A tale of two encoders for neural retrieval

Aditya Krishna Menon

Sep 5th, 2024

Google Research

About me

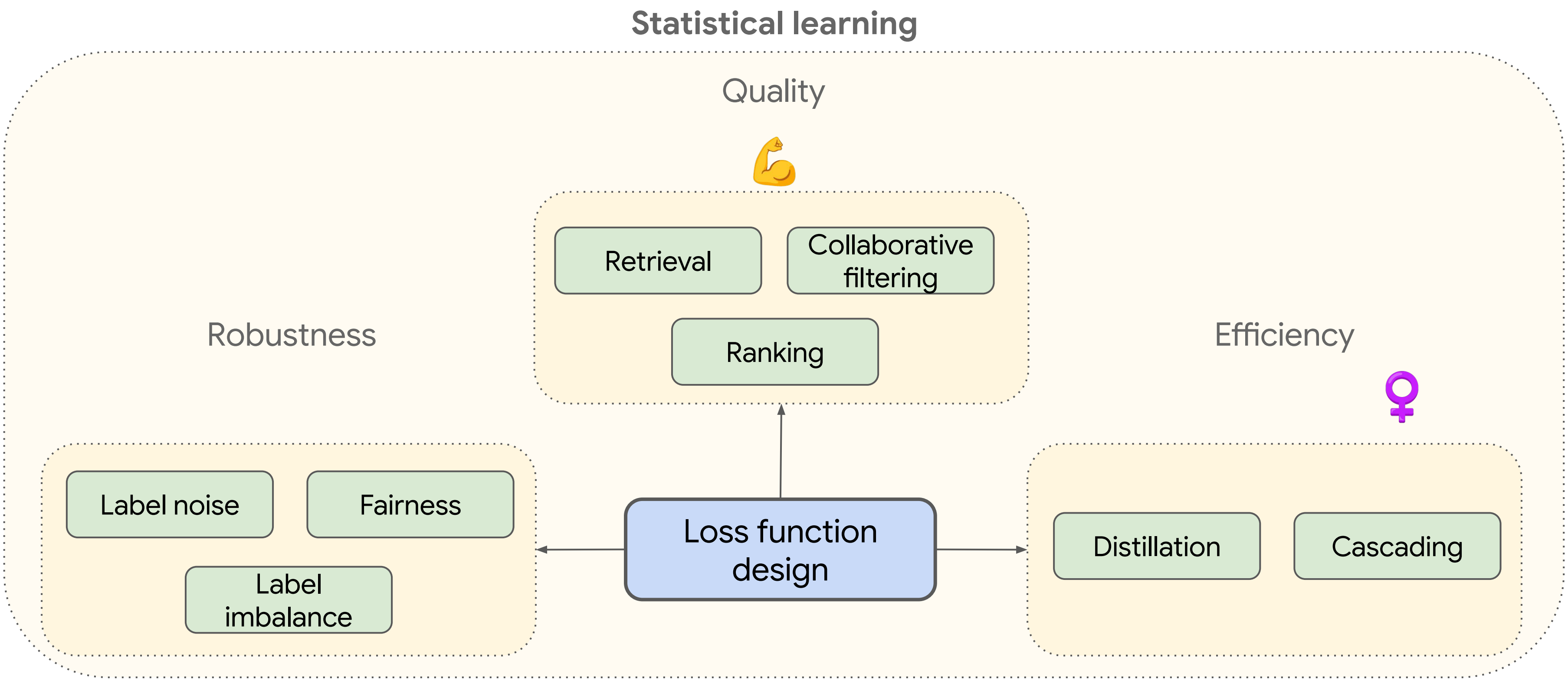
Research Scientist at Google NYC

Working on machine learning algorithm design and analysis

Past lives:

- University of Sydney
- UC San Diego
- NICTA / CSIRO Data61 / Australian National University

About my work



About this talk

Summary of (some of) our team's (+ collaborators') work on neural retrieval

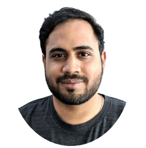
Ankit Singh Rawat

Andreas Veit

Felix Yu

Himanshu Jain

Manzil Zaheer



Rama Pasumarthi

Rob Fergus

Sadeep Jayasumana

Sanjiv Kumar

Sashank Reddi

Seungyeon Kim

Veeru Sadhanala

Wittawat Jitkrittum

Ziwei Ji

Agenda

01 A (neural) retrieval primer

02 Limits of dual encoders

03 Unified cross & dual encoders

04 Hybrid cross & dual encoders

05 Conclusion & future work

Information retrieval

- Given a query, and an item corpus, find the *k* most relevant items

Retrieval phase

- Typically, we first retrieve a set of candidate items

“books with sad endings”

Re-ranking phase

- We then re-rank these items to obtain the final results

Re-ranking phase

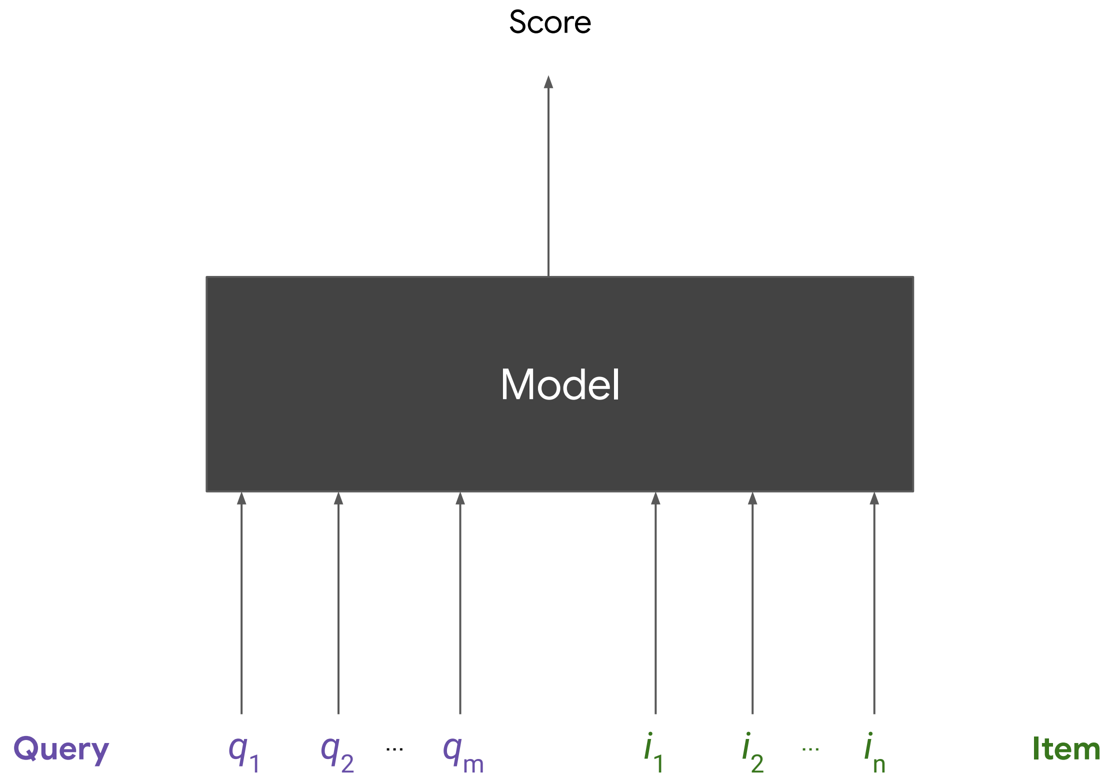
- We then re-rank these items to obtain the final results

Re-ranking phase

- We then re-rank these items to obtain the final results

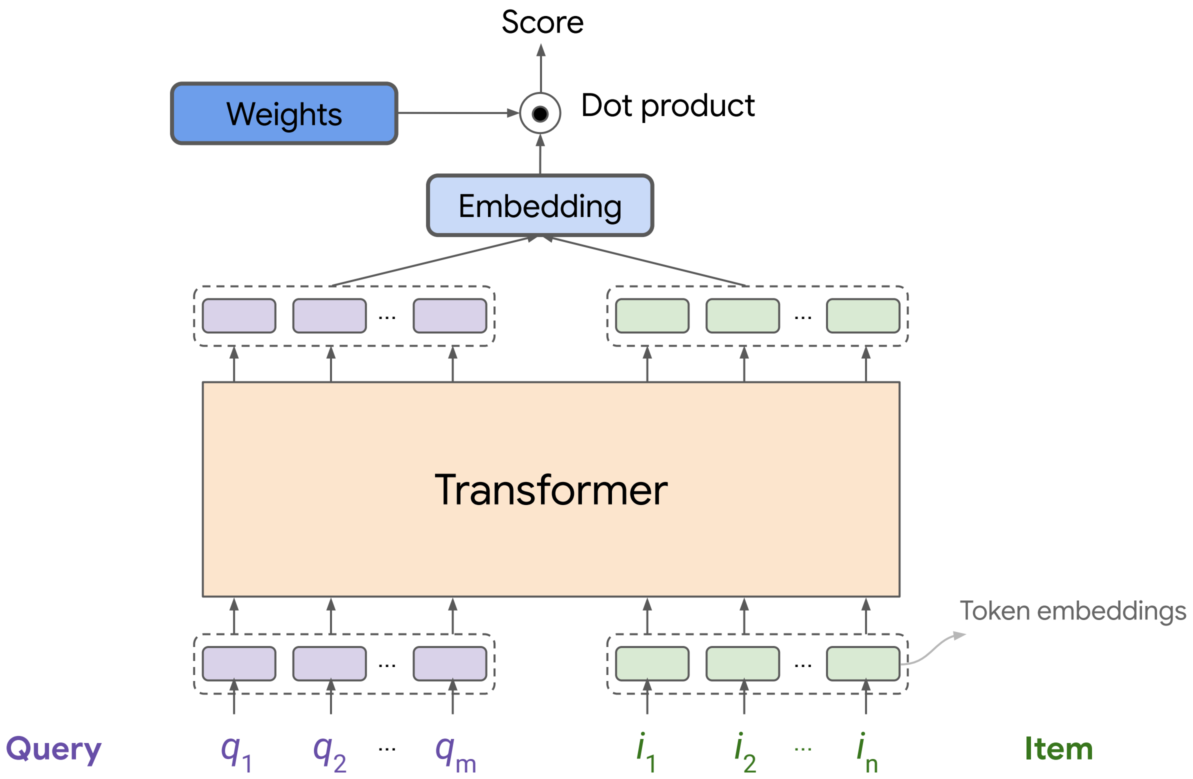
Encoder-based models

- In both phases, we need to score (Query, Item) affinity



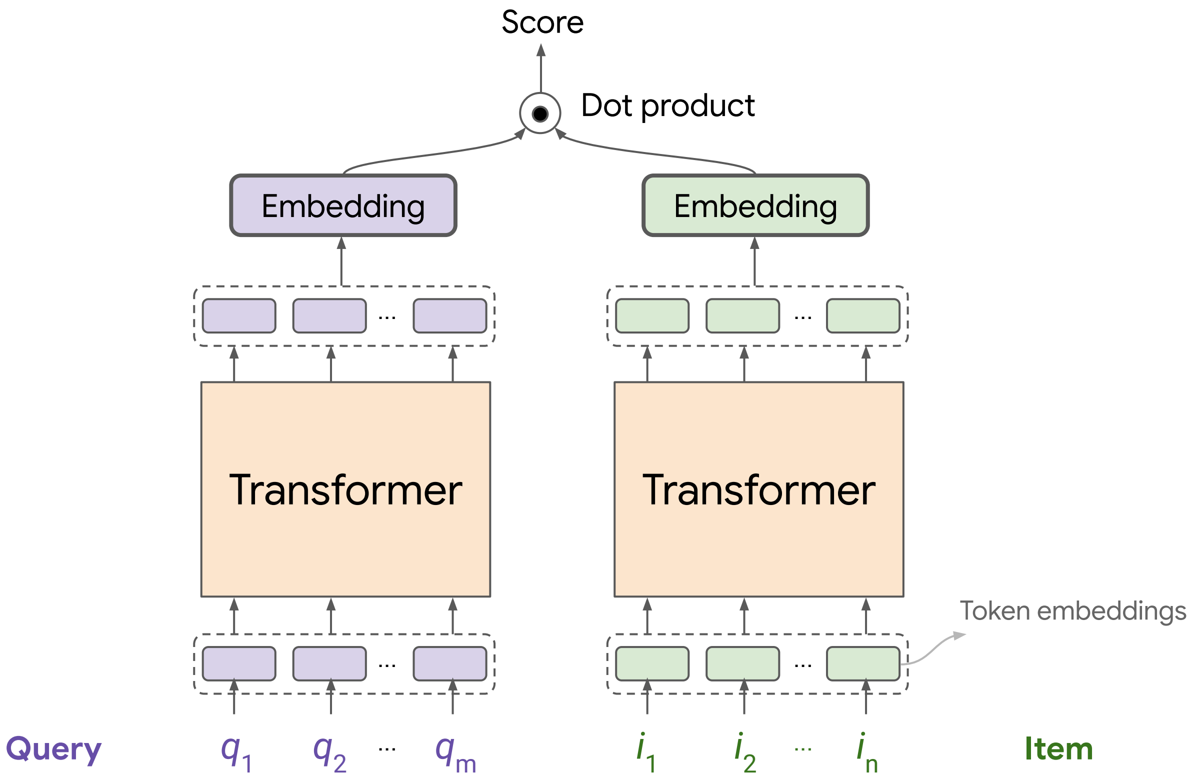
Cross-encoders

- Cross-encoders jointly embed queries and items, and project the embedding



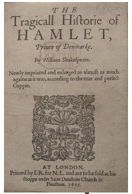
Dual-encoders

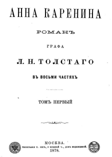
- Dual-encoders separately embed queries and items, and measure embedding similarity



Encoder training

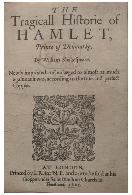
- Each query may have one or more associated positive items
  - Natively, a (featurised) multi-label learning problem

("books with sad endings" , { 

, 

})

- Can create a set of multi-class labels for each positive
  - Now amenable to, e.g., softmax cross-entropy
  - Key challenge becomes suitable negative mining

("books with sad endings" , 

)

("books with sad endings" , 

)

Multilabel reductions: what is my loss optimising? Menon et al. NeurIPS 2019.

Google

Cross- versus dual-encoders

Dual-encoders are highly efficient for retrieval; cross-encoders inapplicable!

Dual-encoders tend to underperform for re-ranking

	MSMARCO re-rank		TREC DL19 re-rank		NQ re-rank	
Model	**MRR**	**nDCG**	**MRR**	**nDCG**	**MRR**	**nDCG**
Cross-attention BERT (12-layer)	0.370	0.430	0.829	0.749	0.746	0.673
Dual-encoder BERT (6-layer)	0.310	0.360	0.834	0.677	0.676	0.601

Maintain separate retrieval and re-ranking models

Passage Re-ranking with BERT. Nogueira and Cho. arXiV 2019.
Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation. Hofstätter et al. arXiV 2020.

Cross- versus dual-encoders

Dual-encoders are highly efficient for retrieval; cross-encoders inapplicable!

Dual-encoders tend to underperform for re-ranking

Model	[illegible]		[illegible]		NQ re-rank	
	[illegible]	[illegible]	[illegible]	[illegible]	MRR	nDCG
Cross-attention BERT (12- [illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.746	0.673
Dual-encoder BERT (6-layer)	0.310	0.360	0.834	0.677	0.676	0.601

Is there more to the story?

Maintain separate retrieval and re-ranking models

Passage Re-ranking with BERT. Nogueira and Cho. arXiV 2019.
Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation. Hofstätter et al. arXiV 2020.

Agenda

01 A (neural) retrieval primer

02 Limits of dual encoders

03 Unified cross & dual encoders

04 Hybrid cross & dual encoders

05 Conclusion & future work

Cross- versus dual-encoders

Dual-encoders tend to underperform for re-ranking

Model	MSMARCO re-rank MRR	MSMARCO re-rank nDCG	TREC DL19 re-rank MRR	TREC DL19 re-rank nDCG	NQ re-rank MRR	NQ re-rank nDCG
Cross-attention BERT (12-layer)	0.370	0.430	0.829	0.749	0.746	0.673
Dual-encoder BERT (6-layer)	0.310	0.360	0.834	0.677	0.676	0.601

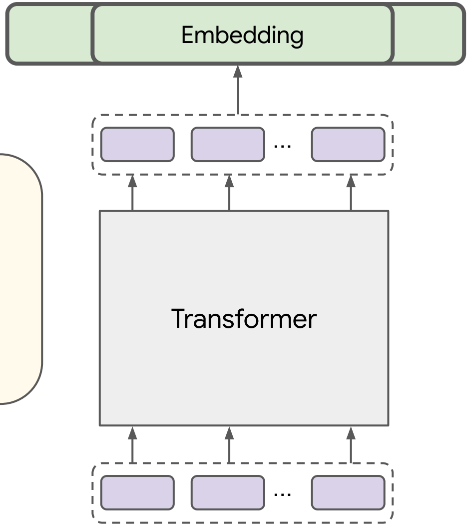
Why does this happen?

- Inherent capacity limit?
- Limitations of training procedure?
- ...

Capacity of dual-encoders: theory

- Can dual-encoders fit any (reasonable) relevance function?

Proposition. Under mild technical conditions, any continuous query-item score function $s(q, i)$ can be approximated by some $Z(q)^{\top} W(i)$, where $Z(q)$, $W(i)$ have at most countably infinite dimension.



In defense of dual-encoders for neural ranking. Menon et al. ICML 2022.

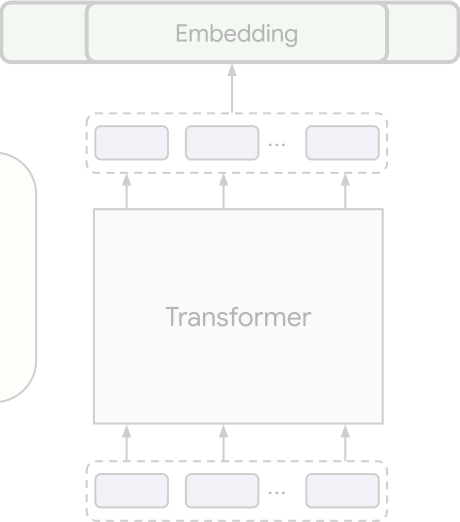
Capacity of dual-encoders: theory

- Can dual-encoders fit any (reasonable) relevance function?

Prop

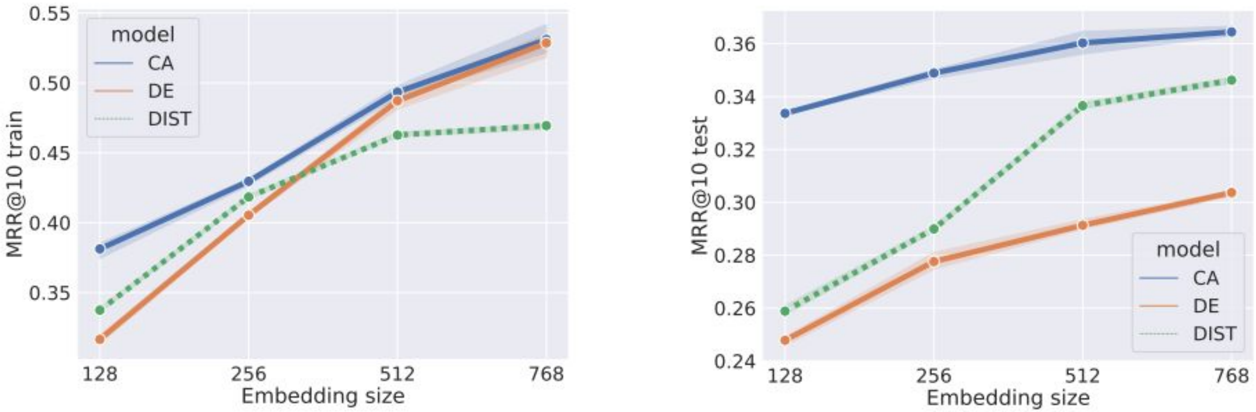
Do we see this in practice?

continuous query-item score function $s(q, i)$ can be approximated by some $Z(q)^T W(i)$, where $Z(q)$, $W(i)$ have at most countably infinite dimension.



Capacity of dual-encoders: practice

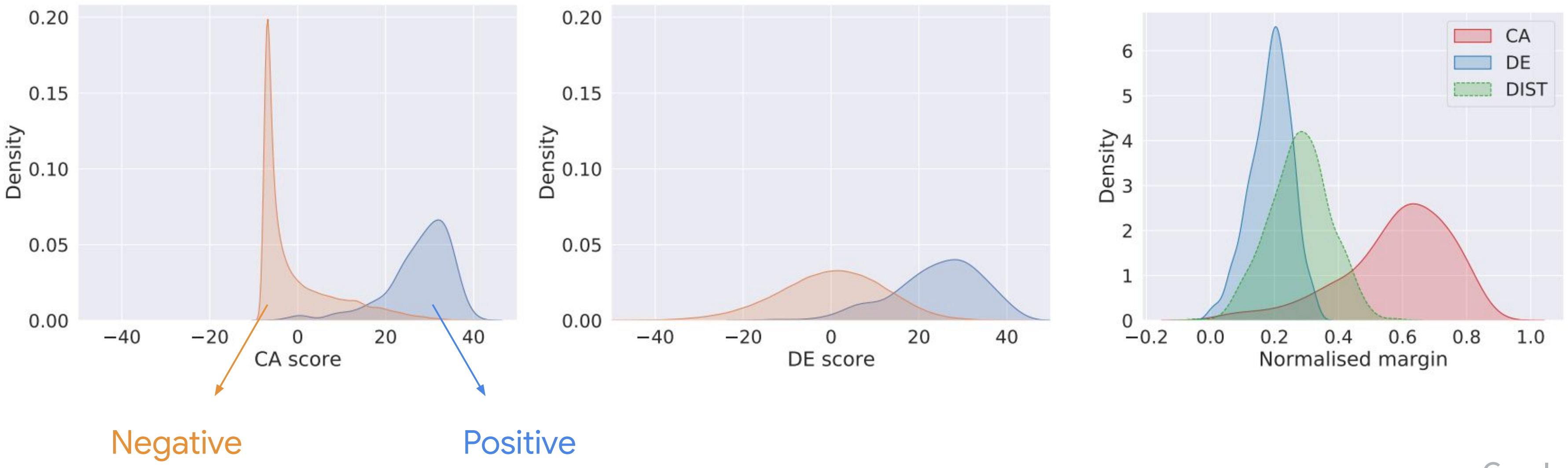
- With large embedding size, dual-encoders work well on **training** set!



BERT-based encoders on
MSMARCO

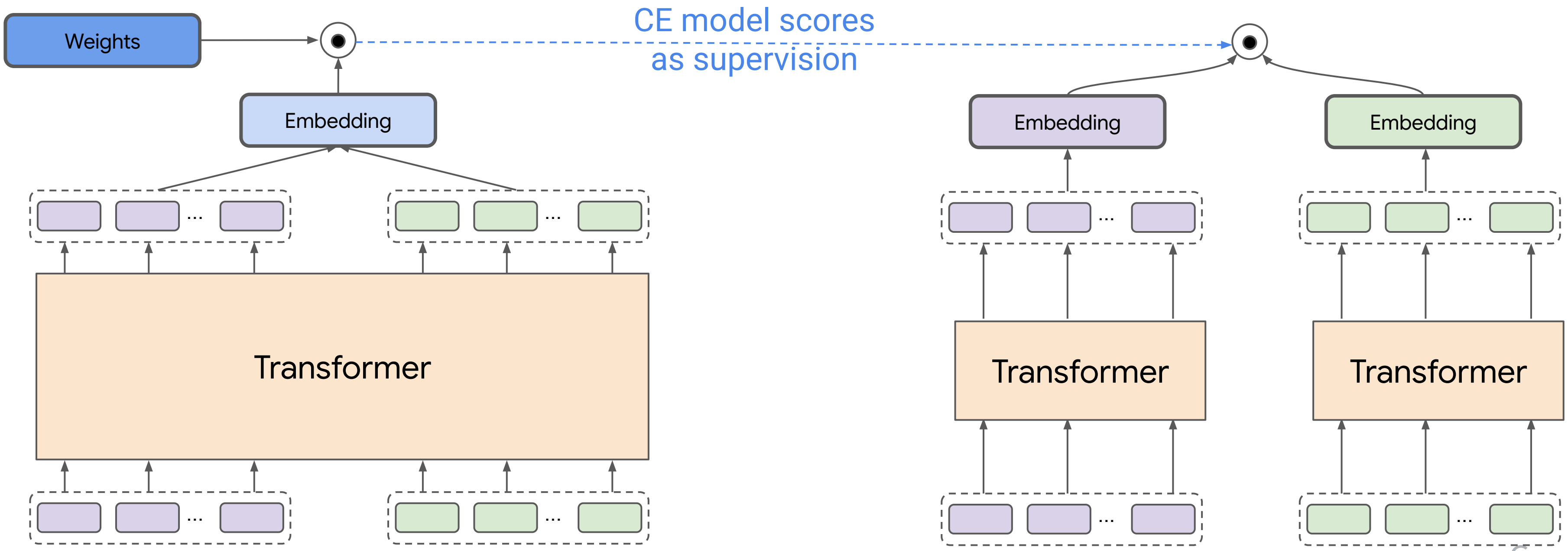
Why is there a generalisation gap?

- Dual-encoders tend to yield poorer margins
    - i.e., poorer gaps between score on positive and negative items



How can we mitigate the generalisation gap?

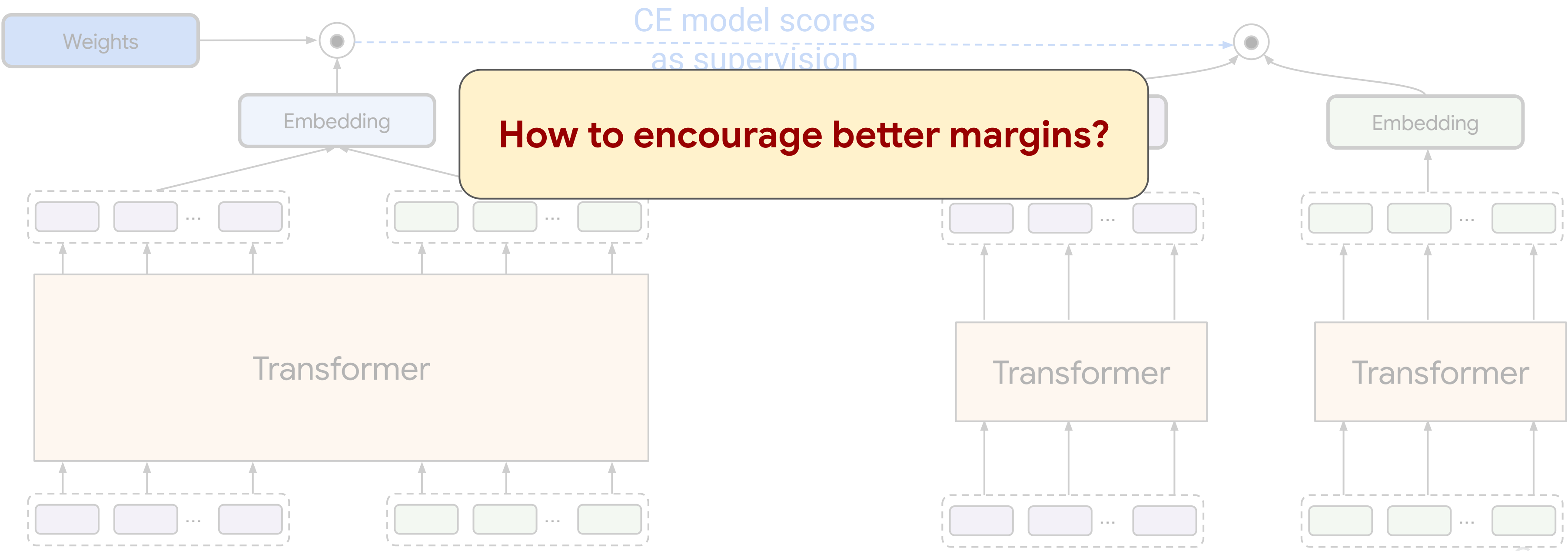
- Distill predictions from a cross-encoder “teacher” to dual-encoder “student”



Distilling knowledge from reader to retriever for question answering. Izacard and Grave. arXiV 2020.

How can we mitigate the generalisation gap?

- Distill predictions from a cross-encoder "teacher" to dual-encoder "student"



How to encourage better margins?

Distilling knowledge from reader to retriever for question answering. Izacard and Grave. arXiV 2020.

Distillation via multi-margin MSE (M^{3}SE)

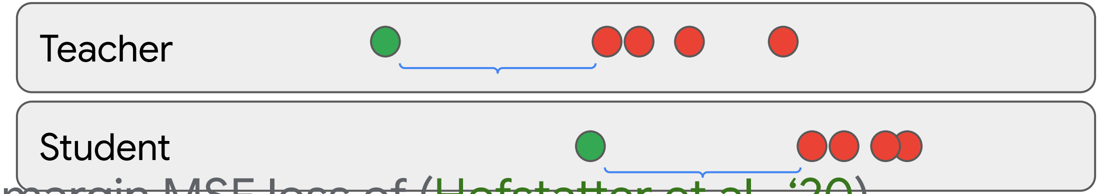
- Encourage matching teacher margin on positives *P*:

Teacher score

Student score

$$\ell_{\mathrm{m3se}}(\mathbf{t}, \mathbf{s}) = \sum_{i \in P} \left((t_i - t_{j^*}) - (s_i - s_{j^*}) \right)^2 + \sum_{j \in N} [s_j - s_{j^*}]_+^2$$

Highest scoring negative



- Generalises margin MSE loss of (Hofstatter et al., ‘20)
    - For a single positive and negative, limiting case of softmax cross-entropy

Distillation via ranking matching

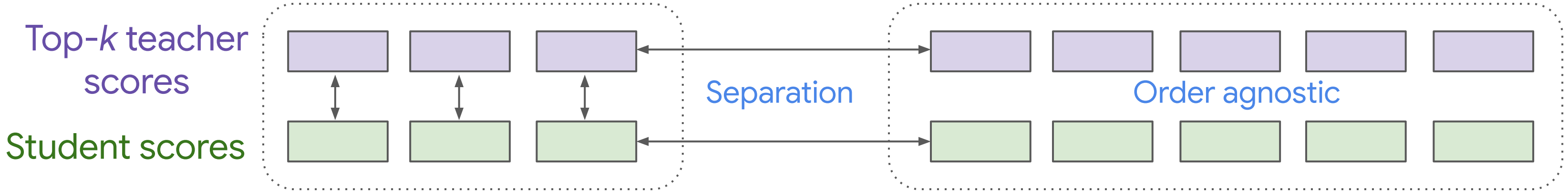
- More generally, we may seek to match teacher's ranking over top-*k* items
- Several versions of RankDistil objective possible:

Multi-class loss

$$\ell_{\textsc{RankDistil}}(t, s, P, N) = \Psi(t, s, P) + \sum_{i \in N} \varphi(-s_i),$$

Binary loss

$$\ell_{\textsc{RankDistil}}(t, s, P, N) = \Psi(t, s, P) + \sum_{i \in N} \sum_{j \in P} \varphi(s_j - s_i)$$



RankDistil: knowledge distillation for ranking. Reddi et al. AISTATS 2021.

Empirical results for re-ranking

- Distillation can help mitigate the generalisation gap!

Model	MSMARCO re-rank MRR	MSMARCO re-rank nDCG	TREC DL19 re-rank MRR	TREC DL19 re-rank nDCG	NQ re-rank MRR	NQ re-rank nDCG	
One-hot models							
BM25 (Robertson & Zaragoza, 2009)	0.194†	0.241†	0.689†	0.501†	—	—	
ANCE (Xiong et al., 2021)	—	—		—	0.677†	—	—
Cross-attention BERT (12-layer)	0.370	0.430	0.829	0.749	0.746	0.673	
Dual-encoder BERT (6-layer)	0.310	0.360	0.834	0.677	0.676	0.601	
Distilled dual-encoders							
MSE (Hofstätter et al., 2020a)	0.289	0.343	0.781	0.693	0.659	0.591	
Margin MSE (Hofstätter et al., 2020a)	0.334	0.392	0.867$^\diamond$	0.718	0.673	0.594	
RankDistil-B (Reddi et al., 2021)	0.249	0.301	0.852	0.708	0.649	0.561	
Softmax CE (Equation 1)	0.346	0.405	0.846	0.726$^\diamond$	0.682	0.607	
M^{3}SE (Equation 4)	0.349	0.406	0.852	0.714	0.699	0.625	

Cross- versus dual-encoders

Dual-encoders tend to underperform for re-ranking

Model	MSMARCO re-rank		TREC DL19 re-rank		NQ re-rank	
	MRR	nDCG	MRR	nDCG	MRR	nDCG
Cross-attention BERT (12-layer)	0.370	0.430	0.829	0.749	0.746	0.673
Dual-encoder BERT (6-layer)	0.310	0.360	0.834	0.677	0.676	0.601

Why does this happen?

- Poorer margins
- Expressivity with small dimension

What can we do about it?

- Distillation

Cross- versus dual-encoders

Dual-encoders tend to underperform for re-ranking

Model	MSMARCO re-rank		TREC DL19 re-rank		NQ re-rank	
	MRR	nDCG	MRR	nDCG	MRR	nDCG
Cross-attention BERT (12-[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.746	0.673
Dual-encoder BERT (6-lay[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.676	0.601

Can we make deeper changes?

Why does this happen?

- Poorer margins
- Expressivity with small dimension

What can we do about it?

- Distillation

Agenda

01 A (neural) retrieval primer

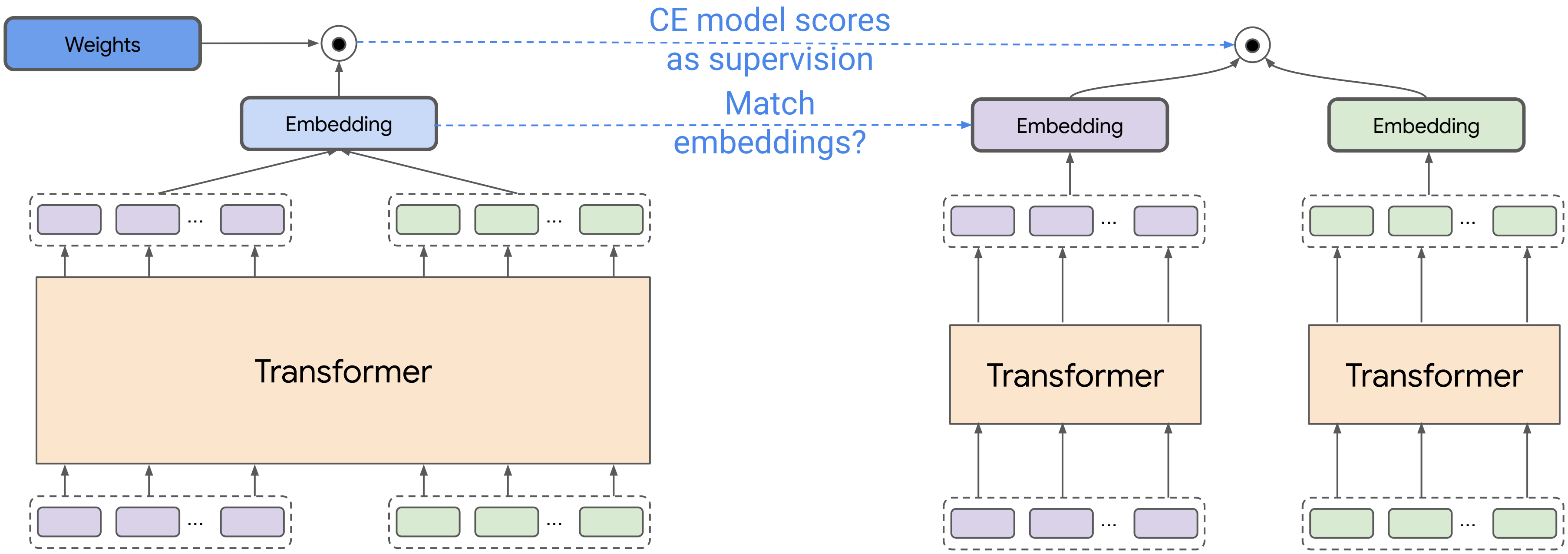
02 Limits of dual encoders

03 Unified cross & dual encoders

04 Hybrid cross & dual encoders

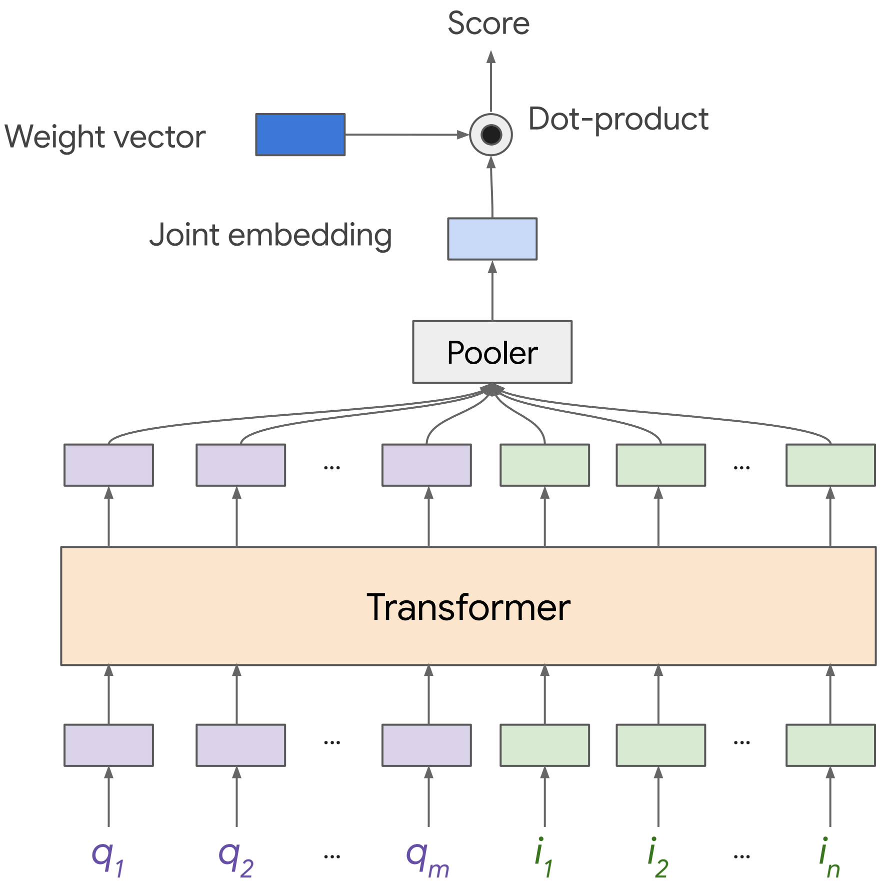
05 Conclusion & future work

Cross- to dual-encoder distillation



Romero et al. Fitnets: Hints for thin deep nets. arXiV, 2014.

Cross-encoder embeddings: a closer look



Do joint embeddings capture semantic structure?

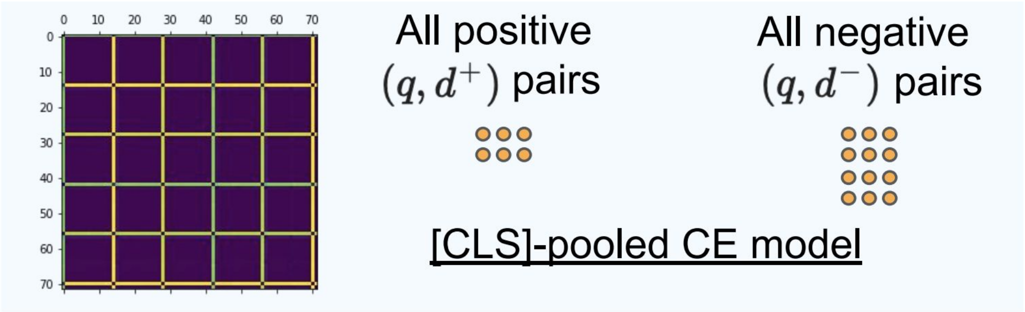
Query + item tokens

The perils of (naïve) pooling

- Cross-encoder training seeks to align embeddings of:
  - Positive pairs with some (learned) weight vector *w*
  - Negative pairs with some (learned) weight vector -*w*

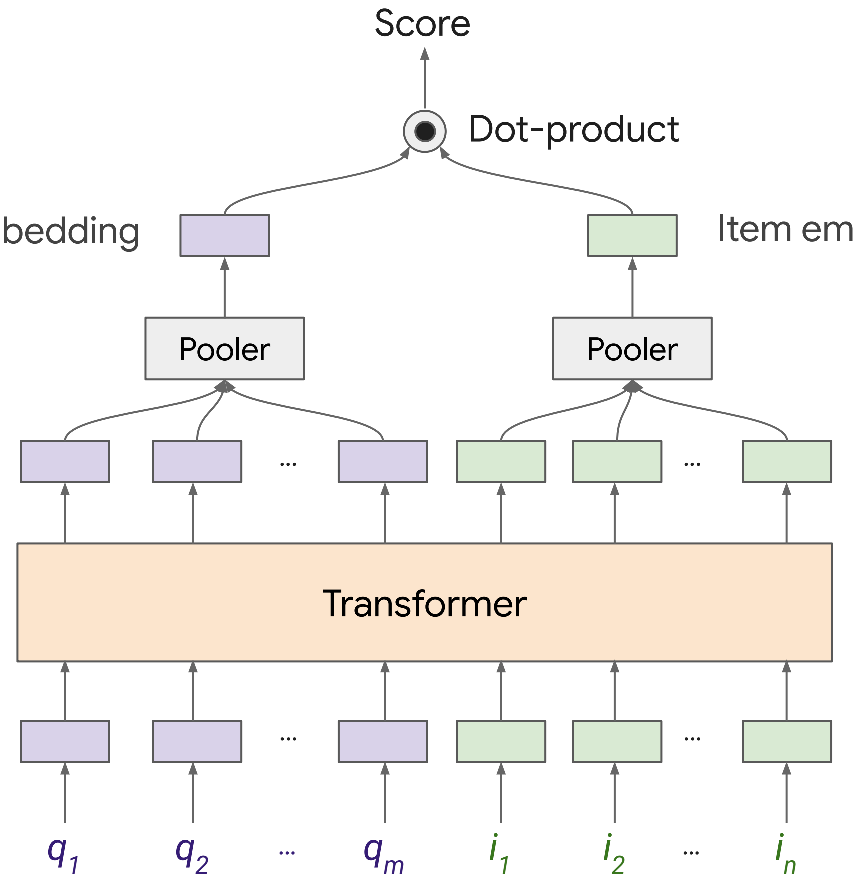
- Joint embeddings tend to not capture semantic structure!
  - No explicit coupling amongst embeddings within a group

Pairwise distance matrix



[CLS]-pooled

The dual pooling trick



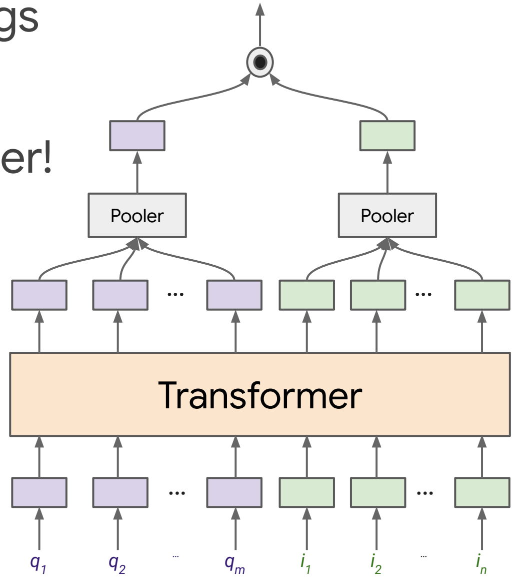
Separately pool query and item tokens!

Query + item tokens

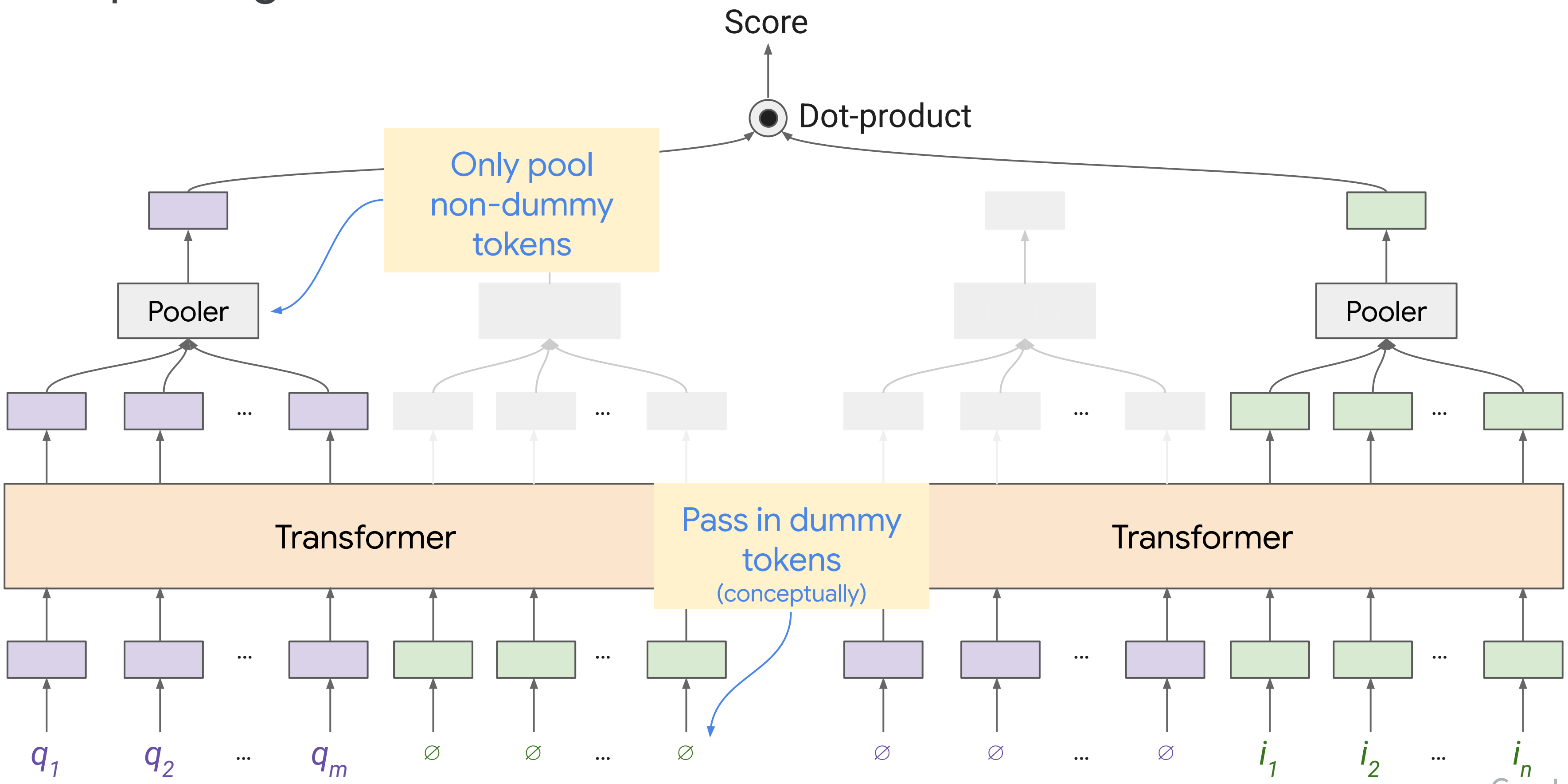
Yadav et al. Efficient Nearest Neighbor Search for Cross-Encoder Models using Matrix Factorization. EMNLP 2022.

Dual pooling = dual encoder?

- Dual pooling produces separate query and item embeddings
- However, these involve joint processing through the encoder!
  - Not suitable for use as a dual encoder!
  - Cannot use this for efficient query ⟶ item search
- Need to separately process queries and items...



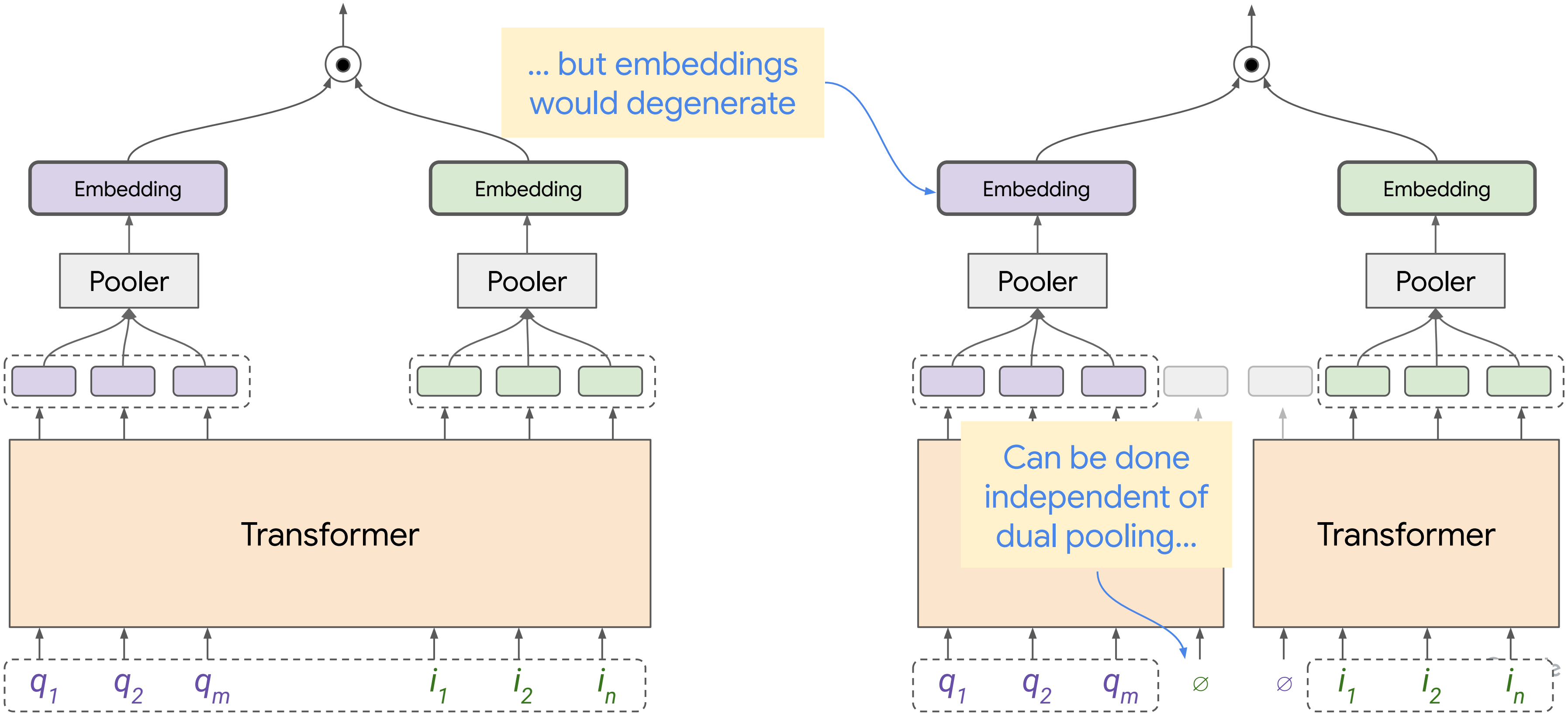
Dual pooling for dual-encoders



Unified Single-model Training Achieving Diverse Scores for Information Retrieval. Kim et al. ICML 2024.

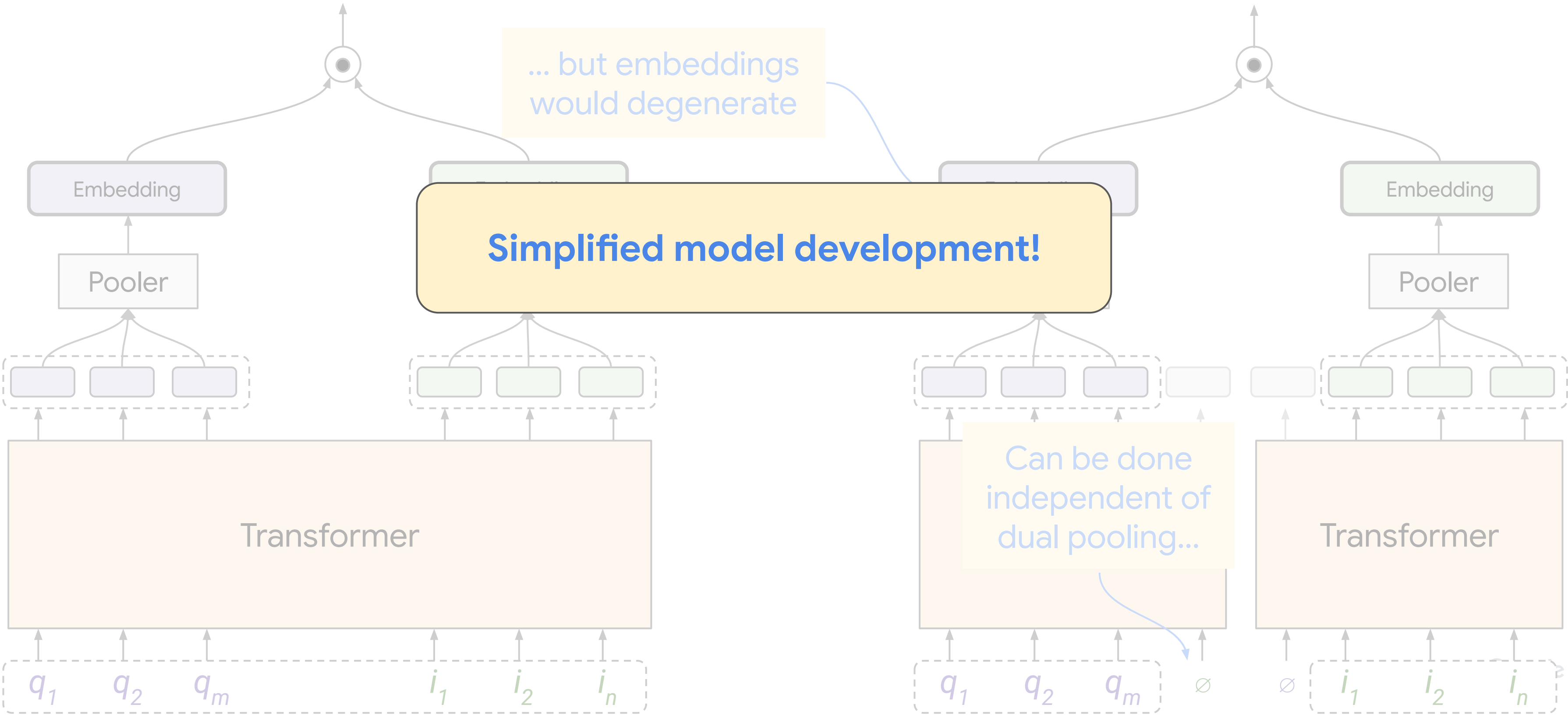
USTAD: unified cross- and dual-encoder

- Re-use same Transformer for both cross- and dual-encoder!



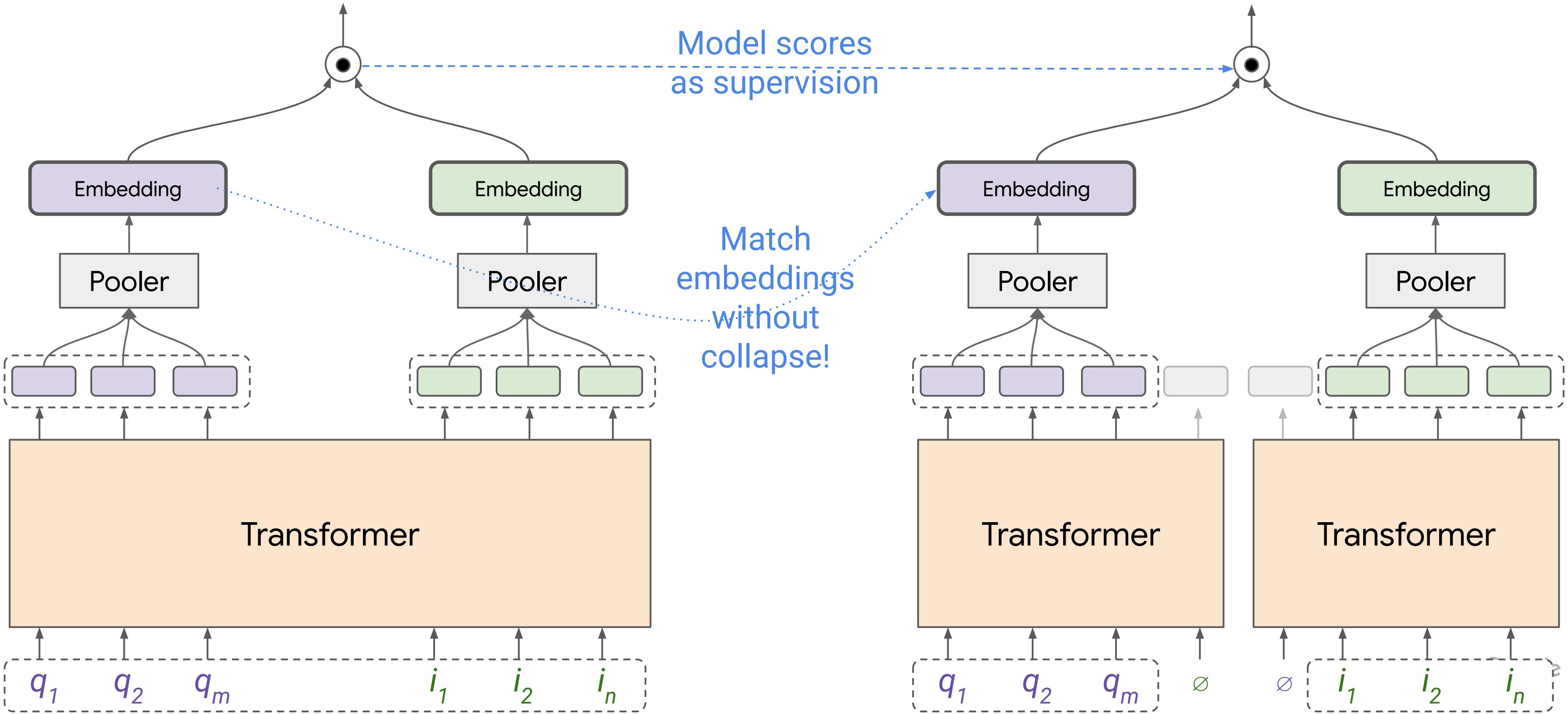
USTAD: unified cross- and dual-encoder

- Re-use same Transformer for both cross- and dual-encoder!



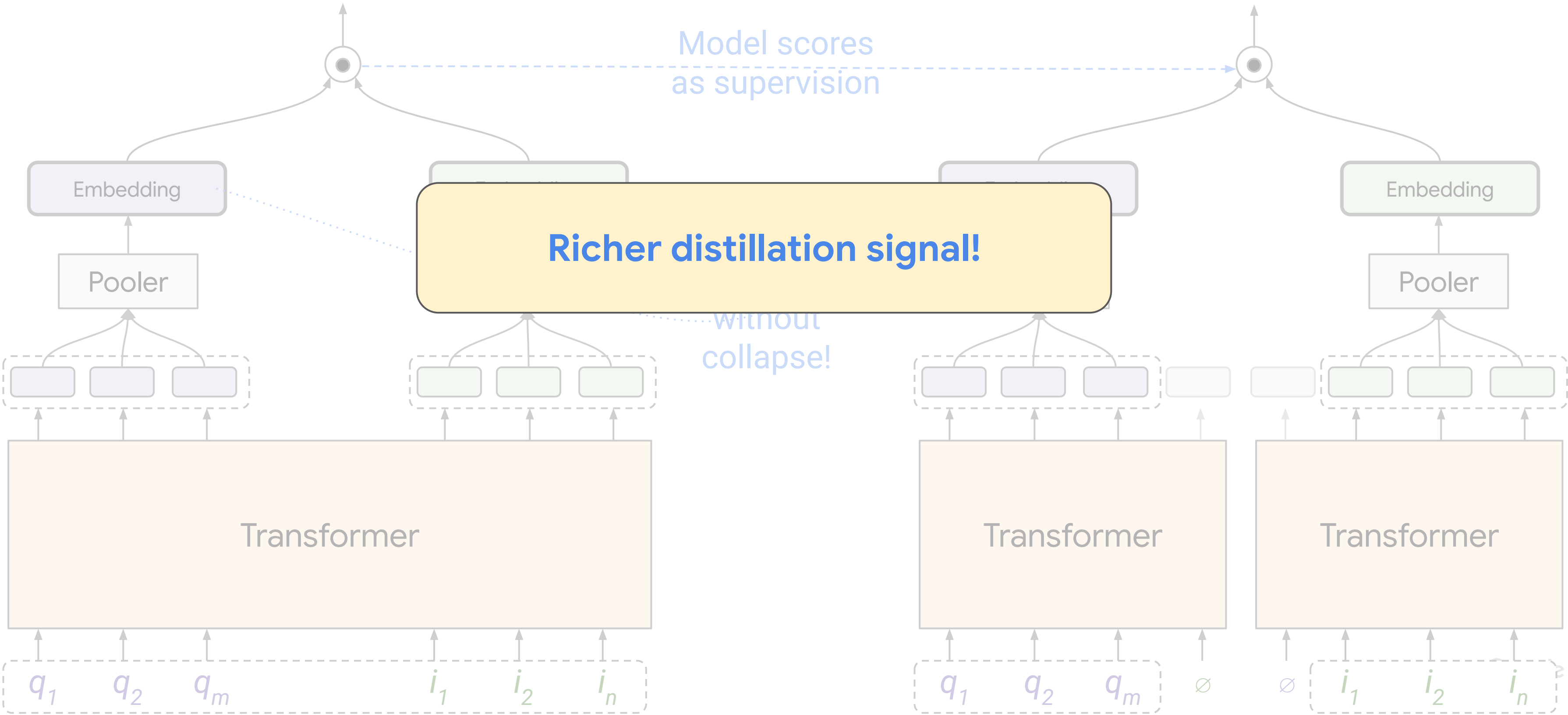
USTAD cross-encoder distillation

- Distill final scores and intermediate embeddings!



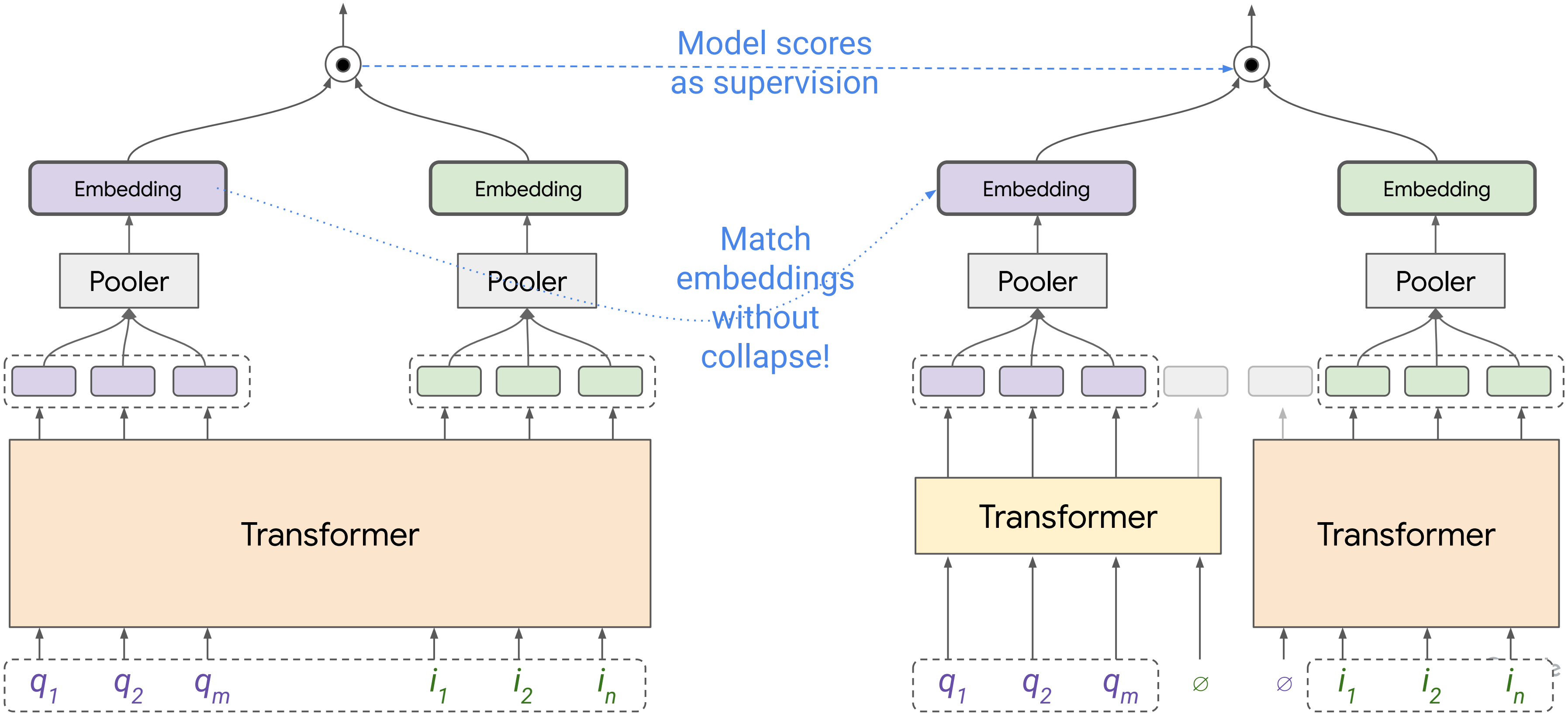
USTAD cross-encoder distillation

- Distill final scores and intermediate embeddings!



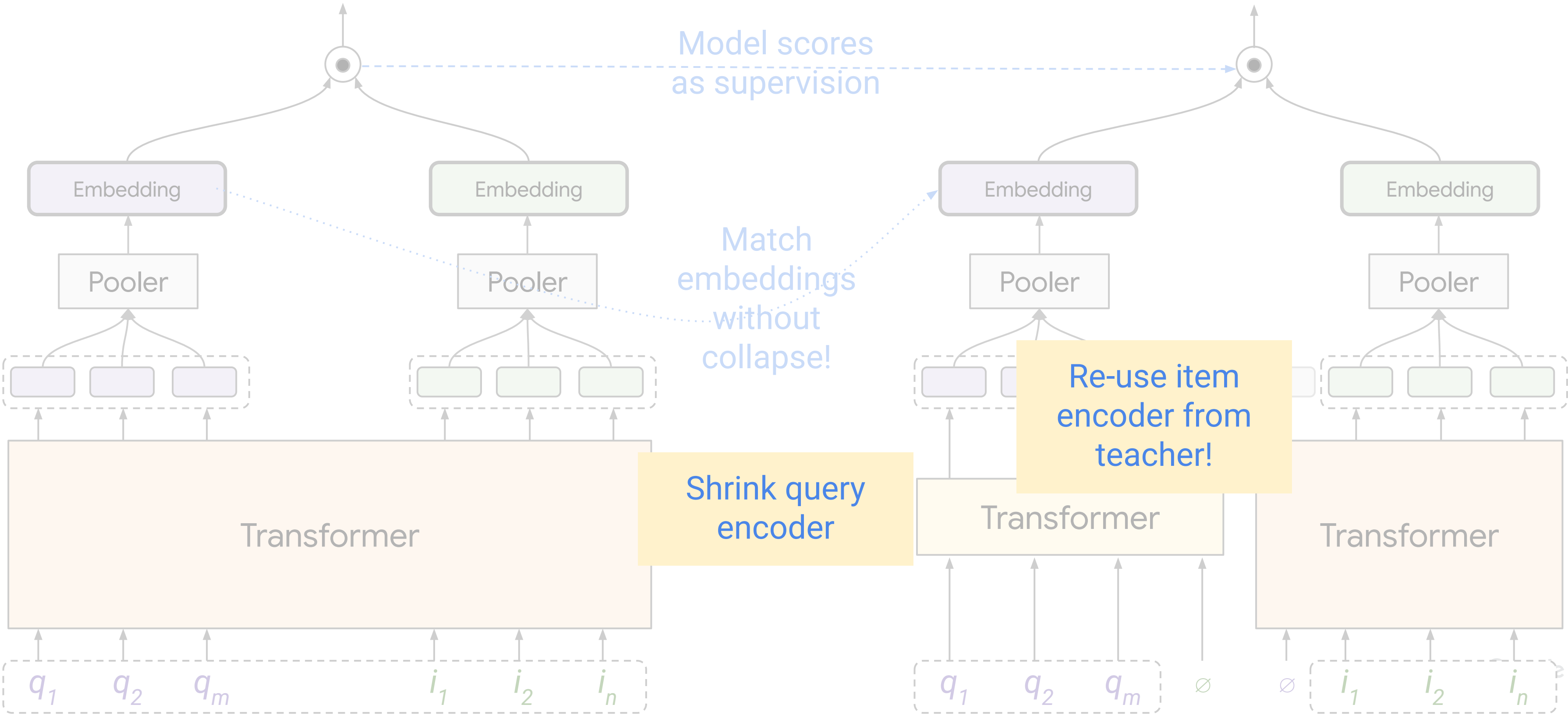
USTAD cross-encoder distillation + item tower re-use

- Distill final scores and intermediate embeddings!



USTAD cross-encoder distillation + item tower re-use

- Distill final scores and intermediate embeddings!



USTAD → smaller dual-encoder

- Embedding matching from USTAD teacher is powerful:

Dataset	Natural Questions (Dev)						MSMARCO (Dev)			
Method	67.5M			11.3M			67.5M		11.3M	
	R@1	R@5	R@10	R@1	R@5	R@10	MRR@10	nDCG@10	MRR@10	nDCG@10
Train student directly	39.5	66.4	74.7	34.1	59.8	68.6	27.0	32.2	23.0	29.7

Table 2. Reranking performance of various student DE models on NQ and MSMARCO dev set, including symmetric DE model (67.5M or 11.3M transformer as both encoders) and asymmetric DE student model (67.5M or 11.3M transformer as query encoder and document embeddings inherited from USTAD teacher). **The USTAD teacher achieves R@1 = 47.4, R@5 = 77.2, R@10 = 83.7, on NQ and MRR@10 = 40.0, nDCG@10 = 45.8 on MSMARCO.**

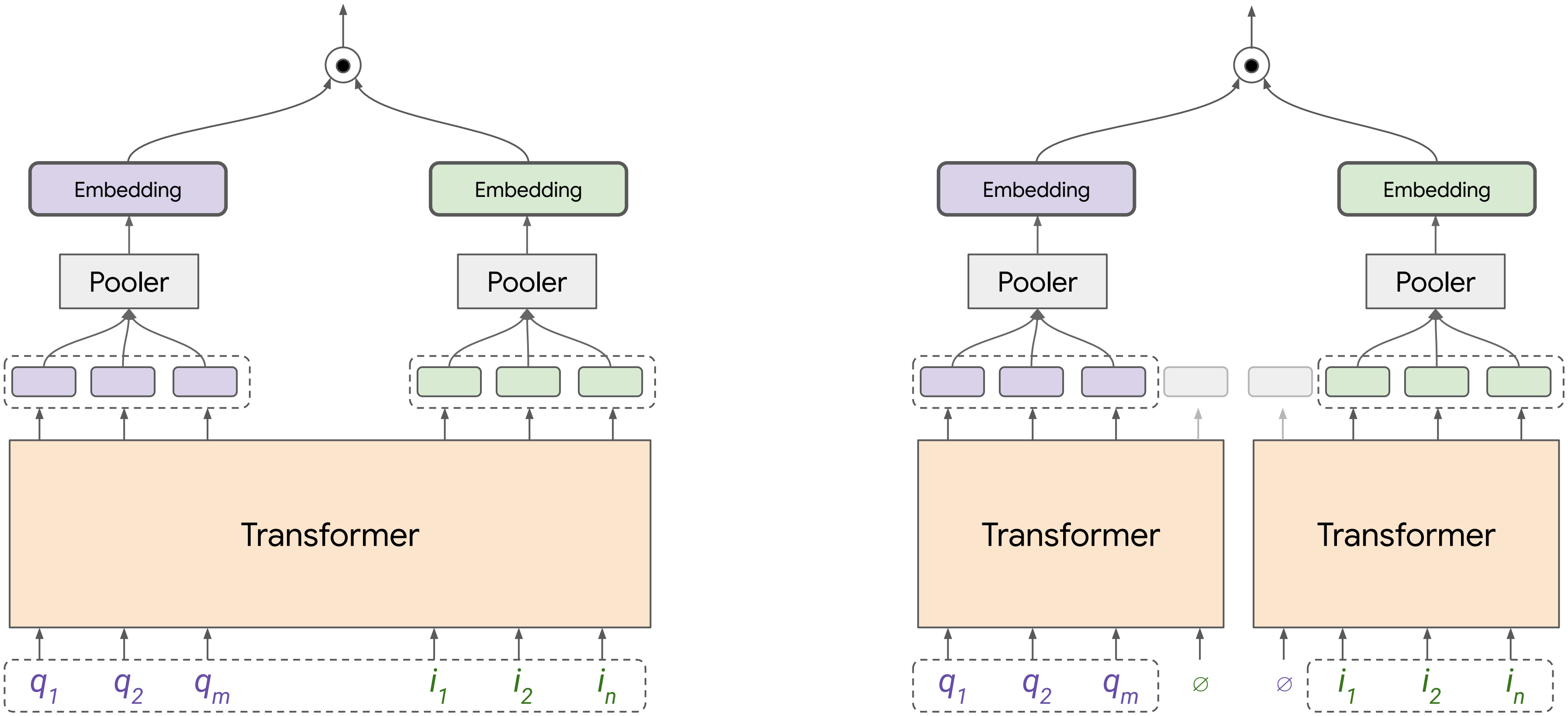
Generic dual-encoder → smaller dual-encoder

- Embedding matching from generic dual-encoder teacher (e.g., SentenceBERT) also shows gains:

Dataset	Natural Questions (Dev)						MSMARCO (Dev)			
Method	67.5M			11.3M			67.5M		11.3M	
	R@5	R@20	R@100	R@5	R@20	R@100	MRR@10	nDCG@10	MRR@10	nDCG@10
Train student directly	36.2	59.7	80.0	24.8	44.7	67.5	22.6	27.2	18.6	22.5

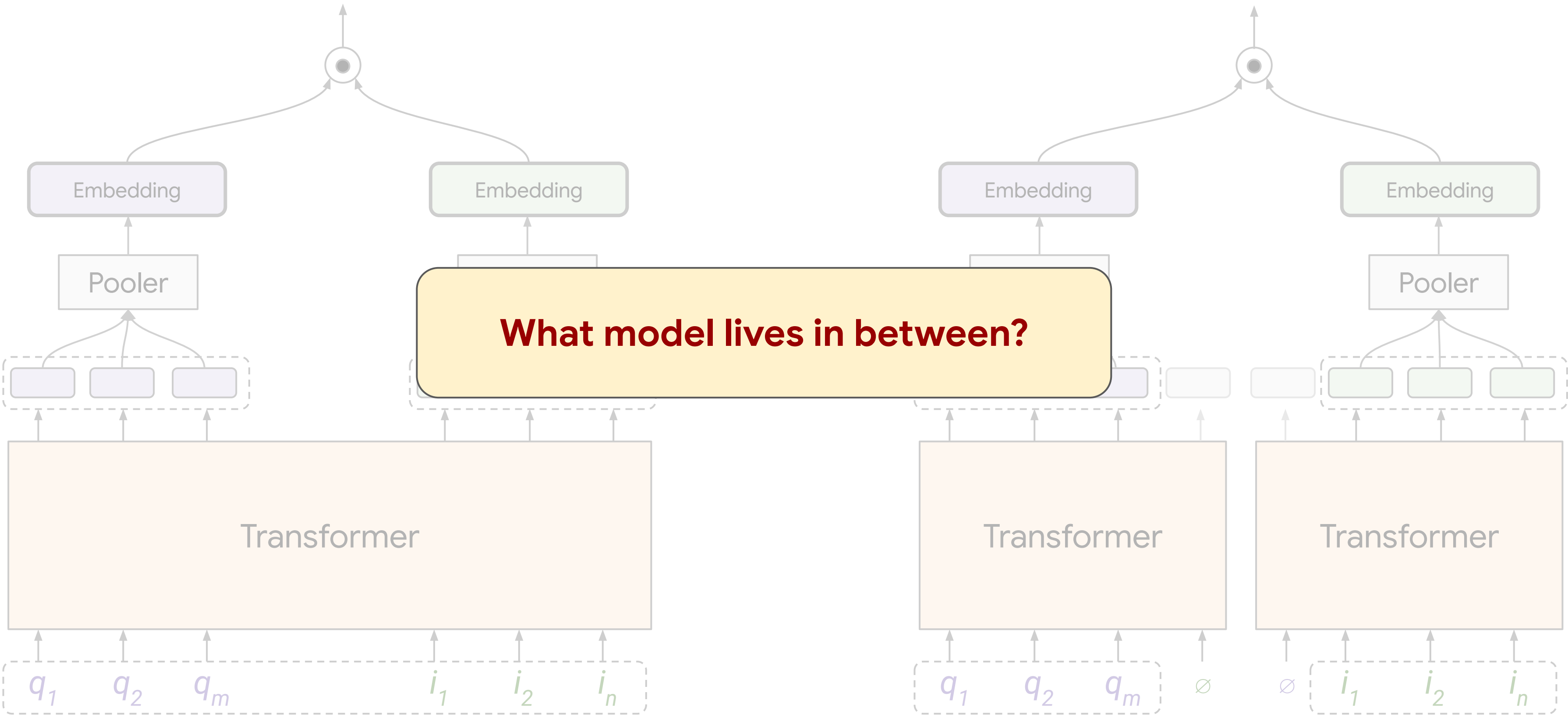
Table 4. Retrieval performance (full recall against all documents in the corpus) of various student DE models on NQ and MSMARCO dev set, including symmetric DE model (67.5M or 11.3M transformer as both encoders) and asymmetric DE student model. **Teacher achieved R@5 = 72.3, R@20 = 86.1, and R@100 = 93.6 on NQ and MRR@10 = 37.2 and nDCG@10 = 44.2 on MSMARCO.**

USTAD: cross- and dual-encoder mode



Google

USTAD: cross- and dual-encoder mode



Agenda

01 A (neural) retrieval primer

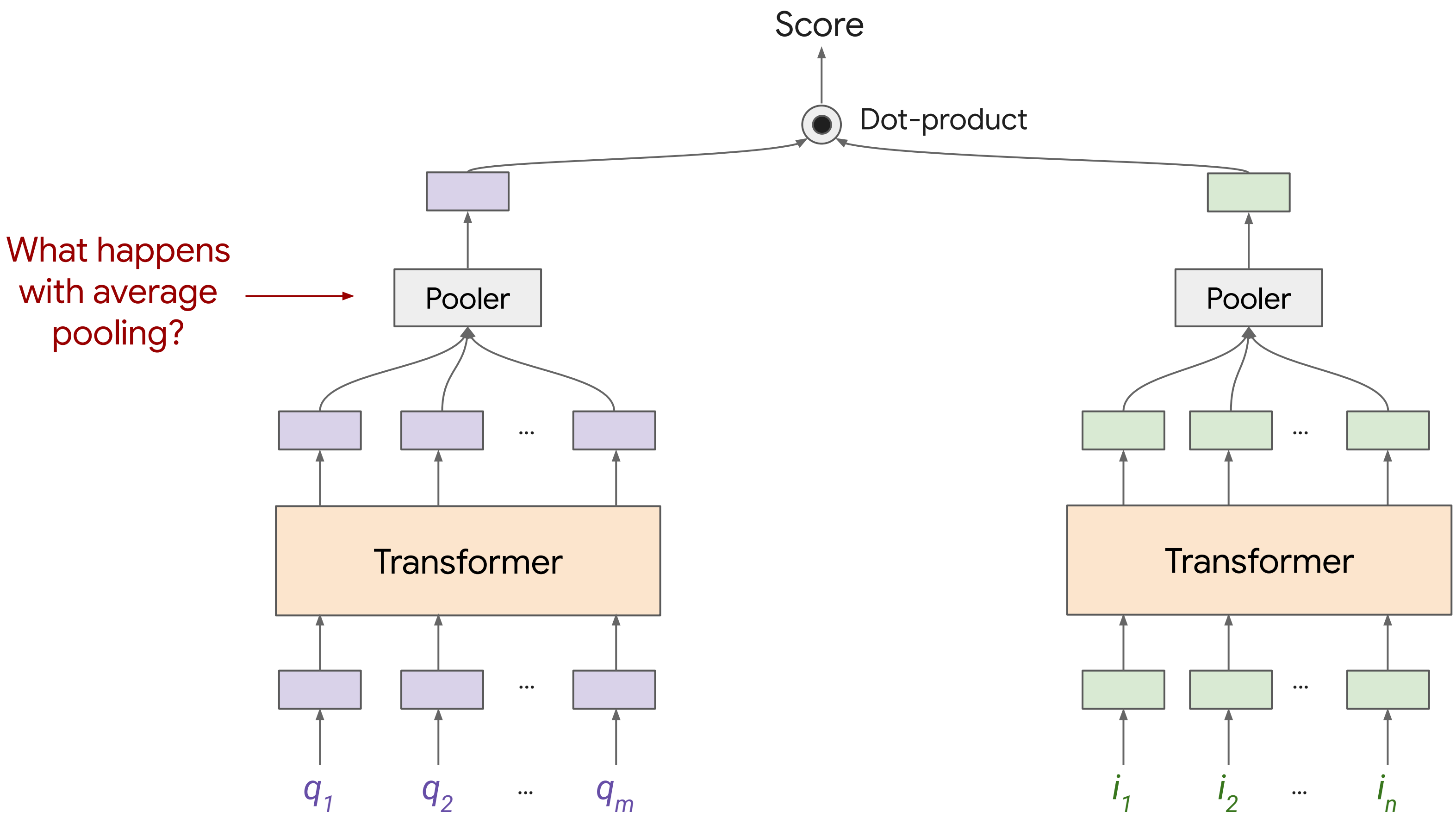
02 Limits of dual encoders

03 Unified cross & dual encoders

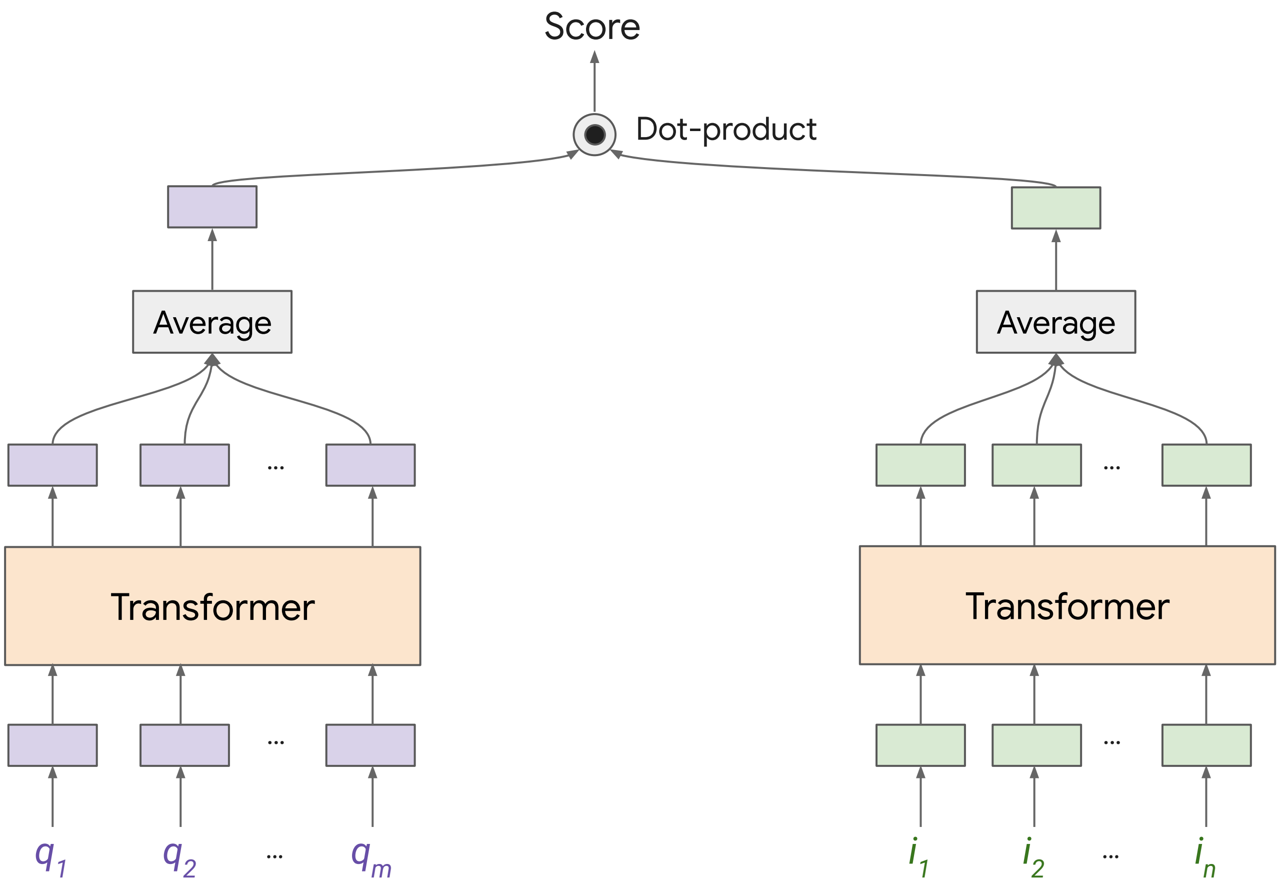
04 Hybrid cross & dual encoders

05 Conclusion & future work

Dual-encoder: recap

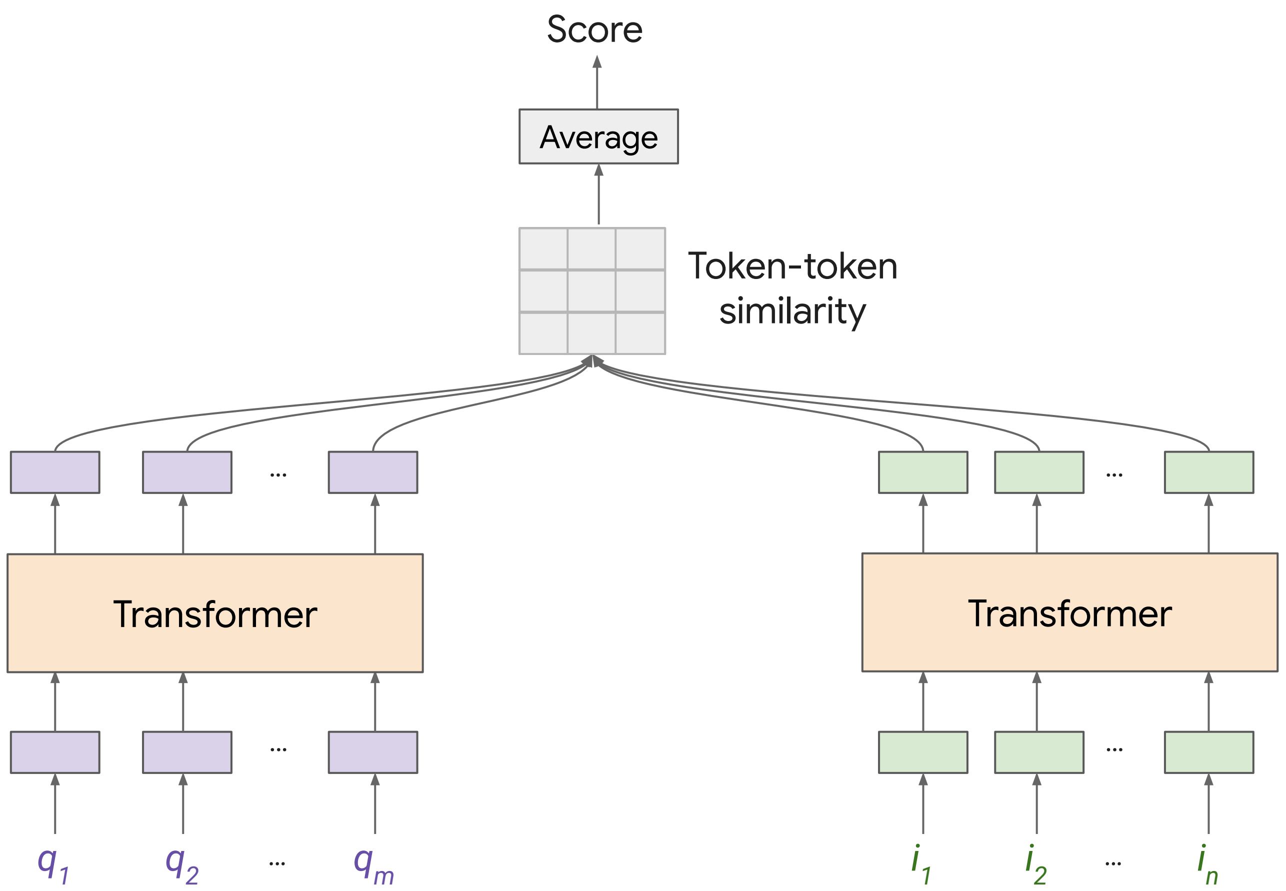


A closer look at average pooling



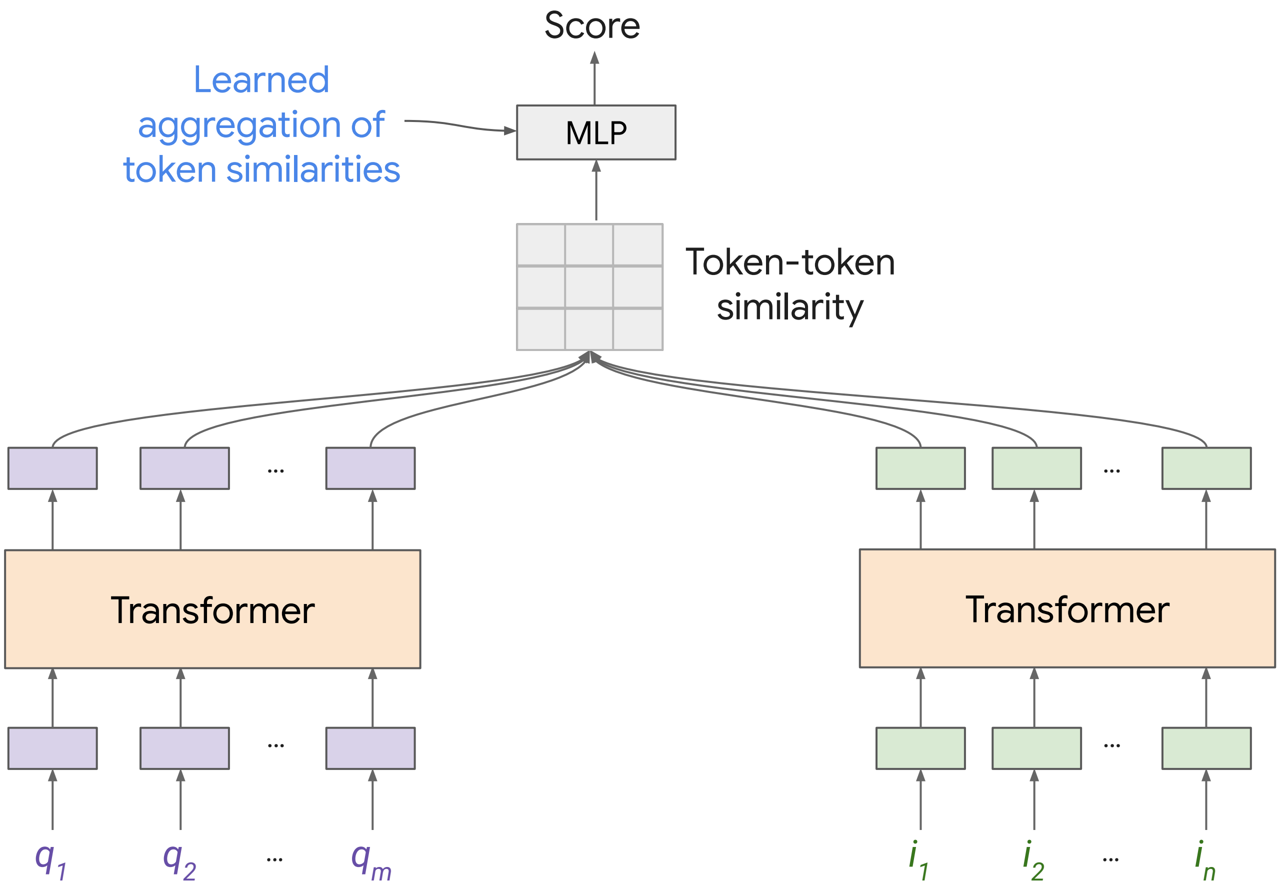
Google

A closer look at average pooling



Google

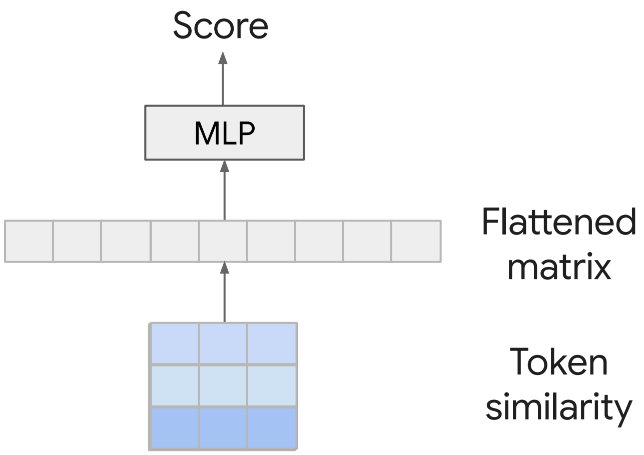
Learnable late-interaction (LITE)



Google

A closer look at the MLP

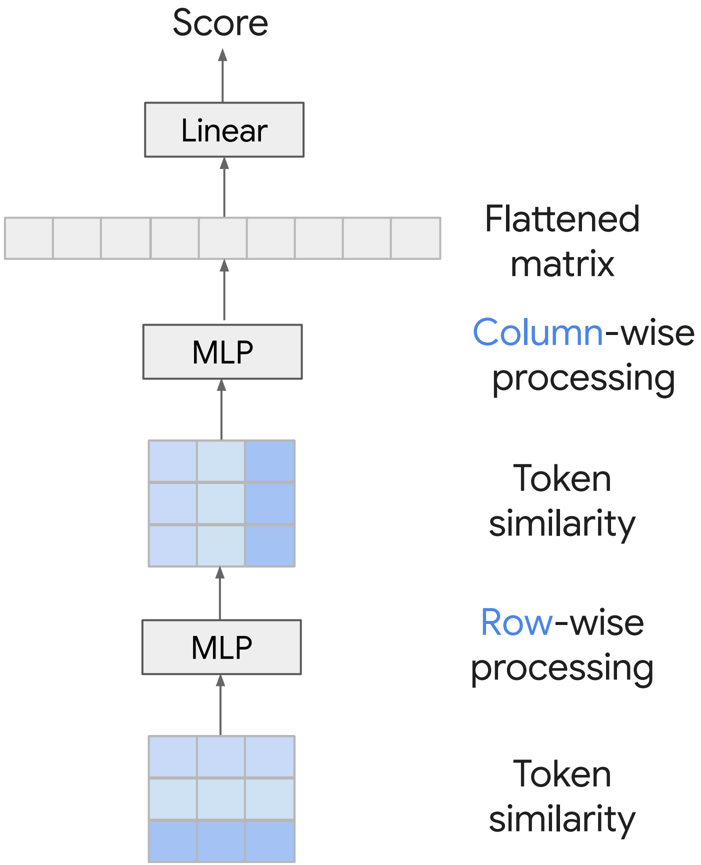
- Flattened LITE: operate on flattened token similarities



Efficient document ranking with learnable late interactions. Ji et al. arXiV 2024.

A closer look at the MLP

- Separable LITE: alternately process rows & columns
- MLP-Mixer style



Efficient document ranking with learnable late interactions. Ji et al. arXiV 2024.

Comparison to ColBERT

- ColBERT is a canonical late-interaction model, of the form:

$$s(q, i) = \Sigma_a \max_b q_a^T i_b$$

- This involves a fixed aggregation of query and item tokens
  - May not be appropriate in all settings
- On the other hand, ColBERT is amenable to retrieval as well

ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. Khattab and Zaharia. SIGIR 2020.

Approximation power of dual-encoders

- Can dual-encoders fit any (reasonable) relevance function?

- **Yes**, with sufficiently high embedding dimension!

> **Proposition.** Under mild technical conditions, any continuous query-item score function $s(q, i)$ can be approximated by some $Z(q)^{\top} W(i)$, where $Z(q)$, $W(i)$ have at most countably infinite dimension.

- But what if the embedding size is restricted?

Approximation limits of dual-encoders

- Dual-encoders cannot approximate arbitrary functions with a restricted dimension!
  - Embedding dimension needs to scale with the sequence length

Proposition. Suppose queries and items are represented as length L sequences in some P embedding space. There exists a continuous function $s(q, i)$ such that, for any encoders $Z(q)$, $W(d)$ into some $Q < P\,L$ dimensional space, $Z(q)^{\mathsf{T}} W(d)$ suffers a constant approximation error against s.

Efficient document ranking with learnable late interactions. Ji et al. arXiV 2024.

Approximation power of LITE

- On the other hand, LITE turns out to be a universal approximator!
- Notably:
  - Without position encodings, result holds (ColBERT fails in this case)
  - With position encodings, result holds over (two!) pooled tokens' similarity

Proposition. Suppose queries and items are represented as length L sequences in some P embedding space. For any continuous function $s(q, i)$, there is a LITE model (i.e., Transformer + MLP) that can approximate s up to arbitrary precision.

Experiments: in-domain re-ranking

- LITE effectively interpolates between cross- & dual-encoders

Scorer	Latency (in ms)	Storage	MS MARCO MRR@10
CE (student)	10990	0×	0.395
DE	42	1×	0.355
ColBERT	62	200×	0.383
Separable LITE	111	200×	0.393
Small sep LITE	56	50×	0.391

Highest quality, but highest cost

4x less document tokens

	MS MARCO		DL 2019		DL 2020		NQ	
Scorer	MRR	nDCG	MRR	nDCG	MRR	nDCG	MRR	nDCG
DE	0.355	0.413	0.861	0.744	0.842	0.723	0.699	0.611
ColBERT	0.383	0.442	0.878	0.753	0.860	0.731	0.756	0.689
Sep LITE	0.393	0.452	0.898	0.765	0.873	0.756	0.769	0.693

Experiments: in-domain re-ranking

- LITE effectively interpolates between cross- & dual-encoders

Scorer	Latency (in ms)	Storage	MS MARCO MRR@10
CE (student)	10990	0×	0.395
DE	42	1×	0.355
ColBERT	62	200×	0.383
Separable LITE	111	200×	0.393
Small sep LITE	56	50×	0.391

Close to CE quality with much lower cost

4x less document tokens

	MS MARCO		**DL 2019**		**DL 2020**		**NQ**	
Scorer	**MRR**	**nDCG**	**MRR**	**nDCG**	**MRR**	**nDCG**	**MRR**	**nDCG**
DE	0.355	0.413	0.861	0.744	0.842	0.723	0.699	0.611
ColBERT	0.383	0.442	0.878	0.753	0.860	0.731	0.756	0.689
Sep LITE	0.393	0.452	0.898	0.765	0.873	0.756	0.769	0.693

Experiments: in-domain re-ranking

- LITE effectively interpolates between cross- & dual-encoders

Scorer	Latency (in ms)	Storage	MS MARCO MRR@10
CE (student)	10990	0×	0.395
DE	42	1×	0.355
ColBERT	62	200×	0.383
Separable LITE	111	200×	0.393
Small sep LITE	56	50×	0.391

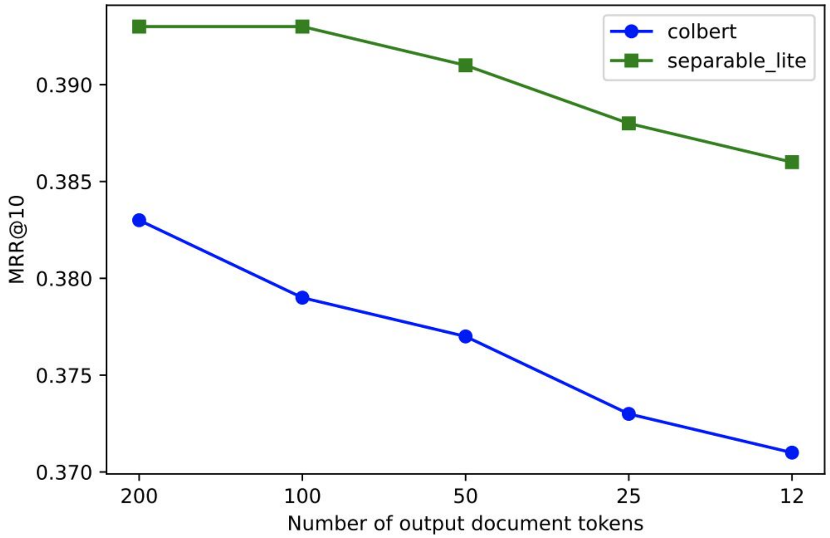
Significantly better than DE quality

4x less document tokens

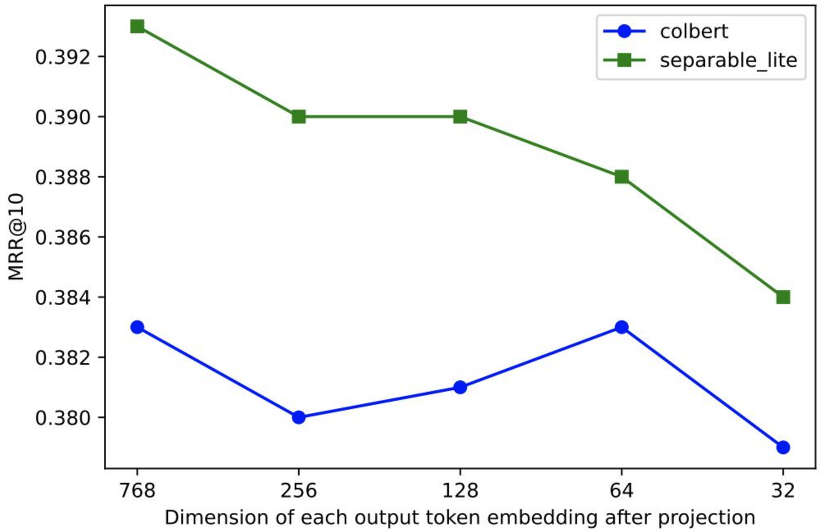
	MS MARCO		DL 2019		DL 2020		NQ	
Scorer	**MRR**	**nDCG**	**MRR**	**nDCG**	**MRR**	**nDCG**	**MRR**	**nDCG**
DE	0.355	0.413	0.861	0.744	0.842	0.723	0.699	0.611
ColBERT	0.383	0.442	0.878	0.753	0.860	0.731	0.756	0.689
Sep LITE	0.393	0.452	0.898	0.765	0.873	0.756	0.769	0.693

Experimental results: cost reduction

- Lightweight scoring methods require more storage than dual-encoders
- LITE performs well with pooling and/or reduced embedding size!



Reduction via local averaging or projection



Reduction via projection

Experiments: out-of-domain re-ranking

- LITE shows consistently good generalisation on BEIR tasks

Dataset	ColBERT	Sep LITE	CE
T-COVID	0.761	0.763	0.771
NFCorpus	0.356	0.358	0.361
NQ	0.525	0.540	0.552
HotpotQA	0.685	0.681	0.728
FiQA-2018	0.330	0.336	0.346
ArguAna	0.433	0.424	0.519
Touché-2020	0.274	0.305	0.300
CQAD	0.363	0.374	0.378
Quora	0.767	0.839	0.832
DBPedia	0.410	0.434	0.438
SCIDOCS	0.155	0.164	0.167
FEVER	0.782	0.788	0.804
C-FEVER	0.190	0.213	0.232
SciFact	0.667	0.633	0.695

Agenda

01 A (neural) retrieval primer

02 Limits of dual encoders

03 Unified cross & dual encoders

04 Hybrid cross & dual encoders

05 Conclusion & future work

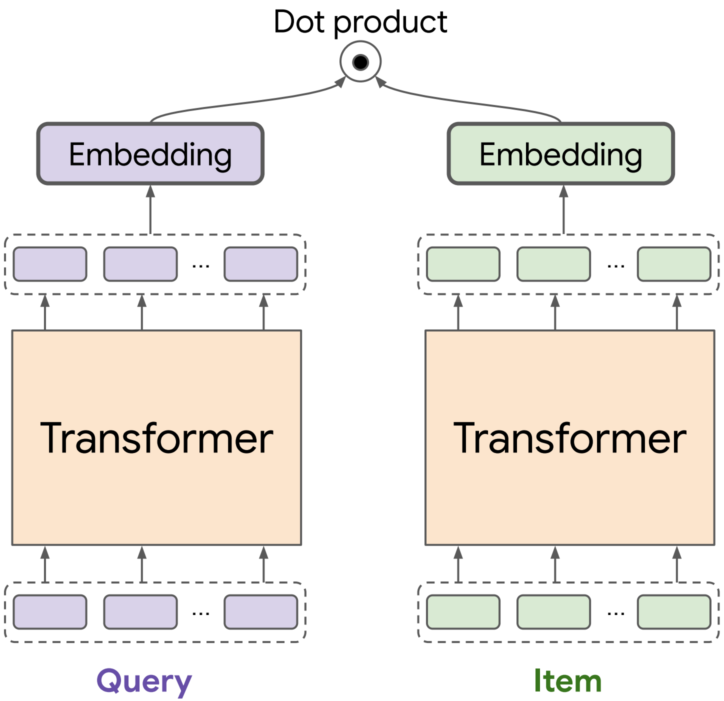
Google Research

Cross- versus dual-encoders

Dual-encoders tend to underperform for re-ranking

Why does this happen?
- Poorer margins
- Expressivity with small dimension

What can we do about it?
- Score-based distillation
- Architecture modification
- 🔬 Embedding-based distillation
- Lightweight scoring



Future work

Further optimising the encoder (cost, quality) tradeoff

Can we get the best of both worlds?

Unified retrieval and re-ranking

Do we really need two phases?

Generative retrieval and re-ranking

Do we even need encoder models?!

Thank You

Aditya Krishna Menon
Research Scientist