ML & Neural Approaches for Information Retrieval (Part 1)

Andrew Yates

Assistant Professor, UvA

@andrewyates

July 20, 2022

The 13th European Summer School in Information Retrieval

Universiteit van Amsterdam

About me

UNIVERSITEIT VAN AMSTERDAM

Assistant Professor at University of Amsterdam (2021-)

- Information retrieval, neural networks, natural language processing, (consumer) medical domain, personal knowledge graphs, …

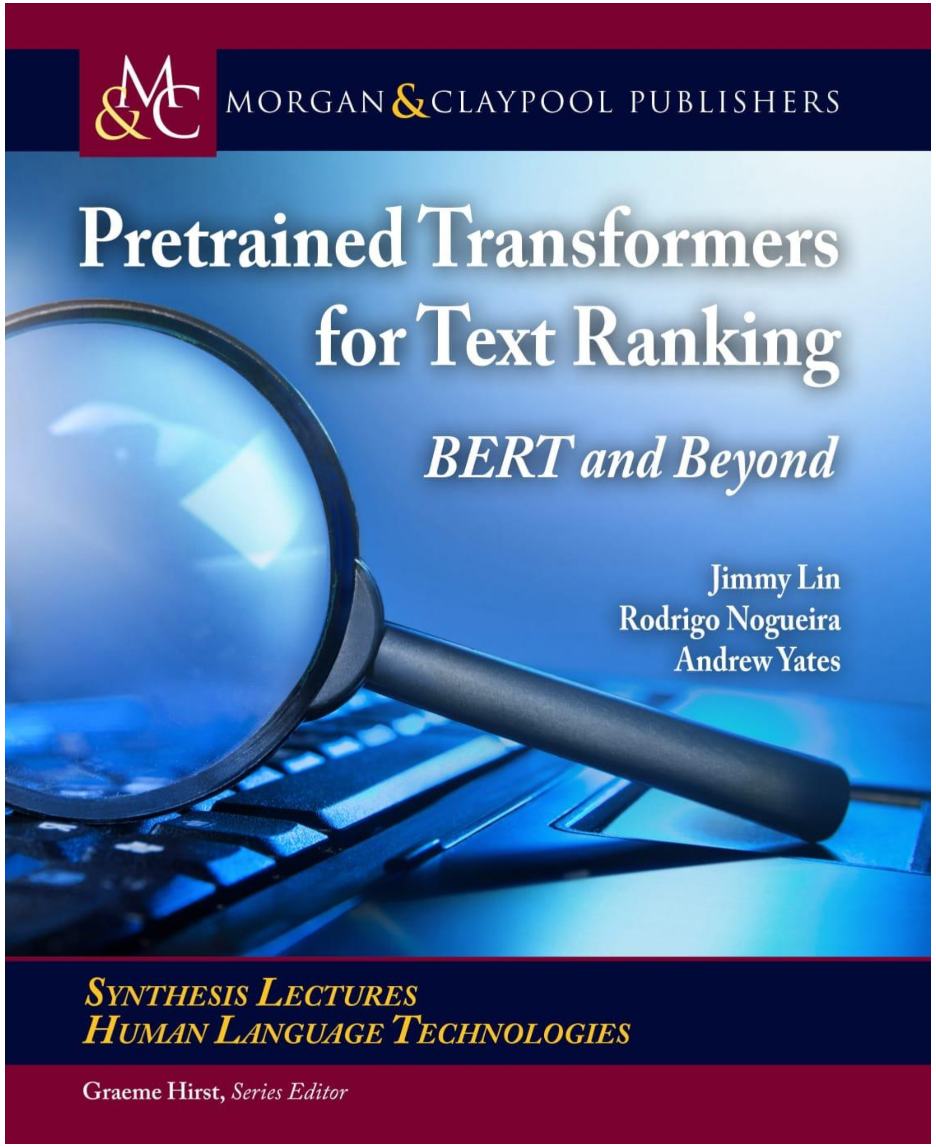
Previously

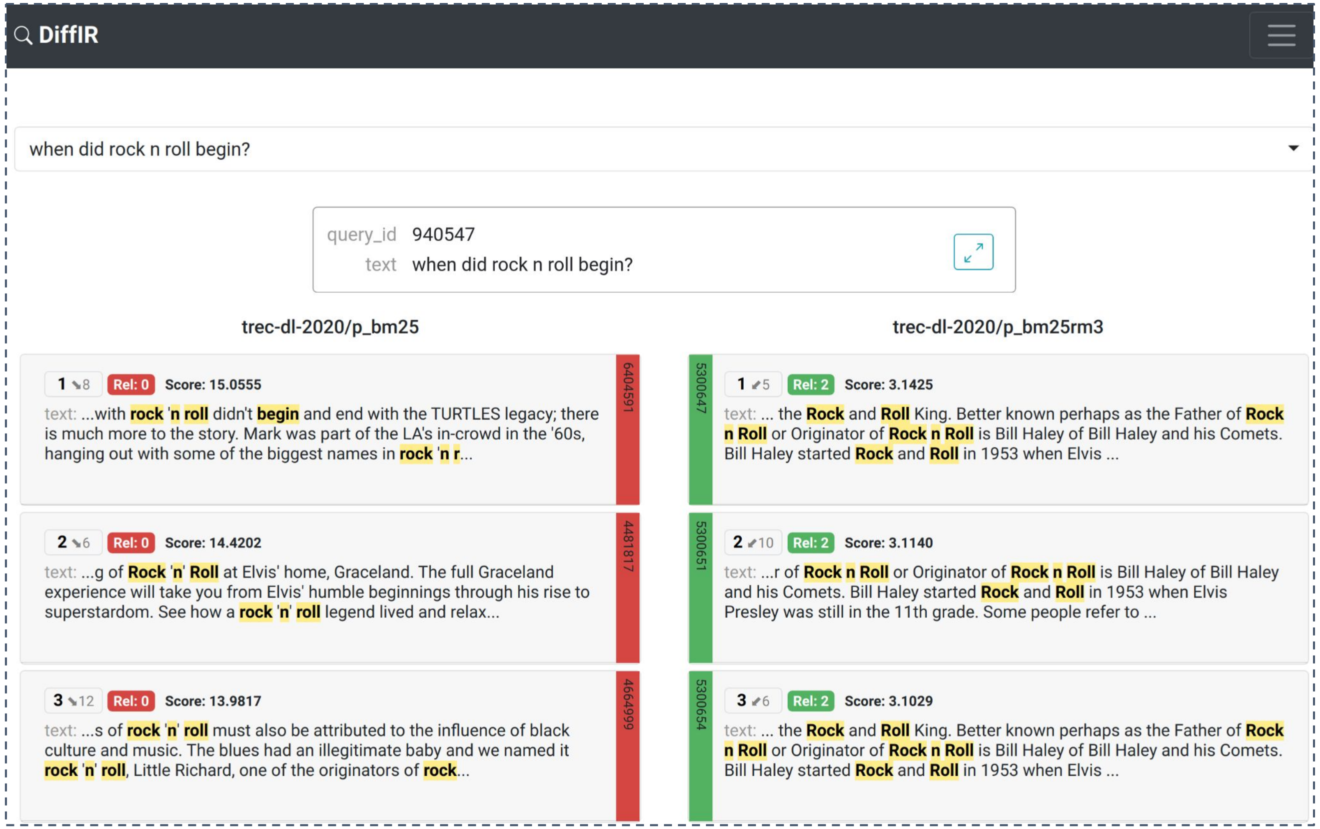
- Postdoc then Senior Researcher (2018-2021) at Max Planck Institute for Informatics
- PhD (2016) in Computer Science from Georgetown University, Washington, DC
- BSc in Computer Science from Illinois Institute of Technology, Chicago

max planck institut informatik

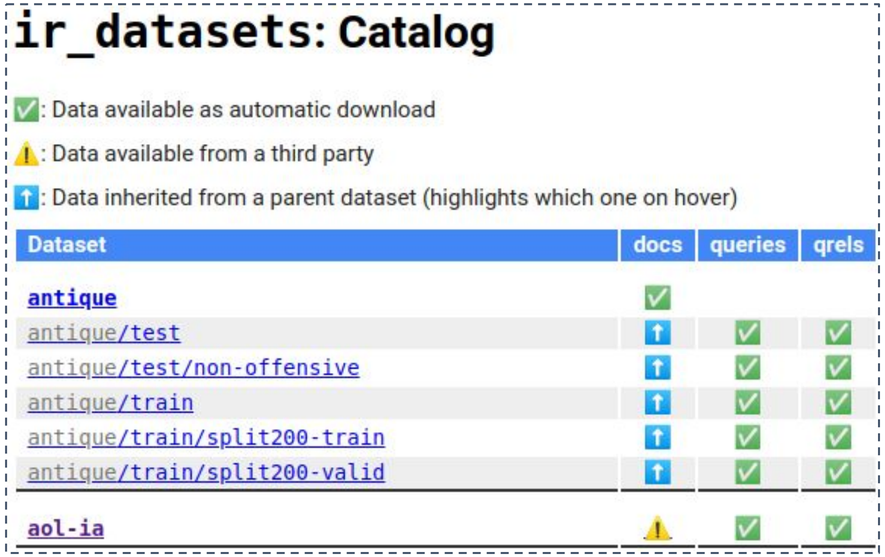
ir LAB GEORGETOWN UNIVERSITY

ILLINOIS INSTITUTE OF TECHNOLOGY 1890





Capreolus: Toolkit for Neural Ad Hoc Retrieval


ir_datasets: Catalog

✅: Data available as automatic download

⚠️: Data available from a third party

⬆️: Data inherited from a parent dataset (highlights which one on hover)

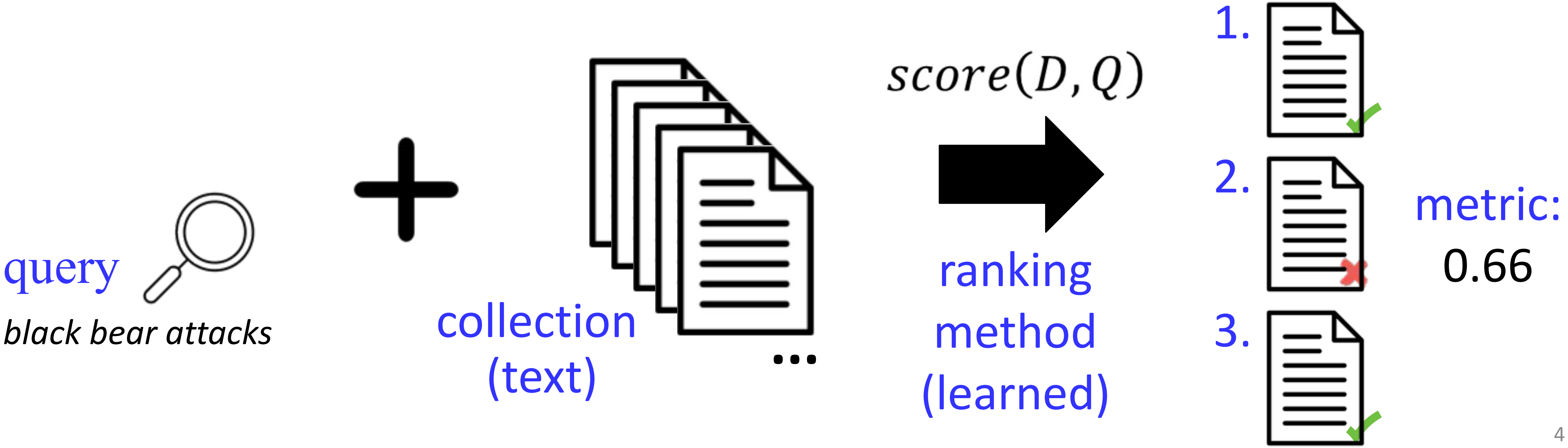
Dataset	docs	queries	qrels
antique	✅		
antique/test	⬆️	✅	✅
antique/test/non-offensive	⬆️	✅	✅
antique/train	⬆️	✅	✅
antique/train/split200-train	⬆️	✅	✅
antique/train/split200-valid	⬆️	✅	✅
aol-ia	⚠️	✅	✅

Collaborators: Siddhant Arora, Arman Cohan, Jeffrey Dalton, Doug Downey, Sergey Feldman, Nazli Goharian, Kevin Martin Jose, Jimmy Lin, Sean MacAvaney, Thong Nguyen, Rodrigo Nogueira, Wei Yang, Xinyu Zhang, …

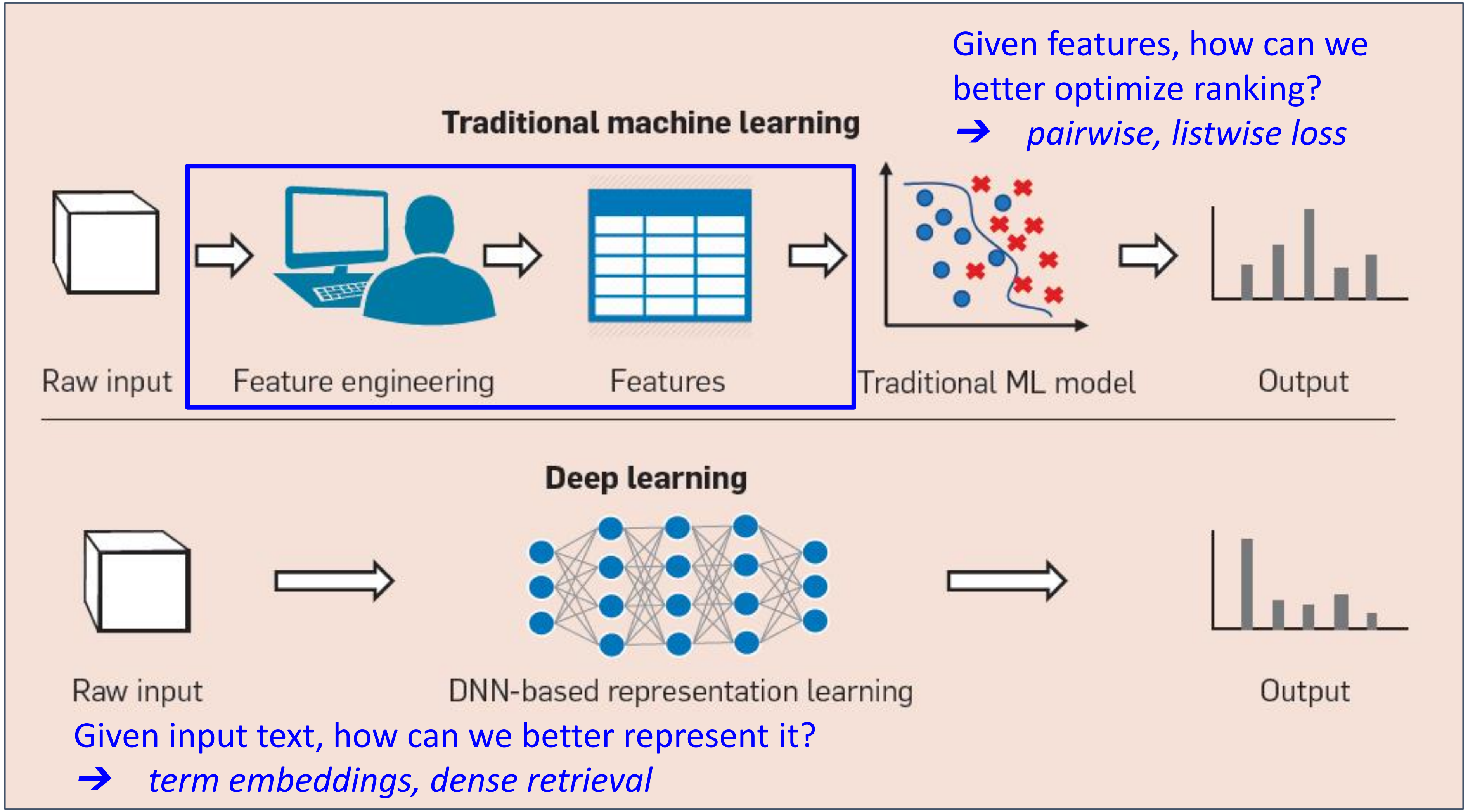
Introduction

Methods that use relevance judgments to **learn** how to rank results

- ➔ **ML** approaches **learn to rank** based on hand-crafted features (e.g., BM25 score between Q and D, spam score for D, etc)
- ➔ **Neural** approaches learn improved **text representations** for ranking



Introduction



Given features, how can we better optimize ranking?
➔ *pairwise, listwise loss*

Given input text, how can we better represent it?
➔ *term embeddings, dense retrieval*

Source: Techniques for Interpretable Machine Learning. Du, Liu, Hu. Communications of the ACM. 2020.

Advantage #1: less handcrafting

$$score(D,Q) = \sum_{i=1}^{|Q|} IDF(q_i) * TF(q_i, D)$$

Inverse **D**ocument **F**requency of query term (collection stat)

Term **F**requency of *q* in document *D* (document stat)

N = # *docs* and n_i = # *docs with term* q_i

IDF defined as:

- log N/n_i
- log $(1 + N/n_i)$
- log $(N - n_i)/n_i$
- …

$$score(D,Q) = \sum_{i=1} IDF(q_i) * TF(q_i, D)$$

Inverse **D**ocument **F**requency of query term

Term **F**requency of *q* in document *D*

TF defined as:

- $TF(q_i, D)$
- $1 + \log TF(q_i, D)$
- $0.5 + 0.5\,[TF(q_i, D) / \max TF(D)]$
- ...

$$score(D,Q) = \sum_{i=1}^{|Q|} IDF(q_i) * TF(q_i, D)$$

Inverse **D**ocument **F**requency of query term

Term **F**requency of *q* in document *D*

Advantage #2: soft matching of terms

$$score(D,Q) = \boxed{\sum_{i=1}^{|Q|}} IDF(q_i) * TF(q_i, D)$$

Exact matching of
Q and D terms

Non-neural approaches: translation models, topic models like pLSI, ...

Advantage #2: soft matching of terms

```
document ranking
 text ranking
  document retrieval
   Web search
```

Match?

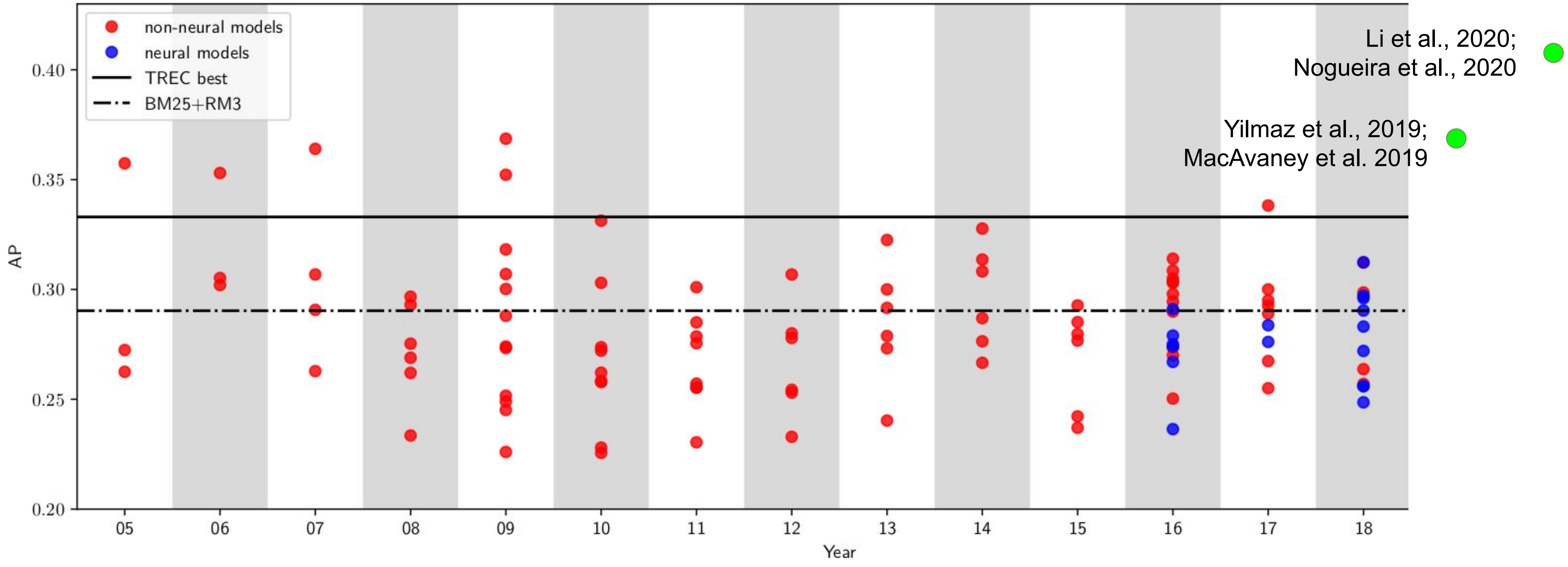
```
query, search,
ranking, retrieval, ...
```

Enrich query or document representations to move beyond exact matching

- Unsupervised: pseudo-relevance feedback
- Neural: embeddings, document expansion, …

Search quality improvements

Robust04 Dataset (news articles)



Source: Yang et al., SIGIR 2019

Search quality improvements

MS MARCO Passage Dataset (Web pages)

Rank	Model	Submission Date	MRR@10 On Eval	MRR@10 On Dev
1	**BERT + Small Training** Rodrigo Nogueira and Kyunghyun Cho - New York University [Nogueira, et al. '19] and [Code]	January 7th, 2019	35.87	
2	**IRNet (Deep CNN/IR Hybrid Network)** Dave DeBarr, Navendu Jain, Robert Sim, Justin Wang, Nirupama Chandrasekaran – Microsoft	January 2nd, 2019	28.06	
3	**Neural Kernel Match IR (Conv-KNRM)**(1)Yifan Qiao, (2)Chenyan Xiong, (3)Zhenghao Liu, (4)Zhiyuan Liu-Tsinghua University(1, 3, 4); Microsoft Research AI(2) [Dai et al. '18]	Novmeber 28th, 2018	27.12	29.02
4	**Neural Kernel Match IR (KNRM)** ((1)Yifan Qiao, (2)Chenyan Xiong, (3)Zhenghao Liu, (4)Zhiyuan Liu-Tsinghua University(1, 3, 4); Microsoft Research AI(2) [Xiong et al. '17]	December 10th, 2018	19.82	21.84
5	**Feature-based LeToR: simple-feature based RankSVM**(1)Yifan Qiao, (2)Chenyan Xiong, (3)Zhenghao Liu, (4)Zhiyuan Liu-Tsinghua University(1, 3, 4); Microsoft Research AI(2)	December 10th, 2018	19.05	19.47
6	**BM25**	Novmeber 1st, 2018	16.49	16.70

~8 points!

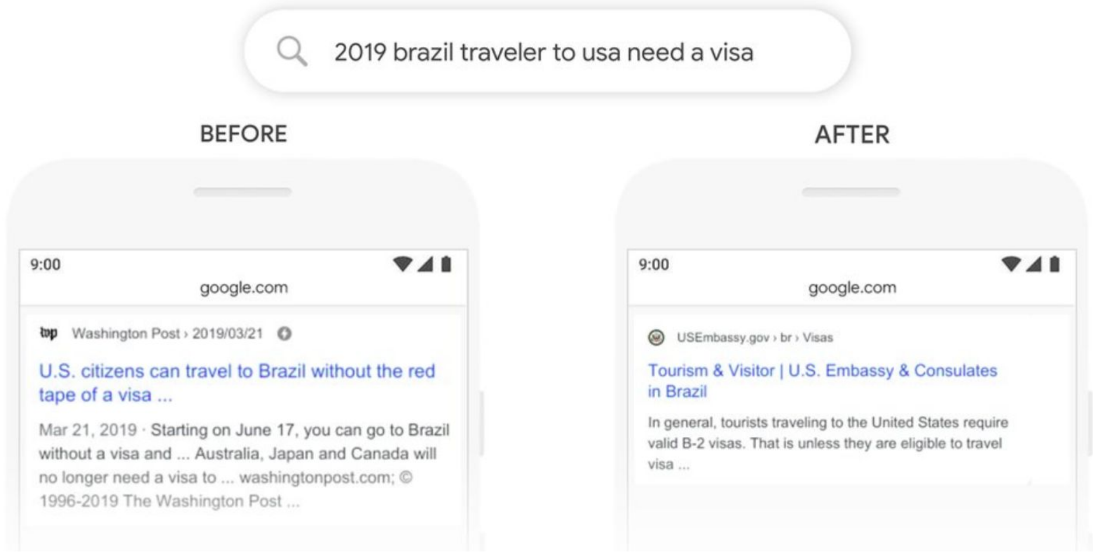
19 point improvement: BM25 vs. best neural

8 point improvement: neural pre-LLM vs. LLM

Source: MS MARCO Leaderboard

Search quality improvements: industry

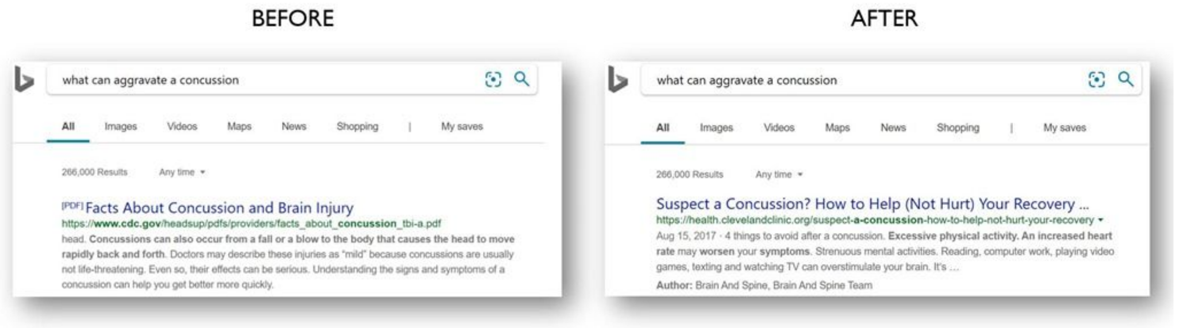
Google Search



*We're making a significant improvement to how we understand queries, representing the **biggest leap forward in the past five years**, and **one of the biggest leaps forward in the history of Search**.*

source

MS Bing



Starting from April of this year (2019), we used large transformer models to deliver the largest quality improvements to our Bing customers in the past year.

source

Outline

- Introduction
- **Ranking with soft matching**
- Transformers & contextualized embeddings
- Transformer approaches for ranking
  - Re-ranking with cross-encoders
  - Learned sparse retrieval
  - Document expansion
- Conclusion

From exact matching to soft matching

Key point: neural methods replace *exact* matching with *soft* matching

With traditional methods, soft matching is possible (but it's less effective)

- *Stemming* handles matches like singular vs. plural and different tenses
  swam, *swimming*, *swim, swims*   replaced with   *swim*

- Query expansion approaches handle context by adding related terms to query
  Query contains *bank,* referring to turning a plane
  Add related terms to query: *flight*, *airplane*, *ailerons*, *spoilers*, ...

Soft matching via embeddings

	living being	feline	human	gender	royalty	verb	plural
cat →	0.6	0.9	0.1	0.4	−0.7	−0.3	−0.2
kitten →	0.5	0.8	−0.1	0.2	−0.6	−0.5	−0.1
dog →	0.7	−0.1	0.4	0.3	−0.4	−0.1	−0.3
houses →	−0.8	−0.4	−0.5	0.1	−0.9	0.3	0.8

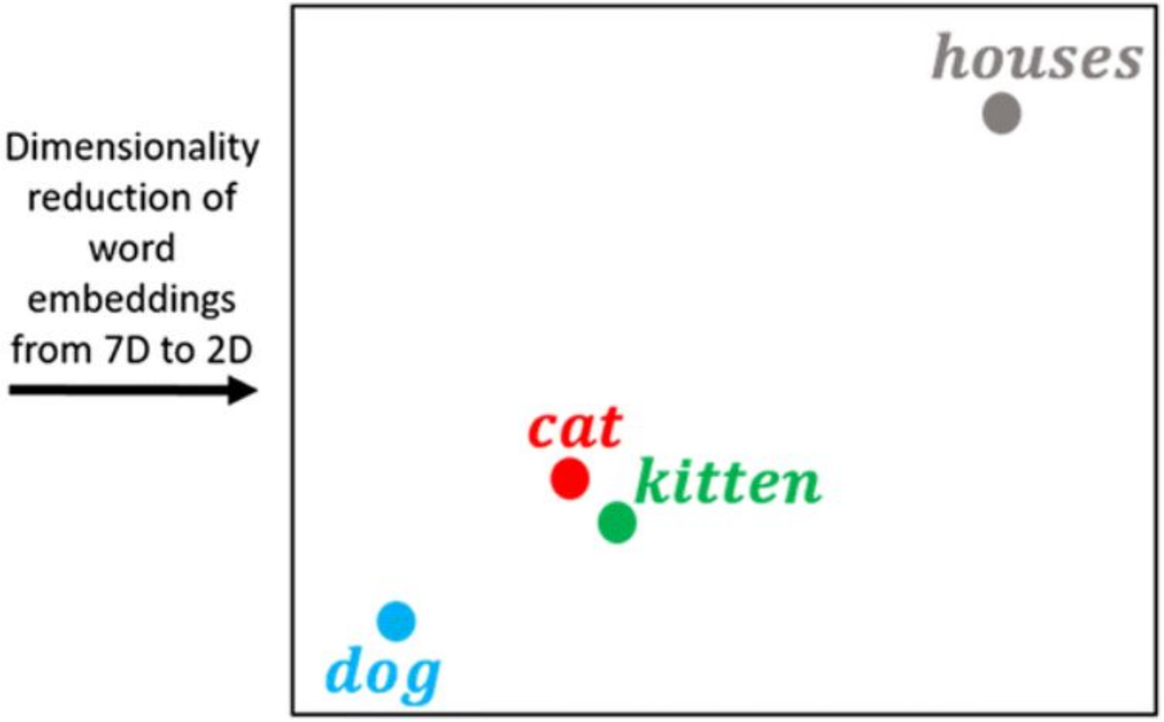
Neural approaches represent words as learned *vectors*

Source: https://medium.com/@hari4om/word-embedding-d816f643140

Soft matching via embeddings

Vector “*embedding*” allows comparisons of different terms

	living being	feline	human	gender	royalty	verb	plural
cat →	0.6	0.9	0.1	0.4	−0.7	−0.3	−0.2
kitten →	0.5	0.8	−0.1	0.2	−0.6	−0.5	−0.1
dog →	0.7	−0.1	0.4	0.3	−0.4	−0.1	−0.3
houses →	−0.8	−0.4	−0.5	0.1	−0.9	0.3	0.8



Source: https://medium.com/@hari4om/word-embedding-d816f643140

Ranking with embeddings

Score can be produced by placing document terms in *similarity buckets*, then computing relevance based on the size of each bucket

Query Terms	sim = 1.0	0.6 < sim < 1.0	sim <= 0.6
curbing	4 terms	3 terms	20 terms
population	1 term	8 terms	40 terms
growth	2 terms	2 terms	15 terms

Similarity Buckets

Weights: *a*, *b*, *c*

Score(*curbing,* D) = 4*a + 3*b + 20*c

Score(Q, D) = Score(*curbing,* D) + Score(*population,* D) + Score(*growth,* D)

Guo, Fan, Ai, Croft. A Deep Relevance Matching Model for Ad-hoc Retrieval. CIKM 2016.

Deep Relevance Matching Model (DRMM)

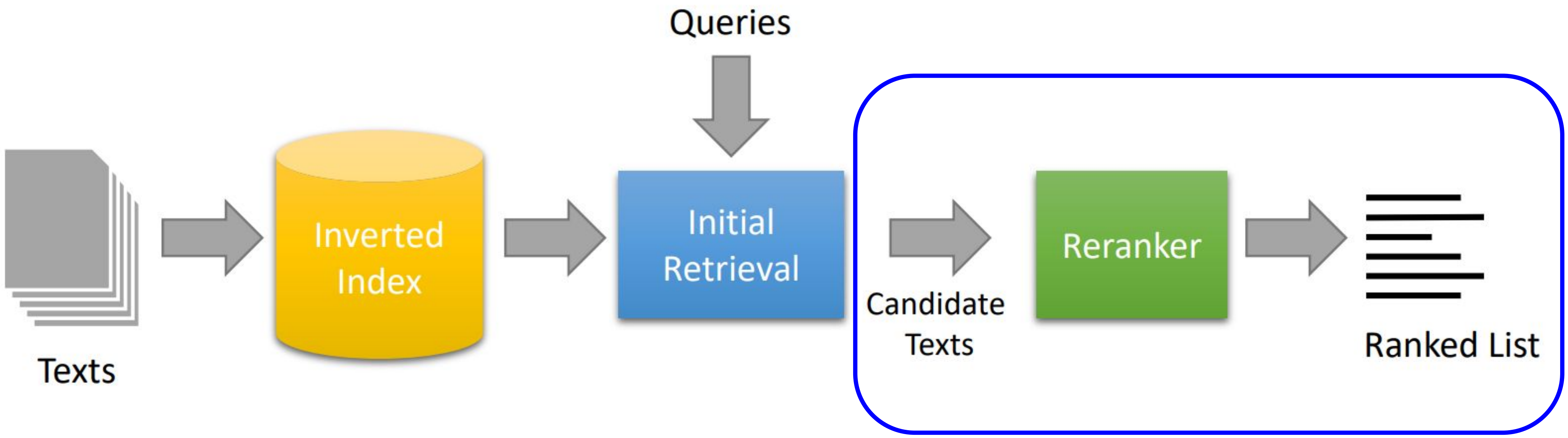
Input: query Q and a document D to score

Scoring procedure:
Load document & embeddings into memory, then compute similarity with query

1. Load document D from the forward index, representing it as a list of term IDs
2. Use term IDs to index into embedding matrix, representing D as a list of embeddings
3. Compute histogram *h(t)* for each query term,
   capturing the cosine similarities between the embeddings of *t* and each doc term
4. Compute term score *z(h(t))* using a feedforward network
5. Compute weight *g(t)* for each query term (IDF or using embedding)
6. Compute relevance score as summation of *z(h(t)) * g(t)* over all query terms

Guo, Fan, Ai, Croft. A Deep Relevance Matching Model for Ad-hoc Retrieval. CIKM 2016.

Reranking with DRMM



Training DRMM

Loss:

$$L = -\sum_{j \in J_{\text{pos}}} \log(s_j) - \sum_{j \in J_{\text{neg}}} \log(1 - s_j)$$

Humans

BM25

Outline

- Introduction
- Ranking with soft matching
- **Transformers & contextualized embeddings**
- Transformer approaches for ranking
    - Re-ranking with cross-encoders
    - Learned sparse retrieval
    - Document expansion
- Conclusion

Contextualized embeddings

Idea: a word's representation should vary with context

➔ Generate embeddings based on given text (e.g., query, document, sentence)

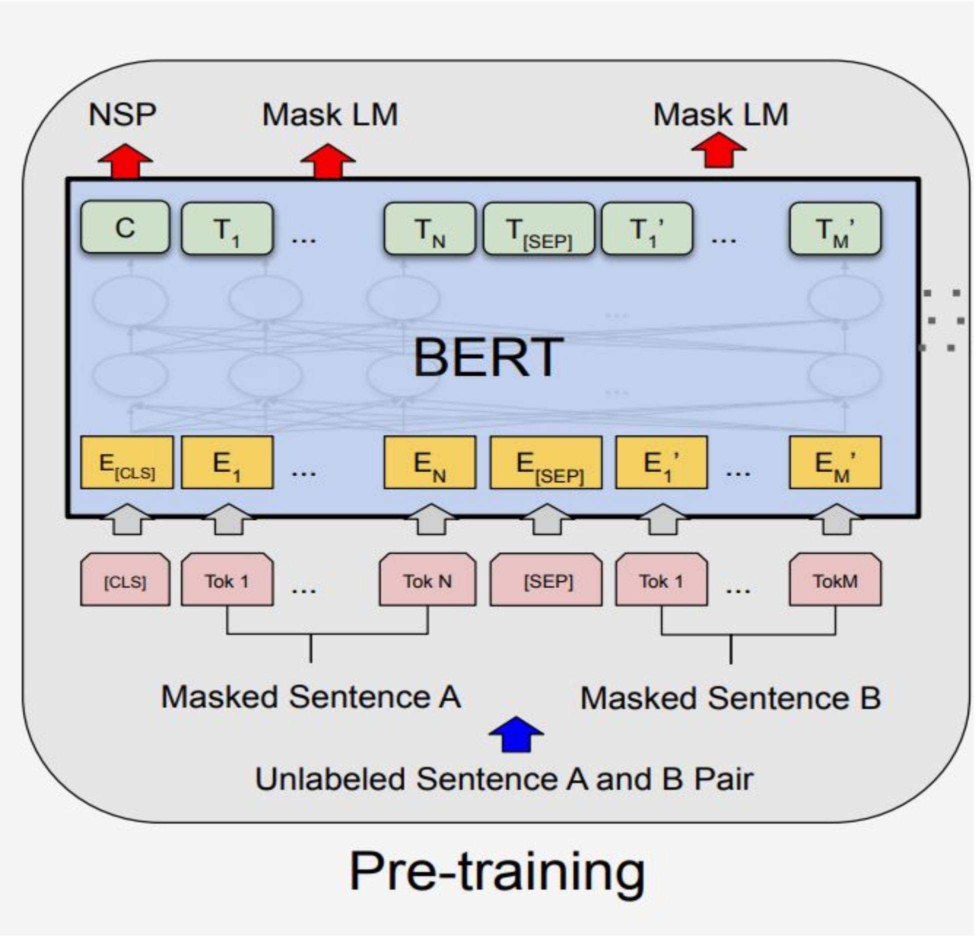
➔ If the context is unique, so is the embedding

In contrast with word2vec, GloVe, FastText, etc, which

- learn one static embedding per word
- learn embeddings based on co-occurrences of word pairs
  e.g., (pet, dog) more likely than (pet, alligator) ... than (pet, taxes)

Canonical approach: BERT

What is BERT?

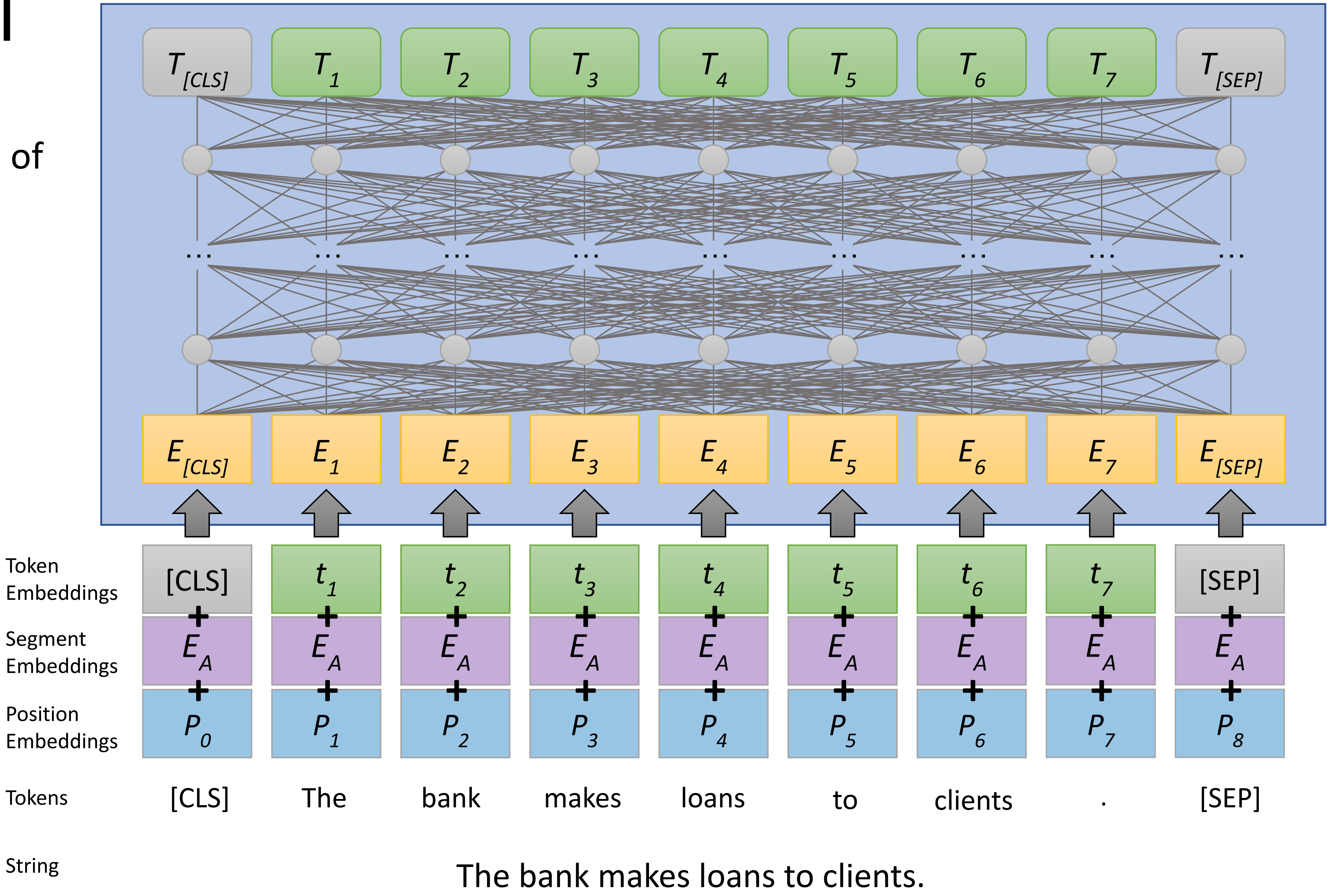


Self-supervised: ∞ training data

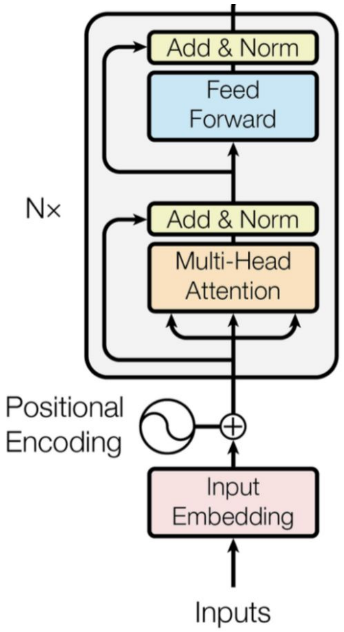
Devlin, Chang, Lee, Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019.

BERT

string → sequence of vectors

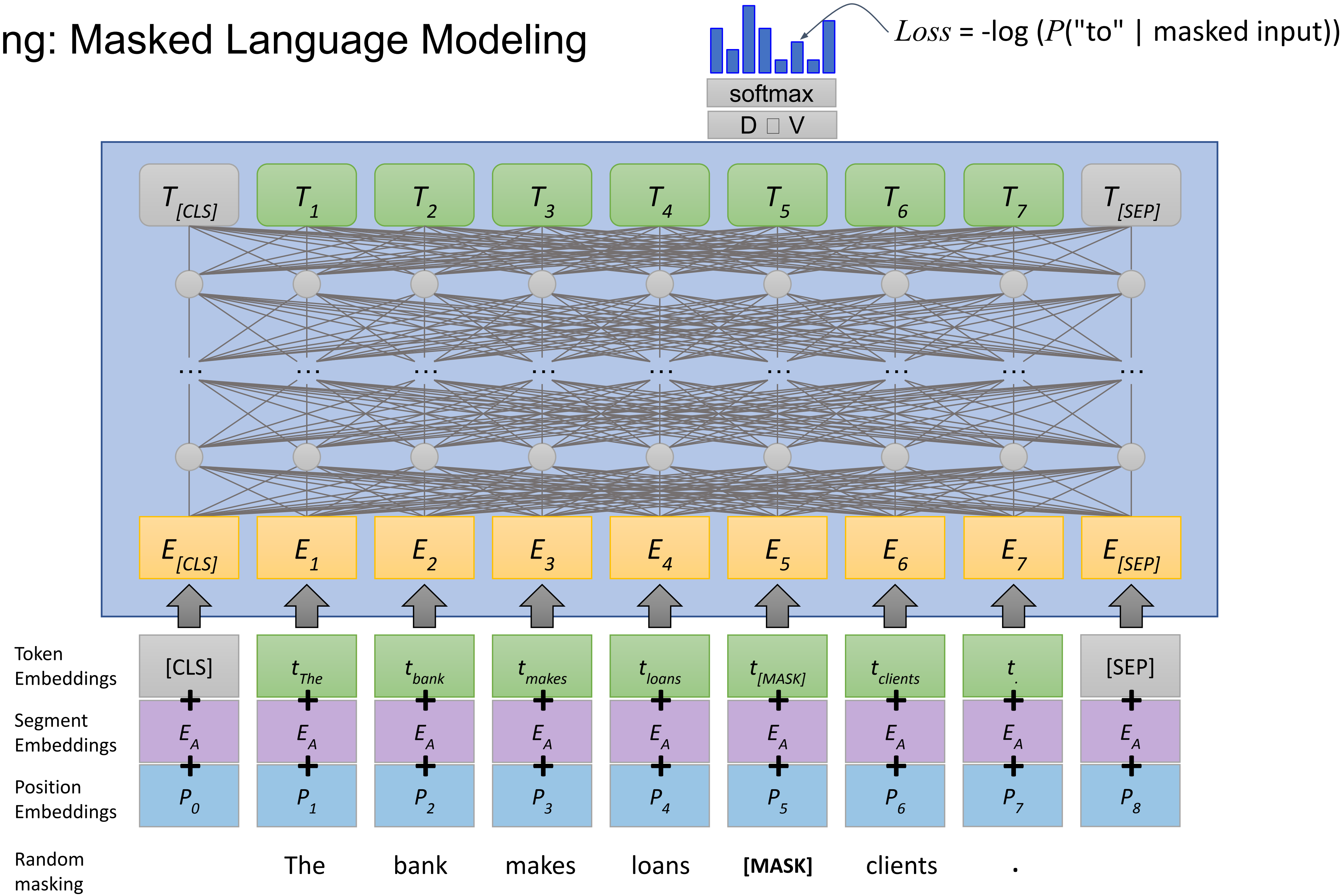


Transformer Encoder

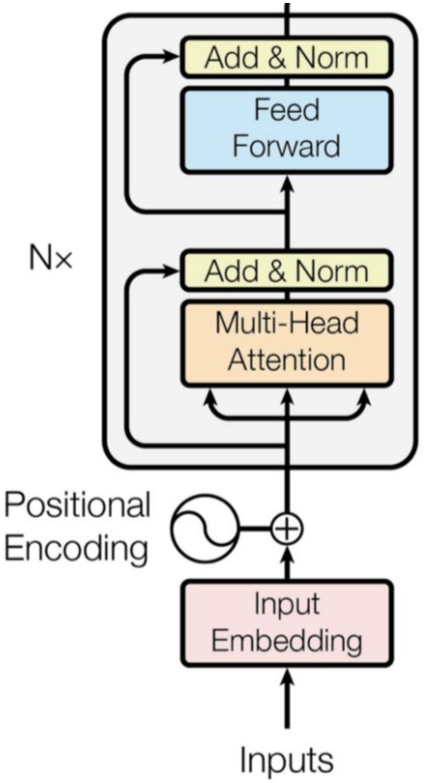


Devlin, Chang, Lee, Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019.

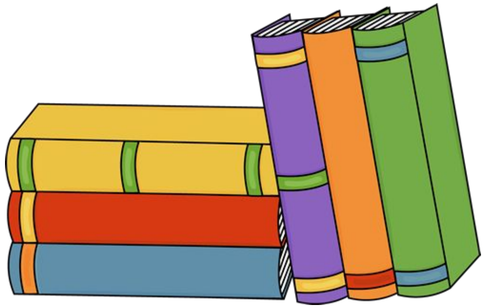
Pretraining: Masked Language Modeling



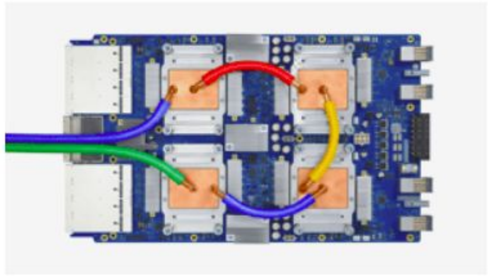
Pretraining ingredients



Lots of parameters
(stack of transformers)



Lots of text



Cloud TPU v3

420 teraflops

128 GB HBM

Cloud TPU v3 Pod

100+ petaflops

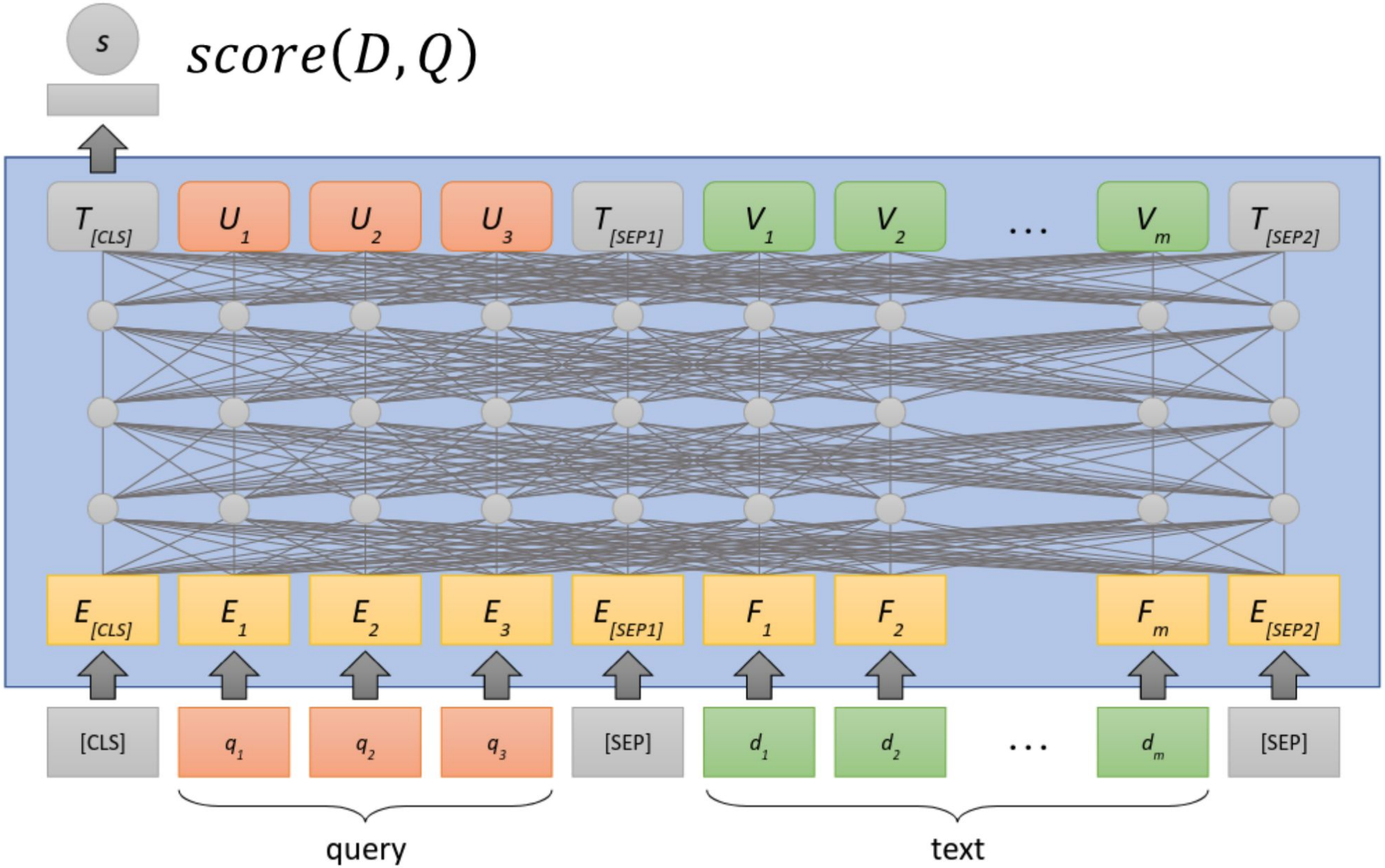
32 TB HBM

Lots of compute

Outline

- Introduction
- Ranking with soft matching
- Transformers & contextualized embeddings
- **Transformer approaches for ranking**
    - Re-ranking with cross-encoders
    - Learned sparse retrieval
    - Document expansion
- Conclusion

Three families of approaches



Cross-encoder

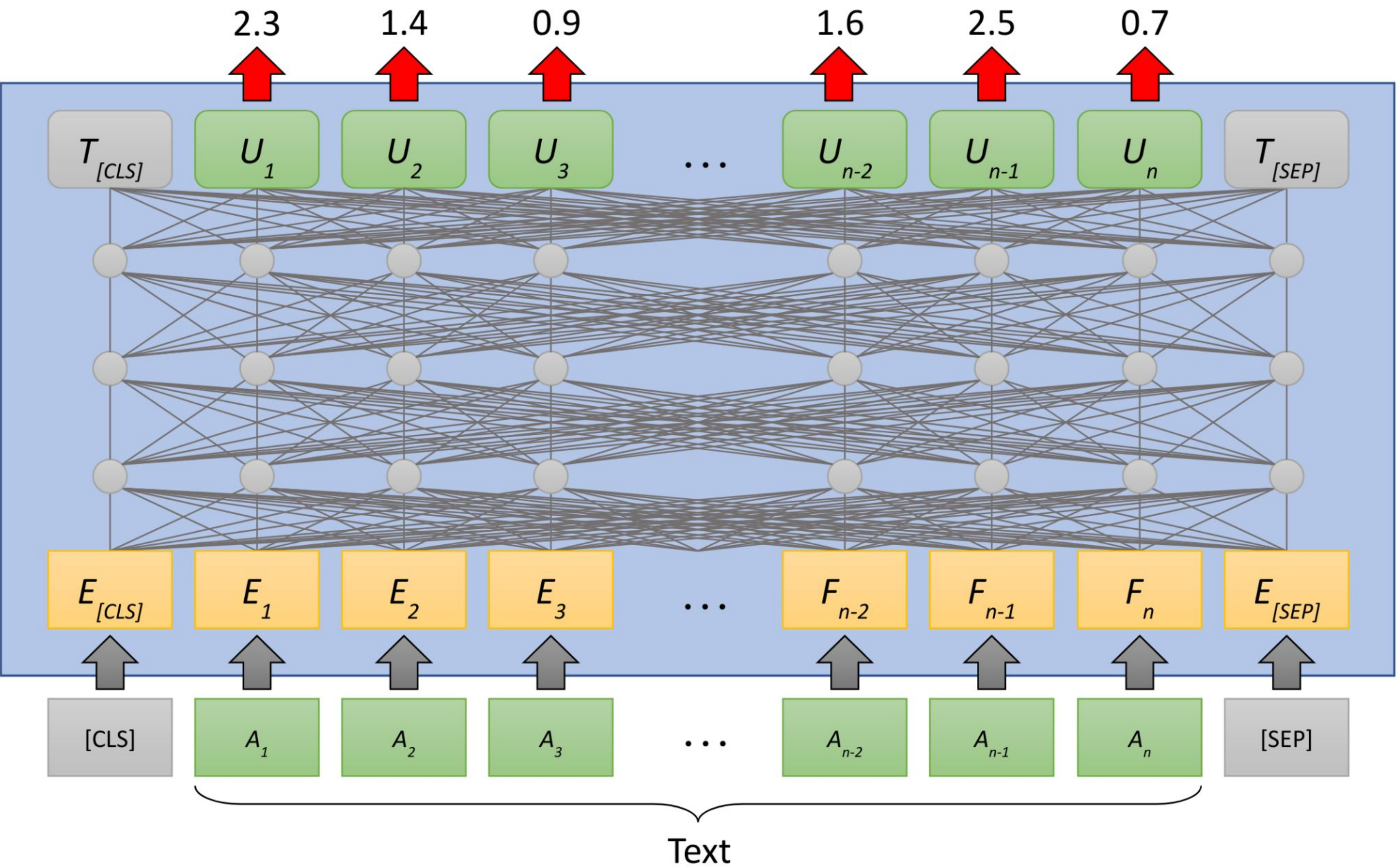
- Input: Q-D pair
- Model outputs score
- Slow but robust

(Family in previous results)

Data structure:

Forward index

Three families of approaches

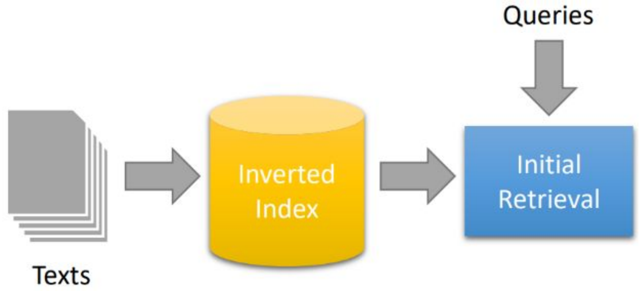


Learned sparse retrieval

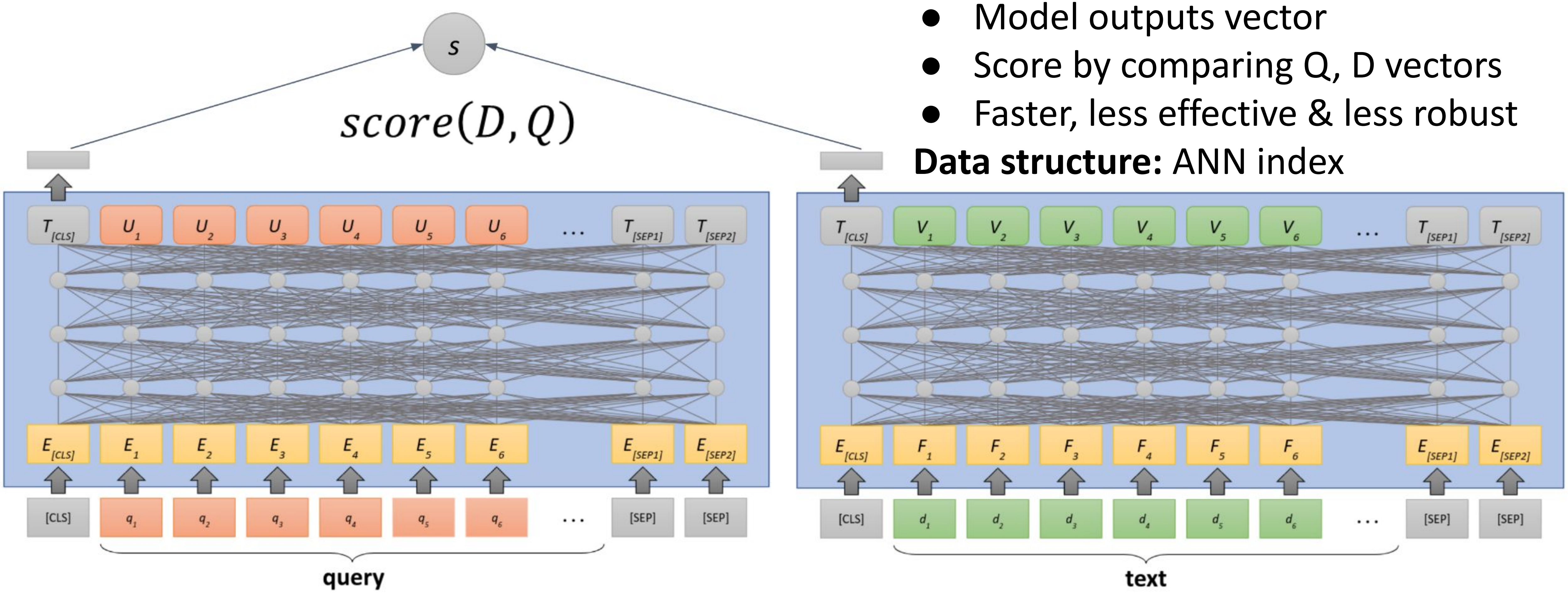
- Input: Q or D
- Model outputs new term weights (replacing TF-IDF)
- Newest / least studied family

Data structure:

Inverted index



Three families of approaches



Bi-encoder

- Input: Q or D
- Model outputs vector
- Score by comparing Q, D vectors
- Faster, less effective & less robust

Data structure: ANN index

Outline

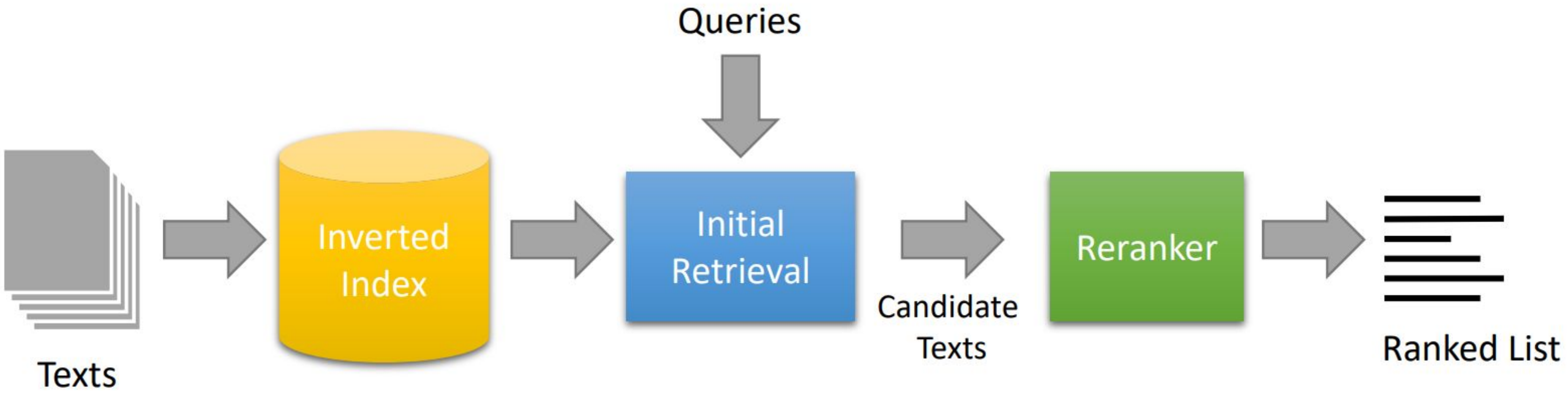
- Introduction
- Ranking with soft matching
- Transformers & contextualized embeddings
- Transformer approaches for ranking
  - **Re-ranking with cross-encoders**
  - Learned sparse retrieval
  - Document expansion
- Conclusion

Re-ranking with cross-encoders

How can we leverage a transformer's improved representations?

Transformer receives Q and D as input, then…

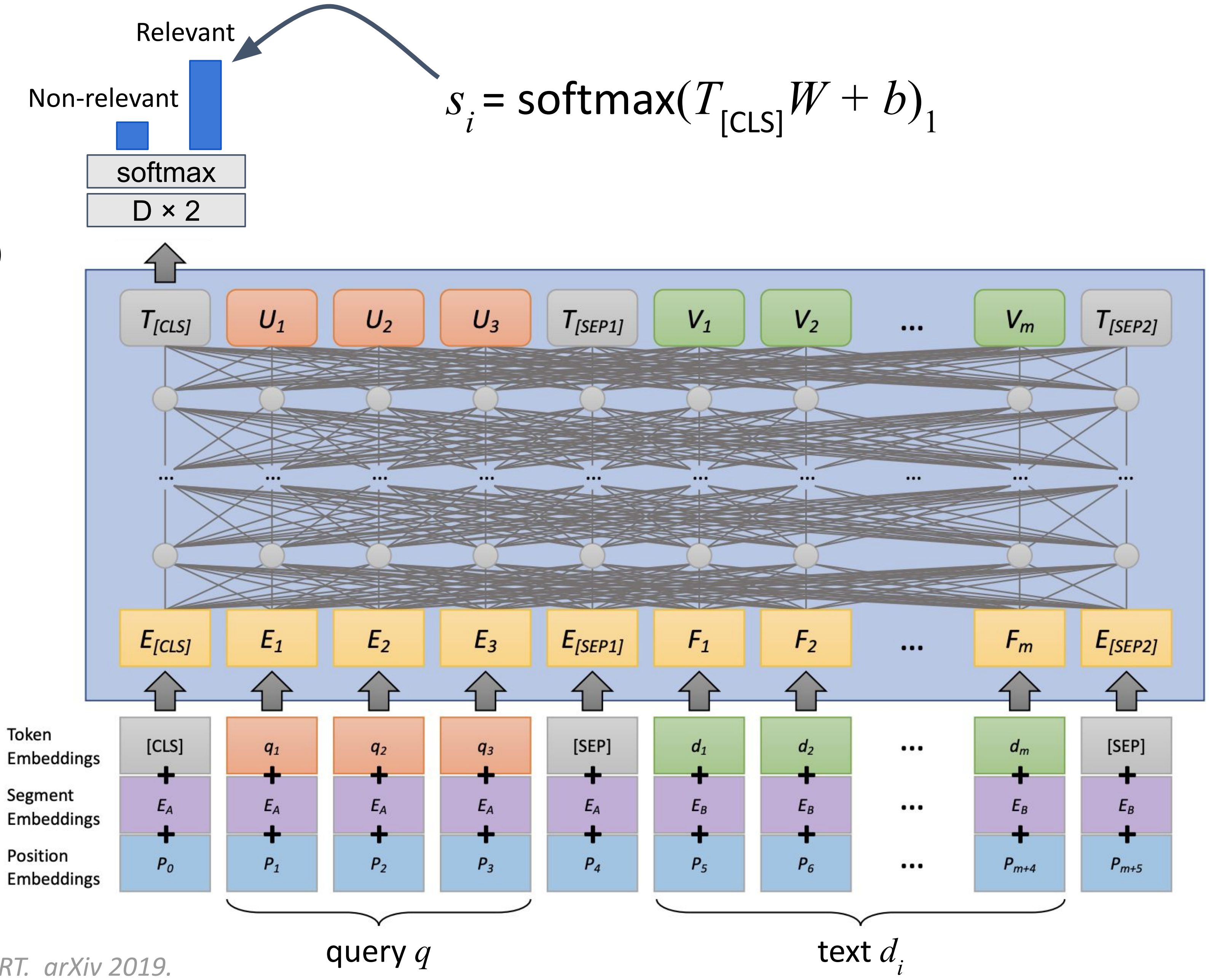
- monoBERT: predicts relevance score directly
- CEDR: produces contextualized embeddings



monoBERT reranker

We want:

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



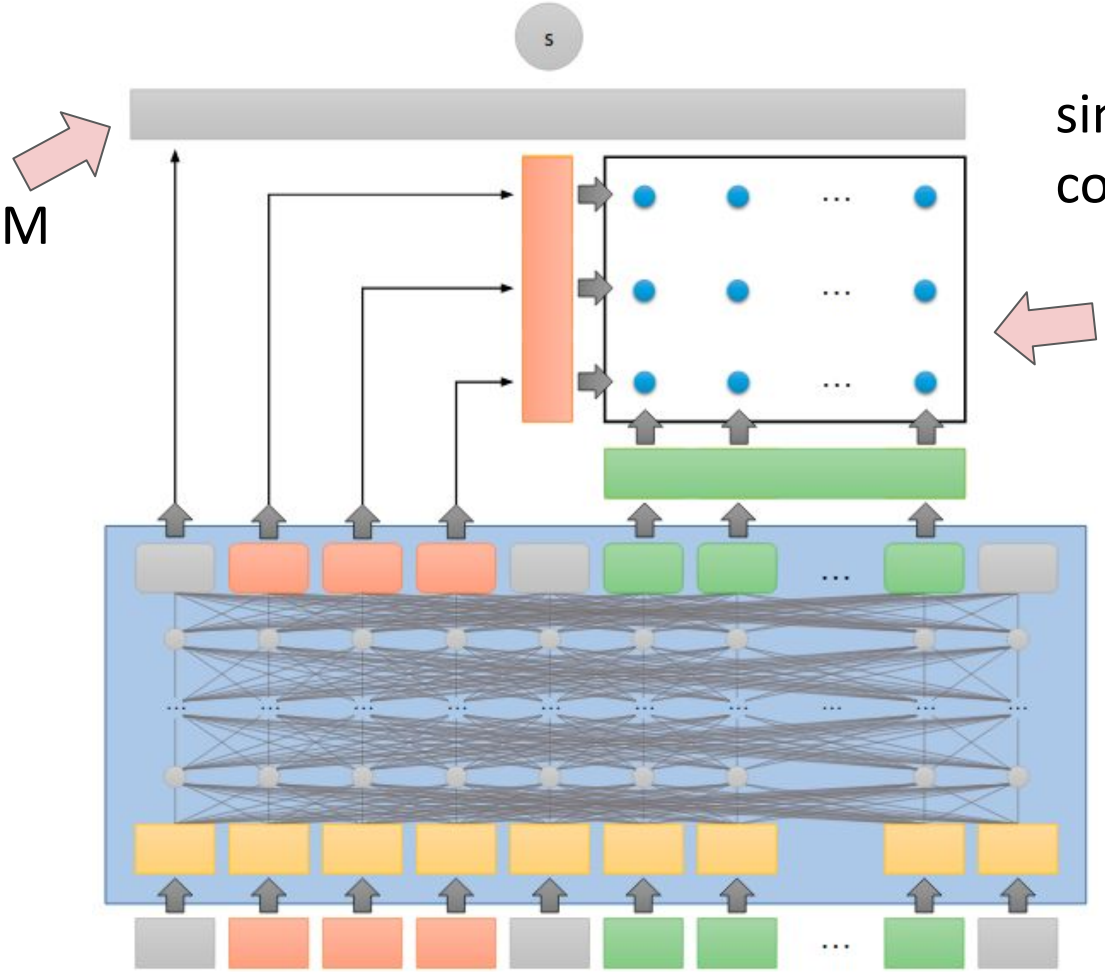
Nogueira and Cho. Passage Re-ranking with BERT. arXiv 2019.

Results: monoBERT on TREC Deep Learning

	nDCG@10	MAP	Recall@1k
BM25	0.506	0.377	0.739
+ monoBERT	**0.738**	**0.506**	0.739
BM25 + RM3	0.518	0.427	0.788
+ monoBERT	**0.742**	**0.529**	0.788

CEDR: Reranking with contextualized embeddings

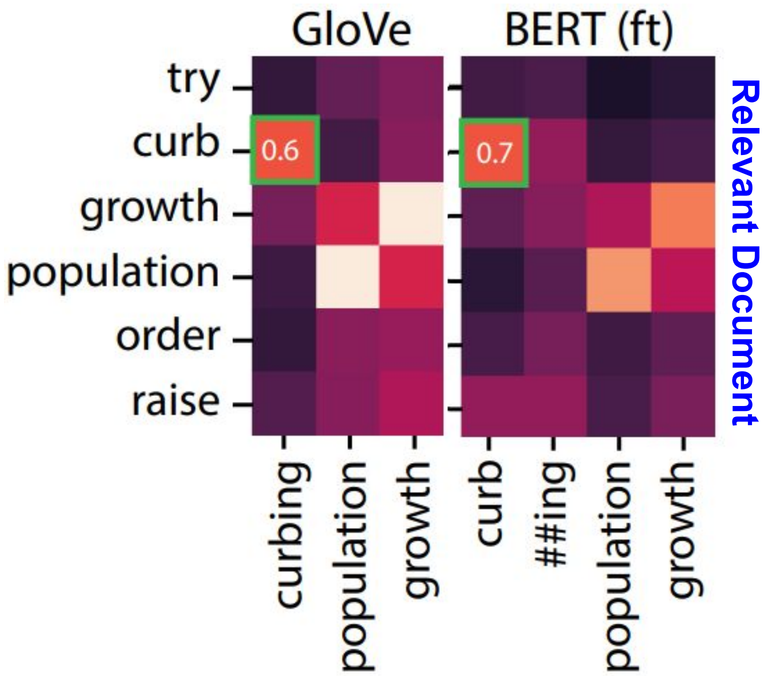
pre-BERT model like DRMM

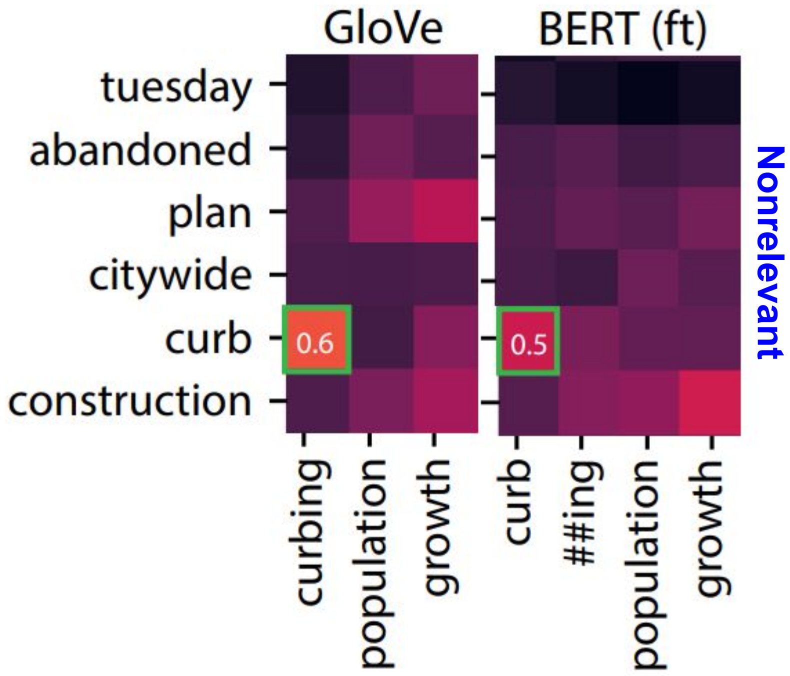


similarity matrix using contextualized embeddings

MacAvaney, Yates, Cohan, Goharian. CEDR: Contextualized Embeddings for Document Ranking. SIGIR 2019.

CEDR: Reranking with contextualized embeddings





MacAvaney, Yates, Cohan, Goharian. CEDR: Contextualized Embeddings for Document Ranking. SIGIR 2019.

Results: CEDR on Robust04

	Method	Input Representation	**Robust04** nDCG@20
(1)	BM25	n/a	0.4140
(2)	Vanilla BERT	BERT (fine-tuned)	[B] 0.4541
(3a)	PACRR	GloVe	0.4043
(3b)	PACRR	BERT	0.4200
(3c)	PACRR	BERT (fine-tuned)	[BVG] 0.5135
(3d)	CEDR–PACRR	BERT (fine-tuned)	**[BVG] 0.5150**
(4a)	KNRM	GloVe	0.3871
(4b)	KNRM	BERT	[G] 0.4318
(4c)	KNRM	BERT (fine-tuned)	[BVG] 0.4858
(4d)	CEDR–KNRM	BERT (fine-tuned)	**[BVGN] 0.5381**
(5a)	DRMM	GloVe	0.3040
(5b)	DRMM	BERT	0.3194
(5c)	DRMM	BERT (fine-tuned)	[G] 0.4135
(5d)	CEDR–DRMM	BERT (fine-tuned)	**[BVGN] 0.5259**

MacAvaney, Yates, Cohan, Goharian. CEDR: Contextualized Embeddings for Document Ranking. SIGIR 2019.

Outline

- Introduction
- Ranking with soft matching
- Transformers & contextualized embeddings
- Transformer approaches for ranking
  - Re-ranking with cross-encoders
  - **Learned sparse retrieval**
  - Document expansion
- Conclusion

DeepCT

$$\text{loss} = \sum_t (\hat{y}_{t,d} - y_{t,d})^2$$

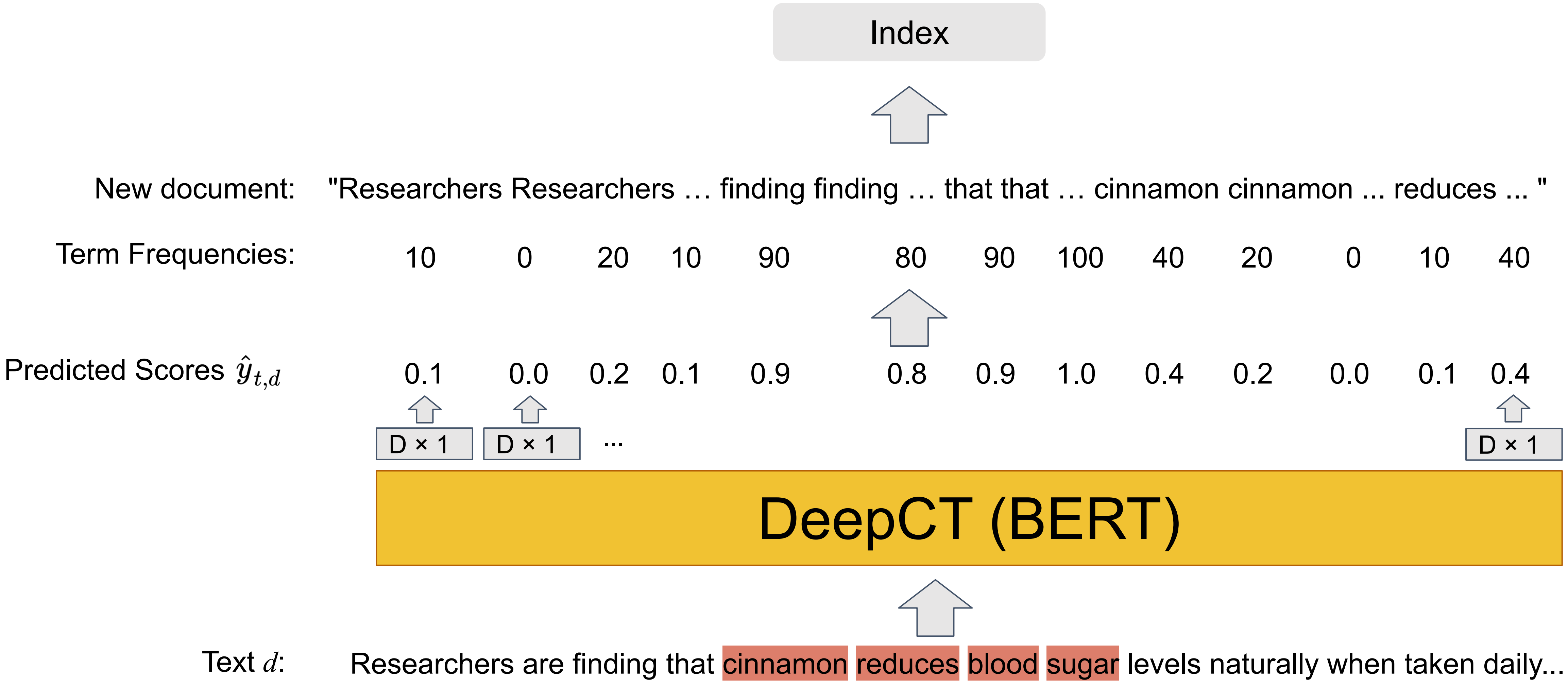
Target Scores	$y_{t,d}$	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Predicted Scores	$\hat{y}_{t,d}$	0.2	0.5	0.2	0.1	0.4	0.1	0.0	0.6	0.2	0.4	0.3

Text d: The Geocentric Theory was proposed by the greeks under the guidance...

Relevant query q: "who proposed the geocentric theory"

Dai, Callan. Context-aware sentence/passage term importance estimation for first stage retrieval. 2019.

Indexing with DeepCT



Results: DeepCT on MS MARCO

Model	**MRR@10**	**R@1000**
BM25	0.184	0.853
+ monoBERT	0.372	0.853
DeepCT	0.243	0.913

SPLADE: Sparse Lexical and Expansion Model

Key improvement: leverage the **MLM head** for weighting and expansion

➔ Recall that MLM head predicts what term should fill in a [MASK]

Input: a query Q or document D

Output: a sparse vector of dimension |V|, to be indexed

1. For each term in the input, use MLM head to predict scores for|V| terms
2. For each term in the vocabulary V, take maximum score as the term's weight
3. Use weights from #2 to represent the input (Q or D) as a |V| vector; index

Results: SPLADE on MS MARCO

Model	MRR@10	R@1000
BM25	0.184	0.853
+ monoBERT	0.372	0.853
SPLADE	0.369	0.979
SPLADE-Doc	0.322	0.946

Outline

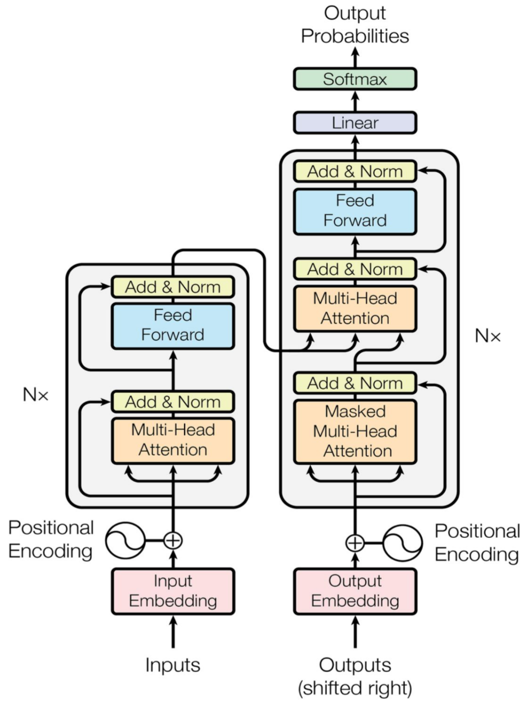
- Introduction
- Ranking with soft matching
- Transformers & contextualized embeddings
- Transformer approaches for ranking
    - Re-ranking with cross-encoders
    - Learned sparse retrieval
    - **Document expansion**
- Conclusion

Document expansion: doc2query

Idea: **generate** possible queries from a given document,
then use them to **expand** the document

Document → seq2seq Transformer → Query

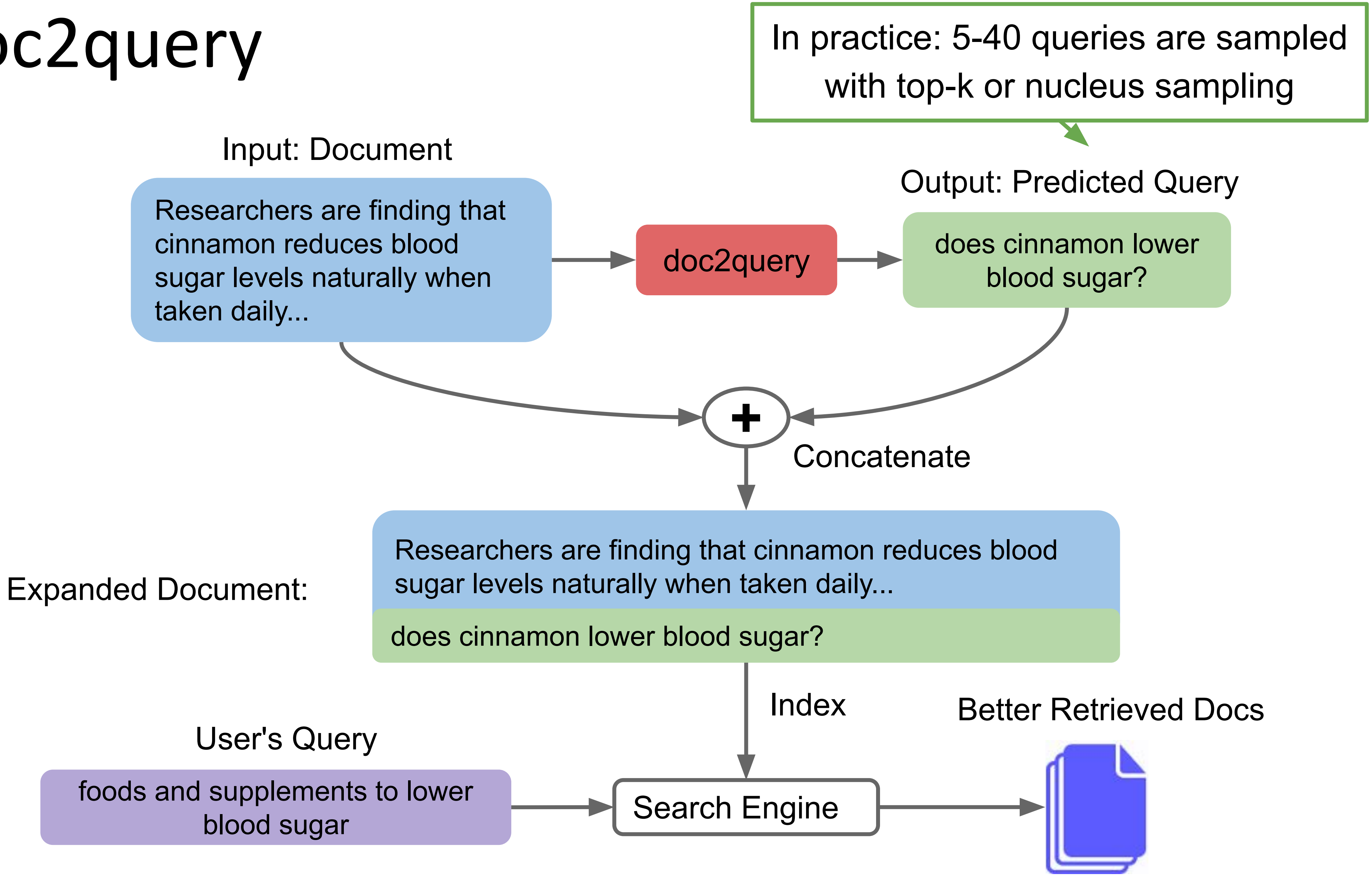
Supervised training:
pairs of <query, relevant document>



Source: Vaswani et al., 2017

Nogueira, Yang, Lin, Cho. Document expansion by query prediction. 2019.

doc2query



In practice: 5-40 queries are sampled with top-k or nucleus sampling
Input: Document
Researchers are finding that cinnamon reduces blood sugar levels naturally when taken daily...
doc2query
Output: Predicted Query
does cinnamon lower blood sugar?
Concatenate
Expanded Document:
Researchers are finding that cinnamon reduces blood sugar levels naturally when taken daily...
does cinnamon lower blood sugar?
Index
Better Retrieved Docs
User's Query
foods and supplements to lower blood sugar
Search Engine

Examples from MS MARCO

Input Document:	July is the hottest month in Washington DC with an average temperature of 27C (80F) and the coldest is January at 4C (38F) with the most daily sunshine hours at 9 in July. The wettest month is May with an average of 100mm of rain.
Predicted Query:	weather in washington dc
Target query:	what is the temperature in washington
Input Document:	The Delaware River flows through Philadelphia into the Delaware Bay. It flows through and aqueduct in the Roundout Reservoir and then flows through Philadelphia and New Jersey before emptying into the Delaware Bay.
Predicted Query:	what river flows through delaware
Target Query:	where does the delaware river start and end
Input Document:	sex chromosome - (genetics) a chromosome that determines the sex of an individual; mammals normally have two sex chromosomes chromosome - a threadlike strand of DNA in the cell nucleus that carries the genes in a linear order; humans have 22 chromosome pairs plus two sex chromosomes.
Predicted Query:	what is the relationship between genes and chromosomes
Target Query:	which chromosome controls sex characteristics

Excluding stop-words:

69% copied

31% new

Results: doc2query on MS MARCO

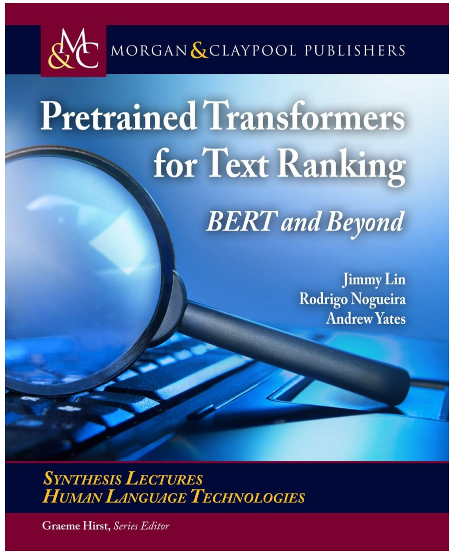
	MRR@10	R@1000
Original Document	.184	.853
+ Expansion New Words	.195	.907
+ Expansion Copied Words	.221	.893
+ Expansion Copied + New	.277	.944
Only Expansions (no original document)	.263	.927

Outline

- Introduction
- Ranking with soft matching
- Transformers & contextualized embeddings
- Transformer approaches for ranking
    - Re-ranking with cross-encoders
    - Learned sparse retrieval
    - Document expansion
- **Conclusion**

Conclusion

- Neural IR approaches can substantially improve search
- Learned sparse retrieval approaches compatible with an inverted index
- Cross-encoder approaches for reranking
- Dense retrieval (bi-encoders) next

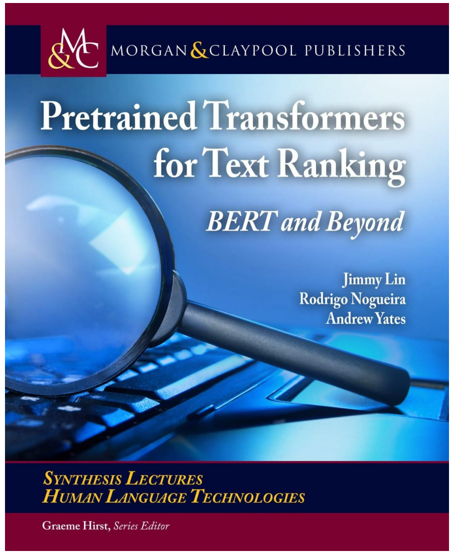


https://arxiv.org/abs/2010.06467
https://bit.ly/tr4tr-book

Conclusion

- Neural IR approaches can substantially improve search
- Learned sparse retrieval approaches compatible with an inverted index
- Cross-encoder approaches for reranking
- Dense retrieval (bi-encoders) next

Thanks!



https://arxiv.org/abs/2010.06467
https://bit.ly/tr4tr-book