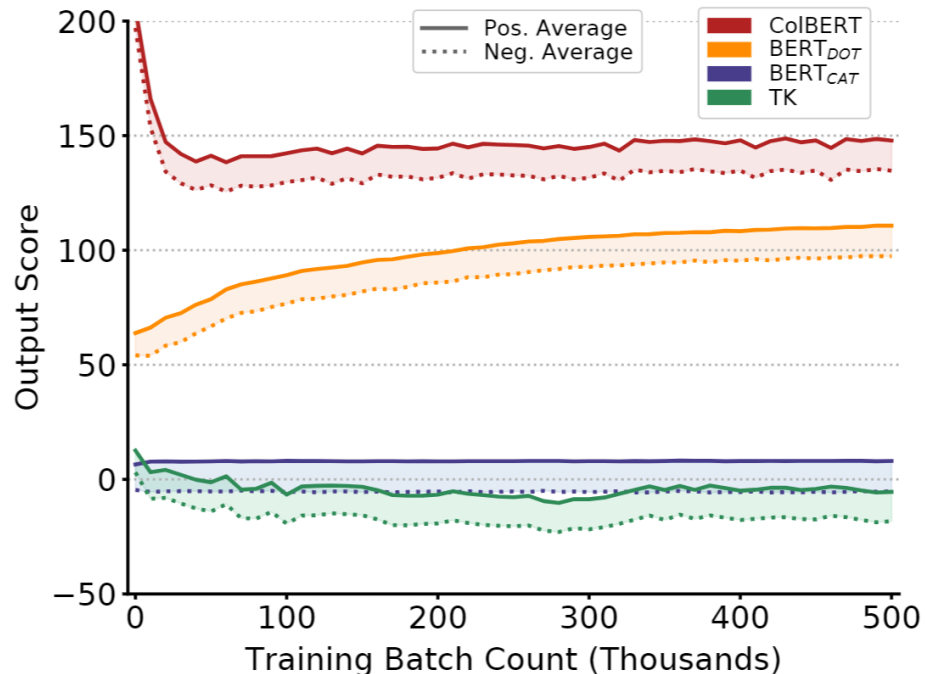
- A distilled, smaller version of the general-purpose BERT model
 - Shares the same vocabulary
 - Shares the general-purpose nature (ready to fine-tune)
- Size: 6 Layers with 768 output dimensions
- Trained with knowledge distillation, retains 97% of effectiveness
- Using DistilBERT as a base model in IR works very well
 - We consistently get good results for different architectures (including BERT_{DOT})
 - If we also apply knowledge distillation during the IR training, hardly a difference to larger BERT instances

Distillation in IR

- Train setup remains the same with triples
 - We first need to train a teacher on a binary loss
 - Then, get teacher scores from trained teacher (we can do that once)
 - Finally use those scores to train student models



Cross-Architecture Issues



- BERT_{CAT} is our most powerful architecture, but also the slowest -> good teacher
 - Ultimately, we want to make more efficient architectures more effective
 - This should work across architectures for the biggest impact
- We observe that different architectures converge to different scoring ranges
 - Only relative differences count for ranking
 - Directly optimizing scores, while still working, does not lead to optimal results

Margin-MSE Loss

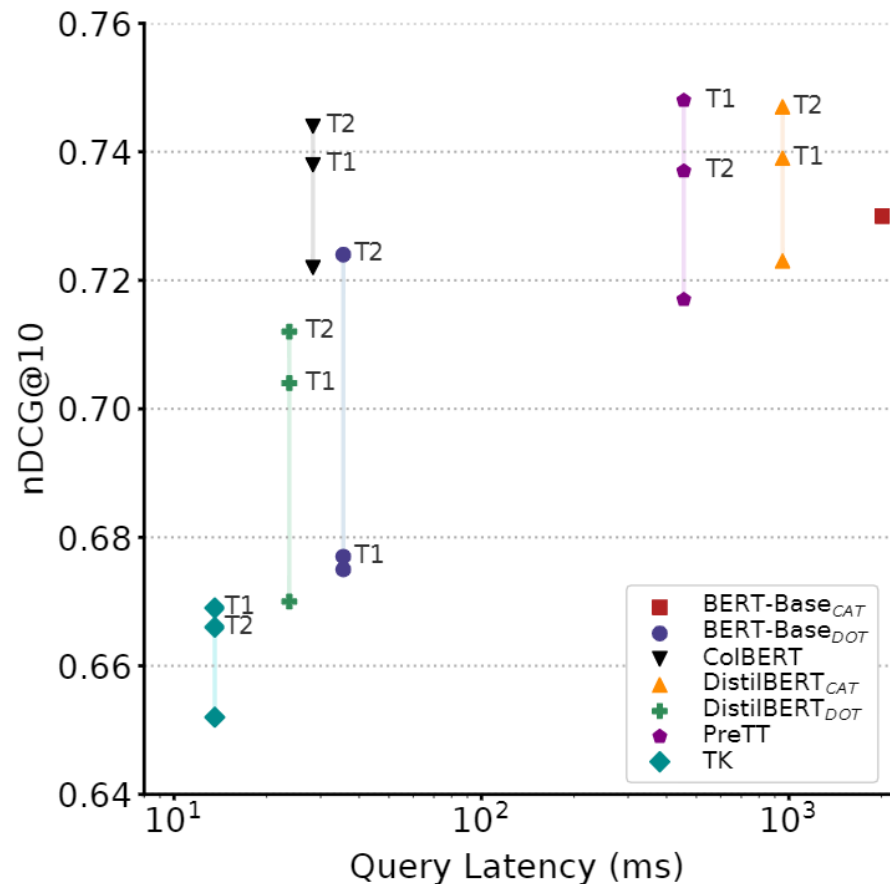
- We propose to optimize the margin between relevant and non-relevant sampled passages
 - Scoring ranges do not matter, only the relative differences

- Margin-MSE loss definition:

$$L(Q, P^+, P^-) = \text{MSE}(M_s(Q, P^+) - M_s(Q, P^-), \\ M_t(Q, P^+) - M_t(Q, P^-))$$

- Loss makes no assumption about the model architecture
 - We can mix and match different neural ranking models
- We can pre-compute the teacher scores once and re-use them

Cross-Architecture Distillation Re-Ranking



- Single Teacher (T1) already improves most efficient models
- Teacher Ensemble (T2) brings further improvements
 - ColBERT, PreTTR, and DistilBERT_{CAT} are even better than a single BERT-Base_{CAT}
- Query Latency does not change
 - As we only change weight values
 - With Margin-MSE we can shift the cost-effectiveness tradeoff only along the effectiveness axis

TREC-DL 2019; query latency (log-scale!) only shows time spent on re-ranking (with cached passage representations, where possible)

Margin-MSE Dense Retrieval

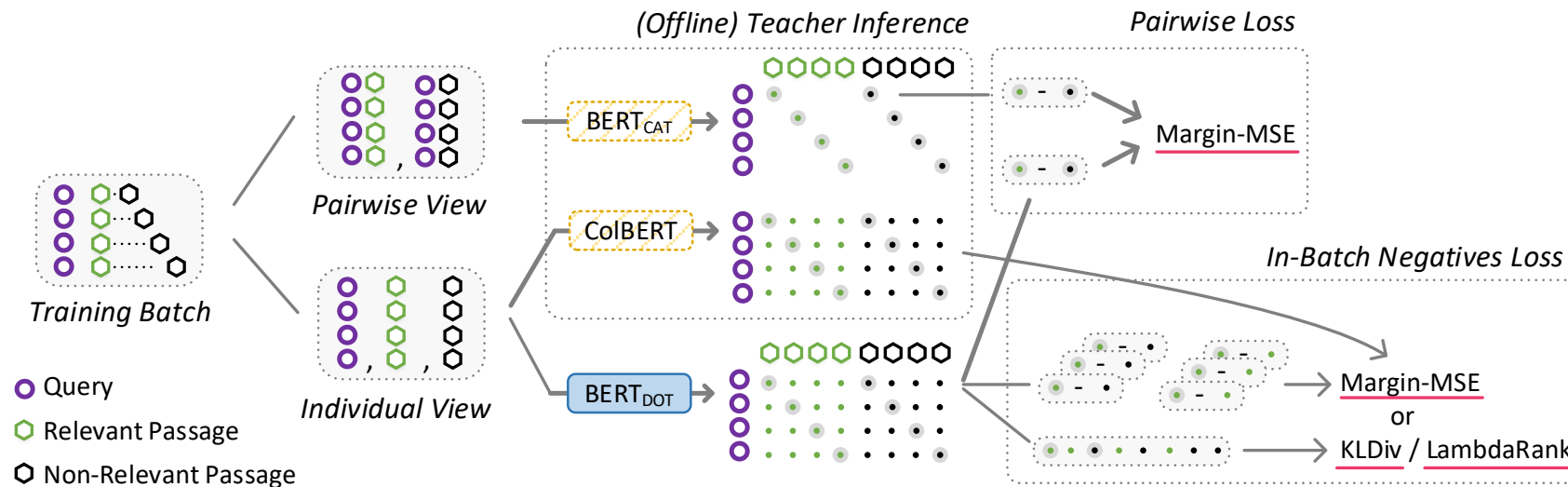
- Margin-MSE also possible for BERT_{DOT} retrieval
 - Even though we do nothing specific for dense retrieval
 - Results are quite respectable and close to much more complex and costly training approaches

Model	Index Size	Teach.	TREC DL Passages 2019			MSMARCO DEV		
			nDCG@10	MRR@10	Recall@1K	nDCG@10	MRR@10	Recall@1K
Baselines								
BM25	2 GB	–	.501	.689	.739	.241	.194	.868
BERT-Base _{DOT} ANCE [44]		–	.648	–	–	–	.330	.959
TCT-ColBERT [26]		–	.670	–	.720	–	.335	.964
RocketQA [12]		–	–	–	–	–	.370	.979
Our Dense Retrieval Student Models								
BERT-Base _{DOT}	12.7 GB	–	.593	.757	.664	.347	.294	.913
		T1	.631	.771	.702	.358	.304	.931
		T2	.668	.826	.737	.371	.315	.947
DistilBERT _{DOT}	12.7 GB	–	.626	.836	.713	.354	.299	.930
		T1	.687	.818	.749	.379	.321	.954
		T2	.697	.868	.769	.381	.323	.957

- Here, DistilBERT is even better than BERT-Base
- Again, T2 improves over T1 which improves over no teacher

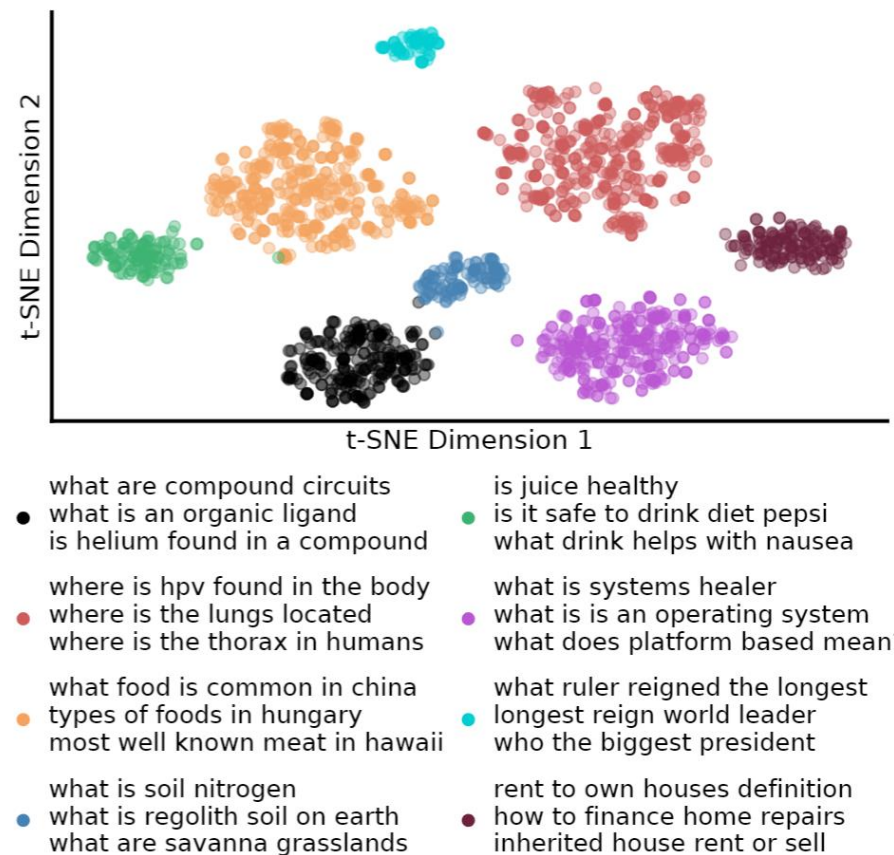
Dual Supervision

- Applying more teachers, specifically for dense retrieval training
- Pairwise Teacher: BERT_{CAT} ; In-batch-negatives: ColBERT
 - ColBERT can also use the individual view (less overhead)



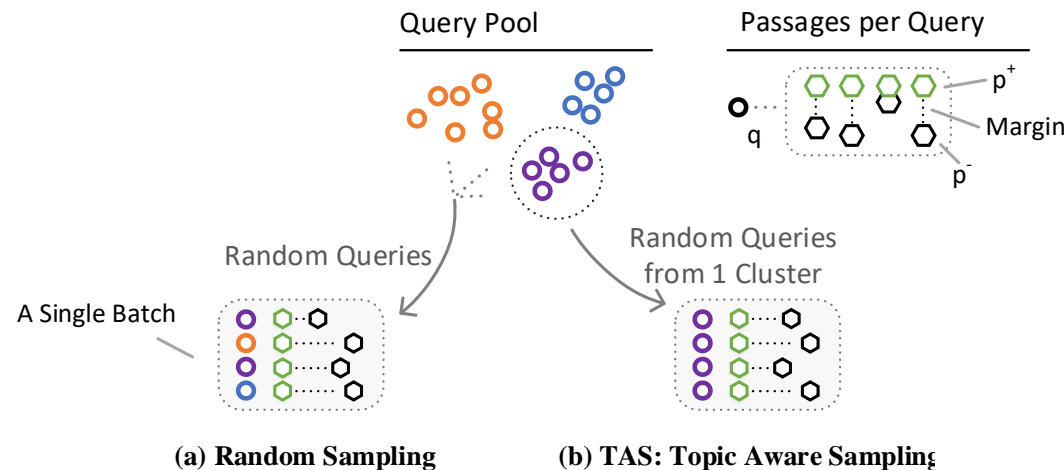
Improve In-Batch Negatives

- Random in-batch negatives provide little information / relevance signal
 - Other than this has nothing to do with each other
- In-batch negatives come from queries in a single batch
- We use a baseline model to cluster query representations
 - Query topic clusters are quite coherent (for MSMARCO we created 2K clusters, with ~200 queries each)



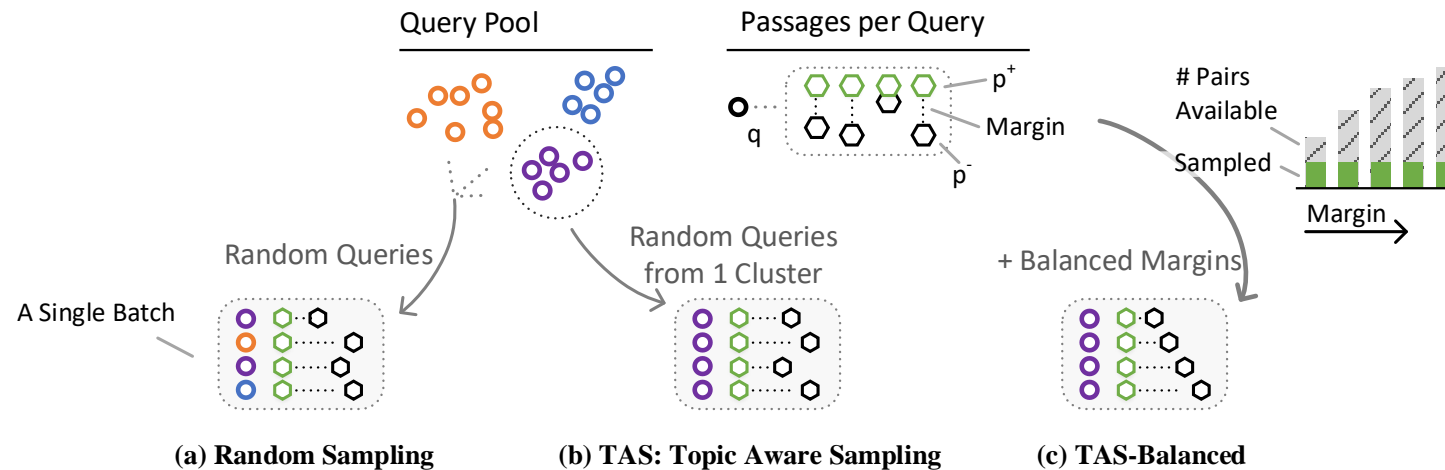
Topic Aware Sampling

- We propose TAS: Topic Aware Sampling
 - Instead of randomly sampled queries (a), we sample topically related queries out of a single cluster per batch (b)
 - Does not add any additional cost to the training process



TAS-Balanced

- In addition to TAS, we also balance pairs of relevant / non-relevant passage by binned margins per query
 - To down-sample large margin pairs (as they appear much more often)
 - Makes the pairwise training harder



TAS-Balanced Results

- TAS-Balanced improves with different teacher types
- Best results with dual supervision (pairwise + in-batch) + TAS-Balanced
 - Especially strong improvements on Recall@1K

Teacher	Sampling	TREC-DL'19			TREC-DL'20			MSMARCO DEV		
		nDCG@10	MRR@10	R@1K	nDCG@10	MRR@10	R@1K	nDCG@10	MRR@10	R@1K
None	Random	.602	.781	.714	.602	.782	.757	.353	.298	.935
Pairwise (BERT _{CAT})	Random	.687	.851	.767	.654	.812	.801	.385	.326	.958
	TAS	.677	.851	.769	.650	.820	.819	.385	.325	.957
	TAS-Balanced	.686	.866	.783	.665	.823	.825 ^r	.393 ^{rt}	.334 ^{rt}	.963 ^r
In-Batch Neg. (ColBERT)	Random	.680	.857	.745	.631	.773	.792	.372	.315	.951
	TAS	.706	.886	.799	.667 ^r	.821	.826 ^r	.396 ^r	.336 ^r	.968 ^r
	TAS-Balanced	.716	.910	.800	.677 ^r	.810	.820 ^r	.397 ^r	.338 ^r	.968 ^r
Pairwise + In-Batch	Random	.695	.891	.787	.673	.812	.839	.391	.331	.968
	TAS	.713	.878	.831	.689	.815	.862 ^r	.401 ^r	.338 ^r	.973 ^r
	TAS-Balanced	.712	.892	.845	.693	.843	.865^r	.402^r	.340^r	.975^{rt}

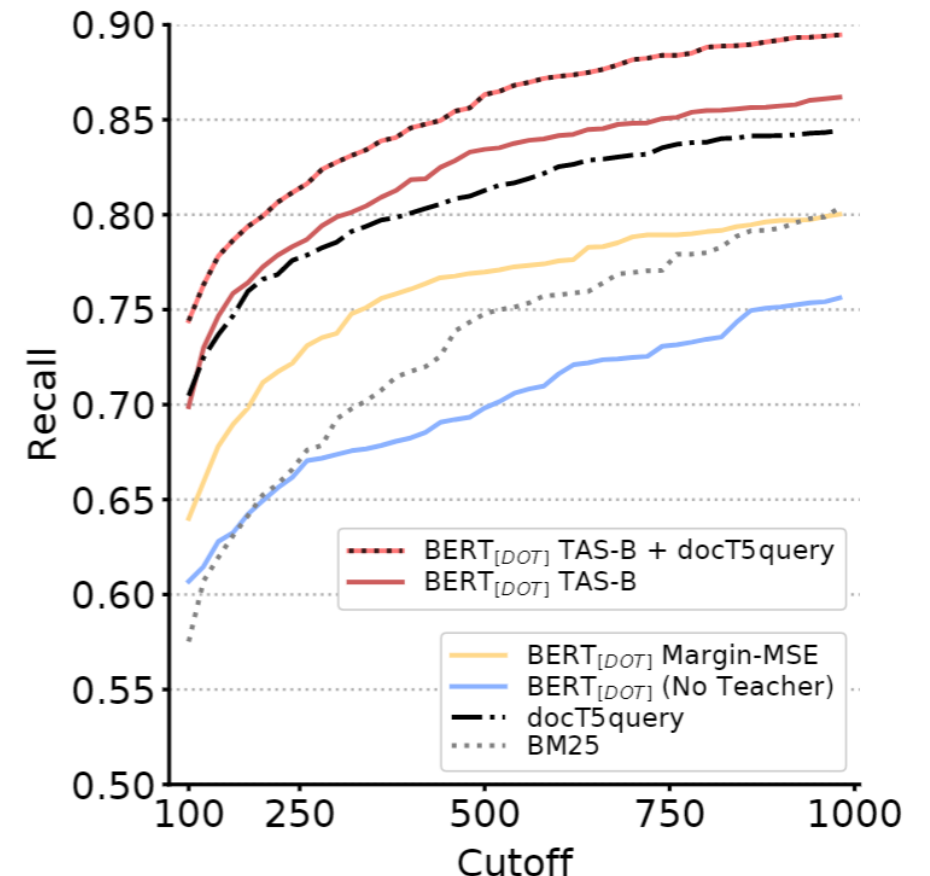
TAS-Balanced Results

- SOTA compared to other BERT_{DOT} training methods
 - We also find a strong influence of the training batch size on MSMARCO-DEV

Training Type	Encoder	L#	Batch Size	TREC-DL'19			TREC-DL'20			MSMARCO DEV		
				nDCG@10	MRR@10	R@1K	nDCG@10	MRR@10	R@1K	nDCG@10	MRR@10	R@1K
Baselines												
BM25	–	–	–	.501	.689	.739	.475	.649	.806	.241	.194	.868
[42] ANCE	BERT-Base	12	32	.648	–	–	–	–	–	–	.330	.959
[44] LTRe				.661	–	–	–	–	–	–	.329	.955
[44] ANCE + LTRe				.675	–	–	–	–	–	–	.341	.962
[11] RocketQA	ERNIE-Base	12	4,000	–	–	–	–	–	–	–	.364	–
			128	–	–	–	–	–	–	–	.309	–
[24] TCT	BERT-Base	12	96	.670	–	.720	–	–	–	–	.335	.964
TCT (ours)	DistilBERT	6	32	.680 ^b	.857 ^b	.745	.631 ^b	.773 ^b	.792	.372 ^b	.315 ^b	.951 ^b
[14] Margin-MSE	DistilBERT	6	32	.697	.868	.769	–	–	–	.381	.323	.957
				.687 ^b	.851 ^b	.767	.654 ^b	.812 ^b	.801	.385 ^{bt}	.326 ^{bt}	.958 ^{bt}
Ours												
TAS-Balanced	DistilBERT	6	32	.712 ^b	.892 ^b	.845 ^{btm}	.693 ^{btm}	.843 ^b	.865 ^{btm}	.402 ^{btm}	.340 ^{btm}	.975 ^{btm}
			96	.722 ^{btm}	.895 ^b	.842 tm	.692 ^{btm}	.841 ^b	.864 ^{btm}	.406 ^{btm}	.343 ^{btm}	.976 ^{btm}
			256	.717 ^{btm}	.883 ^b	.843 tm	.686 ^{btm}	.843 ^b	.875 ^{btm}	.410 ^{btm39}	.347 ^{btm39}	.978 ^{btm3}

TAS-Balanced Recall Results


- Previously, the densely judged TREC-DL query sets were very challenging for DR recall
 - Standalone and Margin-MSE not improving
- With TAS-Balanced we have the first DR training to outperform BM25 by a substantial margin on Recall
 - Same for TREC-DL '19 & '20
 - On every cutoff & binarization setting
 - Combined in a hybrid with docT5query even better



Analyzing Dense Retrieval

Finding out what works and what doesn't work.

BEIR Zero-Shot Benchmark

- Ultimately, we want DR models to be plug-n-play usable
 - This is referred to as zero-shot transfer
(because we use no training data from the target collection)
 - Zero-shot scenario is harder than in-domain evaluation
(because it solely tests generalization instead of a mix of memorization & generalization)
- BEIR  benchmark brings many IR collections in 1 format
 - Including framework to run HuggingFace models on all collections
 - Paper showed that many DR models struggle in zero-shot transfer
 - BM25 is more consistent

Why do DR Models Struggle on Zero-Shot?

- Possible explanations:

- 1 **Generalization:** DR models just don't generalize to other query distributions

- That would be bad, back to the drawing board

- 2 **Quirks:** MSMARCO training data contains too many quirks, specific to a collection

- Need adaption to training data

- 3 **Pool Bias:** Many (older or smaller) collections are heavily biased towards BM25 results

- Ultimately needs re-annotation campaigns



Chenyan Xiong
@XiongChenyan

Thanks for setting this up! Another factor is the hole rate. Many benchmarks' labels are pooled on sparse retrieval systems and have low coverage on DR models thus under-estimates DR accuracy. TREC-COVID has high coverage and DR models have higher scores there.



Nandan Thakur @Nthakur20 · Apr 20



New paper alert



BEIR: a heterogeneous benchmark for IR. 17 datasets, 9 tasks with diverse domains. 9 SOTA retrieval models evaluated in a zero-shot setup.

w/ @Nils_Reimers @arueckle @abhesrivas, IG at @UKPLab

pdf: arxiv.org/abs/2104.08663

More details, code 📄

#NLProc

[Show this thread](#)

See thread:

<https://twitter.com/XiongChenyan/status/1384594186683387905>

TAS-Balanced Zero-Shot Results

- More wins than losses against BM25 (and any other DR model) 🧪

Model	Size (MB)	Rank	<u>Average</u>	MSMARCO	TREC-COVID	NFCorpus	NQ	HotpotQA	FiQA	Signal-1M	TREC-NEWS	ArguAna	DBPedia	SCIDOCS	FEVER	Climate-FEVER	SciFact	Robust04
BM25	-	-	0.384	0.218	0.616	0.297	0.31	0.601	0.239	0.388	0.371	0.441	0.288	0.156	0.648	0.179	0.62	0.387
DistilBERT (TAS-B)	250	1	0.412	0.408	0.481	0.319	0.463	0.584	0.3	0.289	0.377	0.427	0.384	0.149	0.7	0.228	0.643	0.427
ANCE	500	2	0.379	0.388	0.654	0.237	0.446	0.456	0.295	0.249	0.382	0.415	0.281	0.122	0.669	0.198	0.507	0.392
DistilBERT	270	3	0.375	0.389	0.482	0.257	0.45	0.513	0.258	0.261	0.367	0.429	0.339	0.133	0.67	0.205	0.531	0.337
MiniLM-L-12	130	4	0.349	0.385	0.473	0.251	0.422	0.456	0.24	0.271	0.36	0.407	0.307	0.113	0.571	0.179	0.503	0.295
MiniLM-L-6	90	5	0.342	0.379	0.479	0.255	0.394	0.448	0.231	0.259	0.342	0.394	0.292	0.116	0.595	0.165	0.495	0.293
DPR (Multi)	500	6	0.252	0.177	0.332	0.189	0.474	0.391	0.112	0.155	0.161	0.175	0.263	0.077	0.562	0.148	0.318	0.252

Leaderboard snapshot from 14.5.2021 (we removed Touche-2020, as it is too small and results all over the place)

Open Science = Fast Science

- Incredible number of open tools & platforms used for TAS-B @ BEIR
 - Python, PyTorch, Faiss, NumPy, etc..
 - HuggingFace Library + Model Hub (free & open hosting)
 - Including pre-trained BERT/DistilBERT checkpoints
 - Code hosting on GitHub
 - Dissemination via arXiv, Twitter, Google Docs
- Timeline:
 - TAS-B accepted @ SIGIR'21 (Conf. in July), we pushed pre-print to arXiv in April
 - Including model checkpoint on HuggingFace
 - BEIR pre-print published concurrently in April
 - BEIR team integrated our checkpoint and ran all benchmarks by **mid May** 🧠 🤝

Lexical Matching

- DR models can potentially retrieve completely random passages
 - Via the unconstrained vector comparison, there is no guardrail to stop that
 - This a novel (potential) failure type (BM25 must have at least 1 lexical overlap)
- Our Analysis: Only a small problem, even smaller for TAS-B

Training	Common Tokens@1	Queries		Query Tokens		P@1	Δ P@1	Our Annotations 4-Graded Relevance Distribution
		#	%	Total	Subwords			
Standalone	0	163	0.33 %	5.0	0.8	.313	-.435	<div><div>21%</div><div>10%</div><div>13%</div><div>55%</div></div>
	1	1824	3.75 %	4.3	0.3	.580	-.168	<div><div>45%</div><div>13%</div><div>16%</div><div>26%</div></div>
TAS-Balanced	0	37	0.01 %	4.0	0.3	.622	-.173	<div><div>43%</div><div>19%</div><div>11%</div><div>27%</div></div>
	1	1284	2.64 %	3.9	0.1	.715	-.080	<div><div>57%</div><div>14%</div><div>14%</div><div>14%</div></div>

Annotation analysis of queries from MSMARCO-DEV-Large (49K queries) with zero or one subword overlap on the top-1 retrieved passage by the BERT_{DOT} dense retrieval model. Δ P@1 shows the difference to the DEV-7K set.

Can We Retrieve Unknown Acronyms?

- Yes, differentiating between acronyms works pretty good!

Passages									
❶ SIGIR	[CLS] si ##gi ##r is the premier international forum for the presentation of new research results and for the demonstration of new systems and techniques in information retrieval . the conference consists of five days of full papers , short papers , resource papers , demonstrations , tutor ##ials .								
❷ ECIR	[CLS] ec ##ir 2021 , the 43rd edition of the annual bc ##s - irs ##g european conference on information retrieval , initially planned to be a in - person conference in luc ##ca , due to co ##vid - 19 concerns will be held entirely online from march 28 to april 1 , 2021 .								
❸ ICTIR	[CLS] the ac ##m si ##gi ##r international conference on the theory of information retrieval (ict ##ir) is the premier conference of theoretical information retrieval (ir) . general information about ict ##ir is provided here . the conference welcome ##s papers on core ir and any paper on connections between ir and its neighboring disciplines .								
Query	Tokenized	S ❶	S ❷	S ❸	Query	Tokenized	S ❶	S ❷	S ❸
sigir	[CLS] si ##gi ##r	110.4	97.8	106.6	retrieval conference SIGIR	[CLS] retrieval conference si ##gi ##r	116.0	109.1	115.6
ecir	[CLS] ec ##ir	100.0	110.2	106.9	retrieval conference europe	[CLS] retrieval conference europe	106.9	112.7	107.9
ictir	[CLS] ict ##ir	103.2	105.1	112.0	retrieval conference ICTIR	[CLS] retrieval conference ict ##ir	109.3	113.9	118.3
sigr	[CLS] si ##gr	108.1	99.7	104.3	search engine conference	[CLS] search engine conference	104.5	104.2	102.0

Artificial examples showcasing lexical matching abilities of our BERT_{DOT} on IR conference acronyms
(We took the first paragraph from each conference website; not in the MSMARCO collection; acronyms not in the MSMARCO training data)

Are There Still Cases where BM25 is Better?

- Yes, although its increasingly hard to find them
 - We need to be careful with comparisons based on sparse judgements (easy to miss a false negative, when looking at it on a per-query basis)
 - We used sparse judgements as filter for selecting what we should annotate in a fine-grained way
 - Found few queries where sparse judgements tell us that BM25 is much better
 - Turns out it is *mostly* an artefact of sparse judgments, so we need keep looking 🙄🙄

Training	Queries		Query Tokens		P@1	Our Annotations			
	#	%	Total	Subwords		4-Graded Relevance Distribution			
Standalone	911	1.87%	7.5	1.0	.528	37%	16%	28%	19%
TAS-Balanced	472	0.97%	7.7	0.9	.638	48%	16%	27%	9%

Annotation analysis of queries selected from MSMARCO-DEV-Large (49K queries) with BM25 having an RR=1 (P@1 =1) and dense models RR=0 on sparse judgements. We annotated the top-1 retrieved result from the dense models.

Our Analysis Take Away

- Well trained (TAS-B) dense retrieval models are good at lexical matching
 - It is a common myth that they are bad at it
 - *Well trained* is the key here, random initialization would break it
 - This statement only applies to MSMARCO and in-domain training
 - Might be more of a problem in zero-shot transfer scenarios
 - Unknown acronyms get picked up by the model in both query and passage
- On MSMARCO, DR models are already so good, it is quite hard to find large slices of the query distribution where BM25 is much better
 - We couldn't find systematic patterns

Summary: Dense Retrieval & Knowledge Distillation

- 1 Dense retrieval is a promising direction for the future of search
- 2 Knowledge distillation from strong teachers helps DR a lot
- 3 Topic aware sampling represents the current SOTA in DR training

- 1 Dense retrieval is a promising direction for the future of search
- 2 Knowledge distillation from strong teachers helps DR a lot
- 3 Topic aware sampling represents the current SOTA in DR training

Thank You