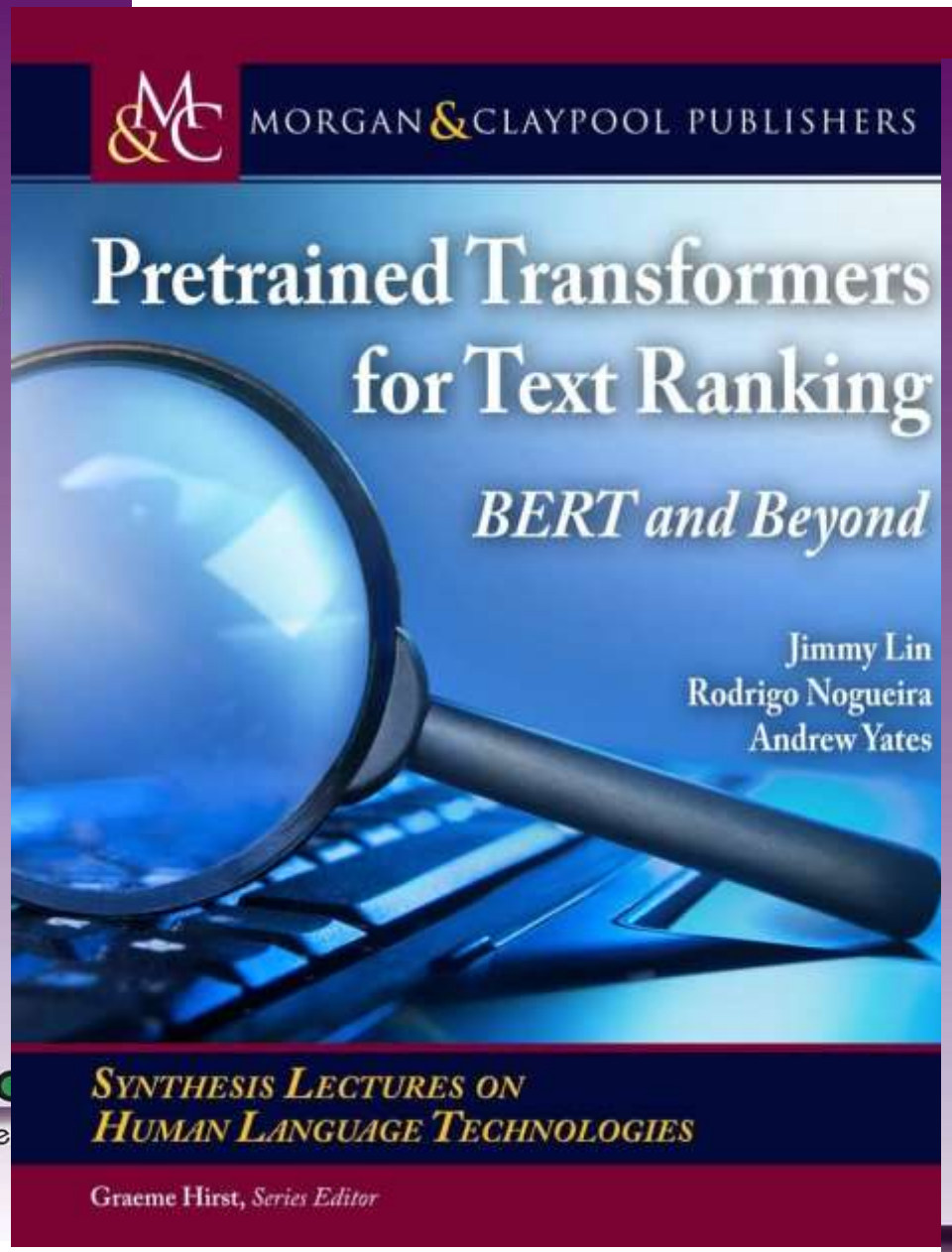
Neural Information Retrieval

Foundations and Trends® in
Information Retrieval
13:1

An Introduction to Neural Information Retrieval

Bhaskar Mitra and Nick Craswell

the essence of knowle



Foundations and Trends® in
Information Retrieval
16:3

Pre-training Methods in Information Retrieval

Yixing Fan, Xiaohui Xie, Yinqiong Cai,
Jia Chen, Xinyu Ma, Xiangsheng Li,
Ruqing Zhang and Jiafeng Guo

now
the essence of knowledge

Outline

- Vector representations
- Pre-training methods in information retrieval

VECTOR REPRESENTATIONS

TYPES OF VECTOR REPRESENTATIONS

Local (or one-hot) representation

Every term in vocabulary T is represented by a binary vector of length |T|, where one position in the vector is set to one and the rest to zero

Distributed representation

Every term in vocabulary T is represented by a real-valued vector of length k. The vector can be *sparse* or *dense*. The vector dimensions may be *observed* (e.g., hand-crafted features) or *latent* (e.g., embedding dimensions).

(a) Local representation **(b)** Distributed representation

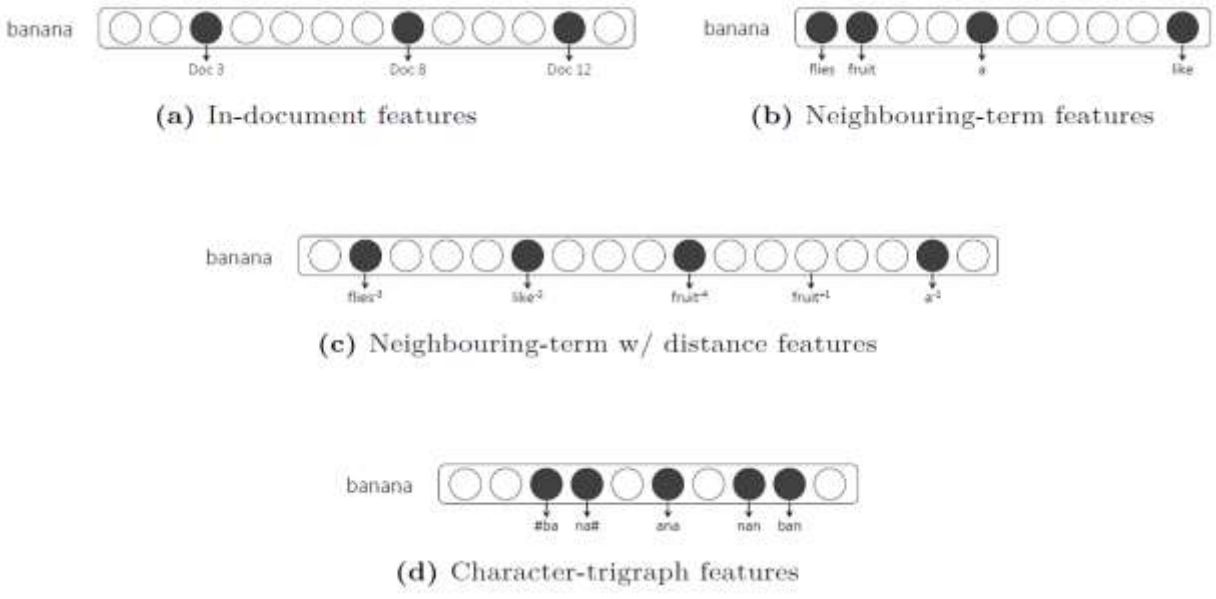
Figure 3.1: Under local representations the terms "banana", "mango", and "dog" are distinct items. But distributed vector representations may recognize that "banana" and "mango" are both fruits, but "dog" is different.

OBSERVED (OR EXPLICIT) DISTRIBUTED REPRESENTATIONS

The choice of features is a key consideration

The distributional hypothesis states that terms that are used (or occur) in similar context tend to be semantically similar [Harris, 1954]

Firth [1957] famously purported this idea of distributional semantics by stating "a word is characterized by the company it keeps".



(a) In-document features (b) Neighbouring-term features

(c) Neighbouring-term w/ distance features

(d) Character-trigraph features

Zellig S Harris. Distributional structure. Word, 10(2-3):146–162, 1954.

Firth, J. R. (1957). A synopsis of linguistic theory 1930–1955. In Studies in Linguistic Analysis, p. 11. Blackwell, Oxford.

Turney and Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research* 2010.

NOTIONS OF SIMILARITY

Two terms are similar if their feature vectors are close

But different feature spaces may capture different notions of similarity

Is *Seattle* more similar to…

Sydney (similar *type*)

or

Seahawks (similar *topic*)

Depends on your choice of features

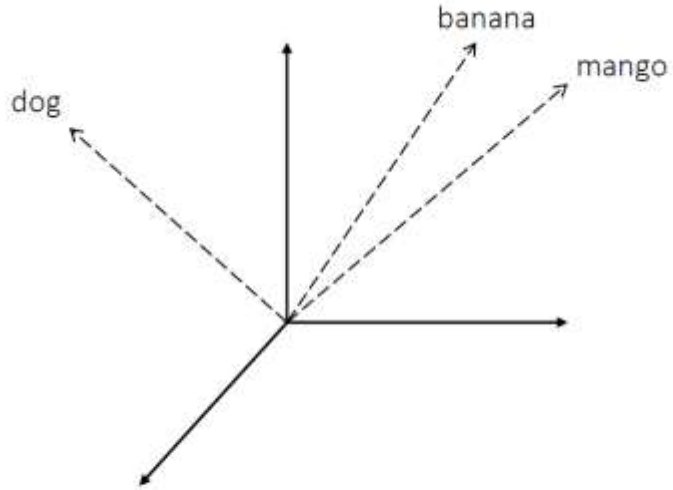


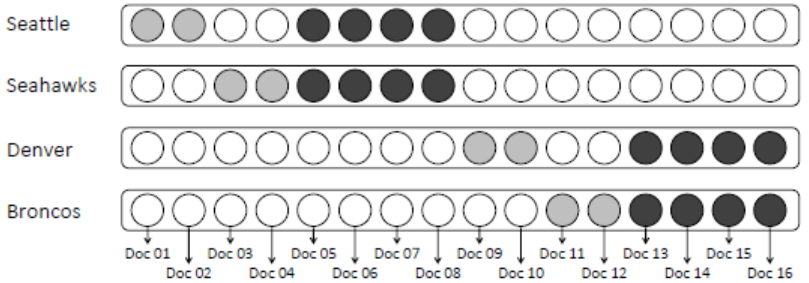
Figure 3.3: A vector space representation of terms puts "banana" closer to "mango" because they share more common attributes than "banana" and "dog".

NOTIONS OF SIMILARITY

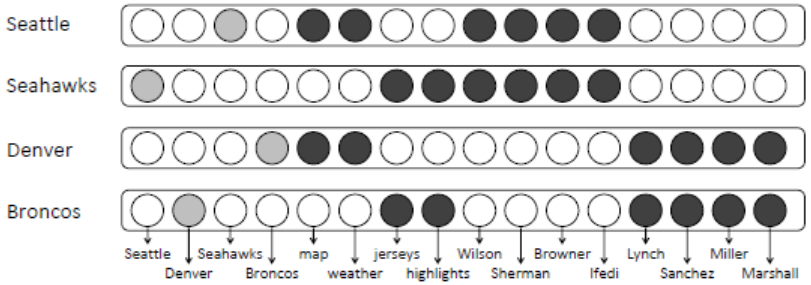
Consider the following toy corpus...

Sample documents			
doc 01	Seattle map	doc 09	Denver map
doc 02	Seattle weather	doc 10	Denver weather
doc 03	Seahawks jerseys	doc 11	Broncos jerseys
doc 04	Seahawks highlights	doc 12	Broncos highlights
doc 05	Seattle Seahawks Wilson	doc 13	Denver Broncos Lynch
doc 06	Seattle Seahawks Sherman	doc 14	Denver Broncos Sanchez
doc 07	Seattle Seahawks Browner	doc 15	Denver Broncos Miller
doc 08	Seattle Seahawks Ifedi	doc 16	Denver Broncos Marshall

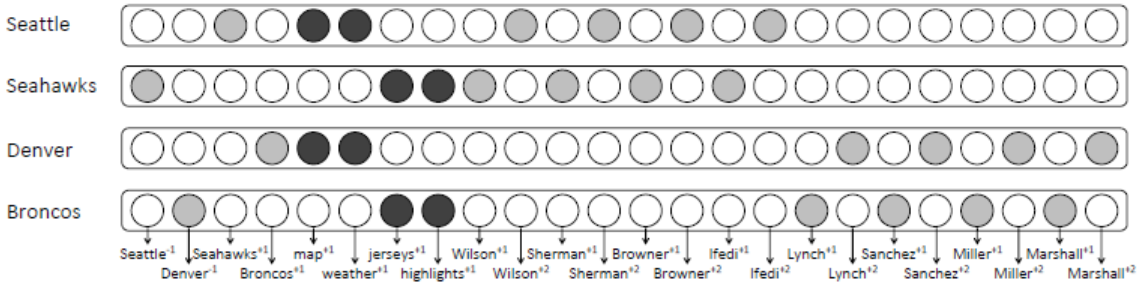
Now consider the different vector representations of terms you can derive from this corpus and how the items that are similar differ in these vector spaces



(a) "In-documents" features



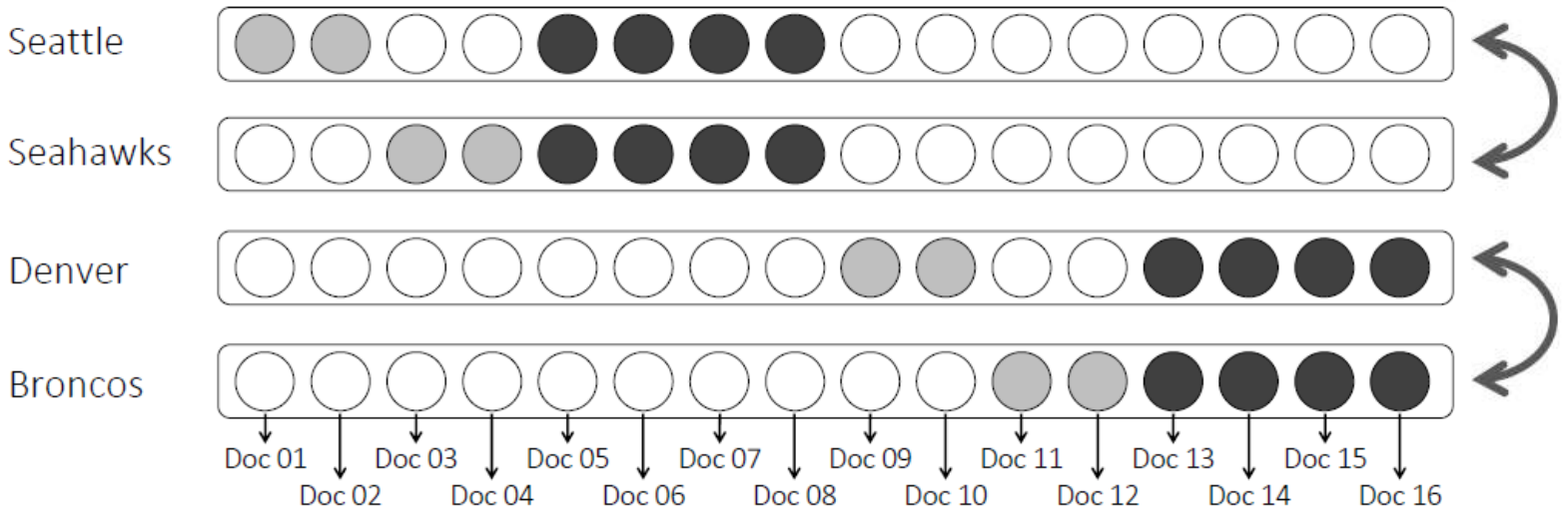
(b) "Neighbouring terms" features



(c) "Neighbouring terms w/ distances" features

NOTIONS OF SIMILARITY

Sample documents			
doc 01	Seattle map	doc 09	Denver map
doc 02	Seattle weather	doc 10	Denver weather
doc 03	Seahawks jerseys	doc 11	Broncos jerseys
doc 04	Seahawks highlights	doc 12	Broncos highlights
doc 05	Seattle Seahawks Wilson	doc 13	Denver Broncos Lynch
doc 06	Seattle Seahawks Sherman	doc 14	Denver Broncos Sanchez
doc 07	Seattle Seahawks Browner	doc 15	Denver Broncos Miller
doc 08	Seattle Seahawks Ifedi	doc 16	Denver Broncos Marshall

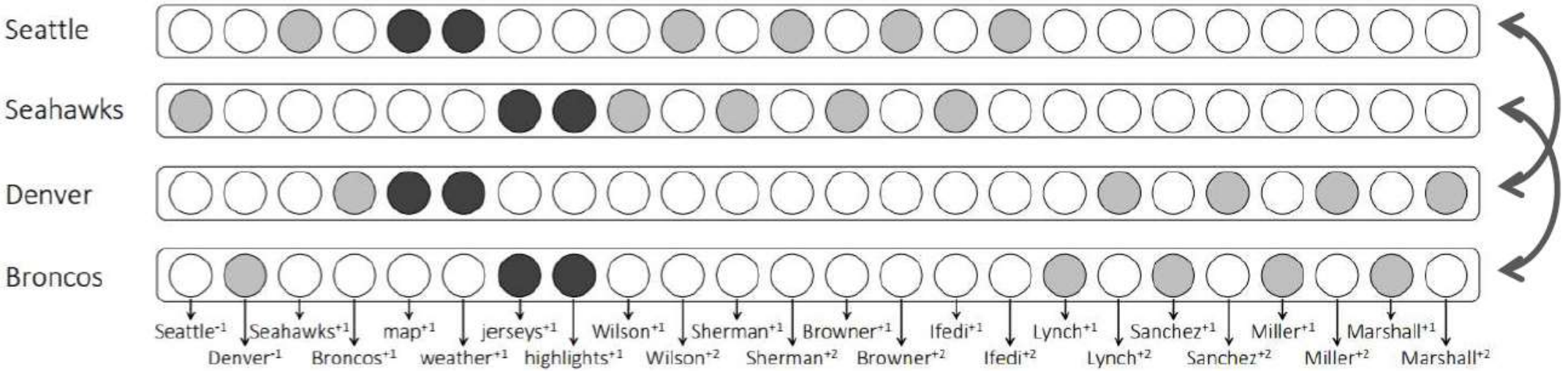


(a) "In-documents" features

Topical or Syntagmatic similarity

NOTIONS OF SIMILARITY

Sample documents			
doc 01	Seattle map	doc 09	Denver map
doc 02	Seattle weather	doc 10	Denver weather
doc 03	Seahawks jerseys	doc 11	Broncos jerseys
doc 04	Seahawks highlights	doc 12	Broncos highlights
doc 05	Seattle Seahawks Wilson	doc 13	Denver Broncos Lynch
doc 06	Seattle Seahawks Sherman	doc 14	Denver Broncos Sanchez
doc 07	Seattle Seahawks Browner	doc 15	Denver Broncos Miller
doc 08	Seattle Seahawks Ifedi	doc 16	Denver Broncos Marshall

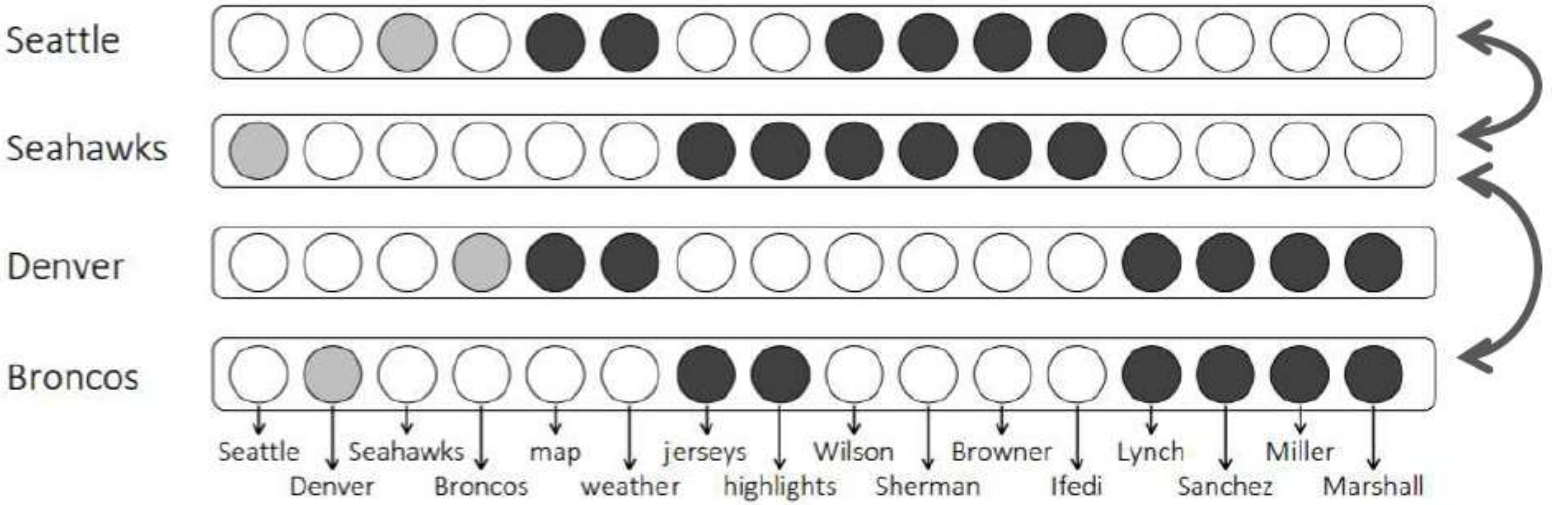


(c) "Neighbouring terms w/ distances" features

Typical or Paradigmatic similarity

NOTIONS OF SIMILARITY

Sample documents			
doc 01	Seattle map	doc 09	Denver map
doc 02	Seattle weather	doc 10	Denver weather
doc 03	Seahawks jerseys	doc 11	Broncos jerseys
doc 04	Seahawks highlights	doc 12	Broncos highlights
doc 05	Seattle Seahawks Wilson	doc 13	Denver Broncos Lynch
doc 06	Seattle Seahawks Sherman	doc 14	Denver Broncos Sanchez
doc 07	Seattle Seahawks Browner	doc 15	Denver Broncos Miller
doc 08	Seattle Seahawks Ifedi	doc 16	Denver Broncos Marshall



(b) "Neighbouring terms" features

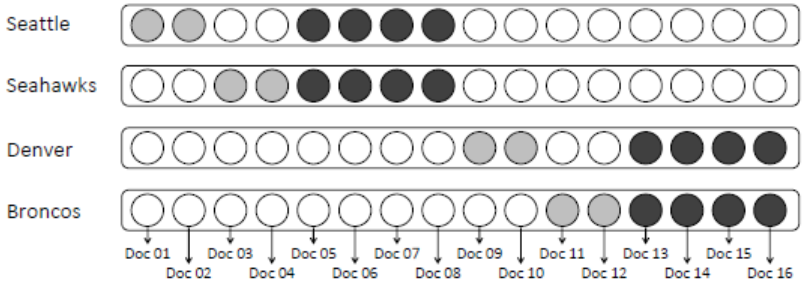
A mix of Topical and Typical similarity

NOTIONS OF SIMILARITY

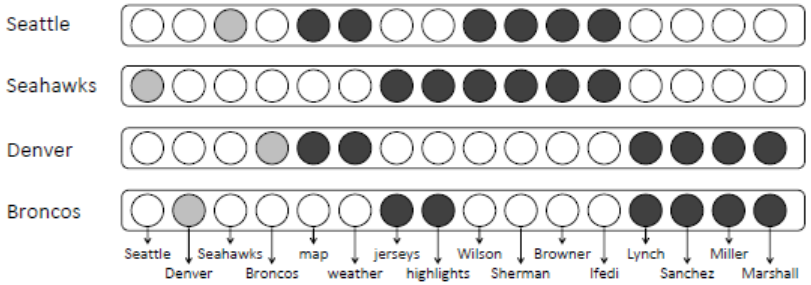
Consider the following toy corpus...

Sample documents			
doc 01	Seattle map	doc 09	Denver map
doc 02	Seattle weather	doc 10	Denver weather
doc 03	Seahawks jerseys	doc 11	Broncos jerseys
doc 04	Seahawks highlights	doc 12	Broncos highlights
doc 05	Seattle Seahawks Wilson	doc 13	Denver Broncos Lynch
doc 06	Seattle Seahawks Sherman	doc 14	Denver Broncos Sanchez
doc 07	Seattle Seahawks Browner	doc 15	Denver Broncos Miller
doc 08	Seattle Seahawks Ifedi	doc 16	Denver Broncos Marshall

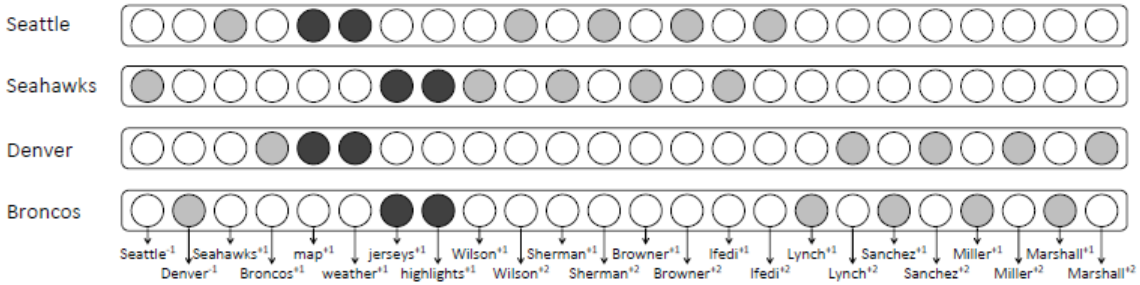
Now consider the different vector representations of terms you can derive from this corpus and how the items that are similar differ in these vector spaces



(a) "In-documents" features



(b) "Neighbouring terms" features



(c) "Neighbouring terms w/ distances" features

RETRIEVAL USING VECTOR REPRESENTATIONS

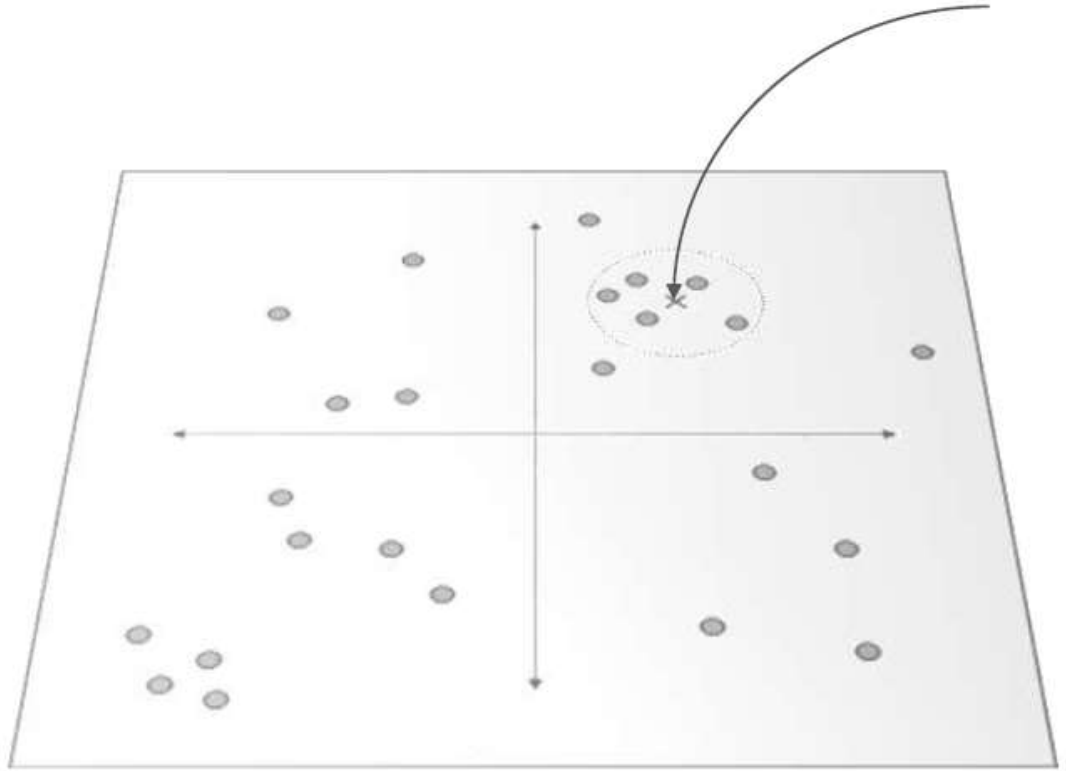
Map both query and candidate documents into the same vector space

Retrieve documents closest to the query

e.g., using Salton's vector space model

$$sim(q,d) = \frac{v_q . v_d}{\|v_q\| . \|v_d\|}$$

Where, v_q and v_d are vectors of TF-IDF scores over all terms in the vocabulary



G. Salton , A. Wong , C. S. Yang, A vector space model for automatic indexing, Communications of the ACM, Nov. 1975

REGULARITIES IN OBSERVED FEATURE SPACES

Some feature spaces capture interesting linguistic regularities

e.g., simple vector algebra in the term-neighboring term space may be useful for word analogy tasks

$$\vec{v}_{king} - \vec{v}_{man} + \vec{v}_{woman} \approx \vec{v}_{queen}$$

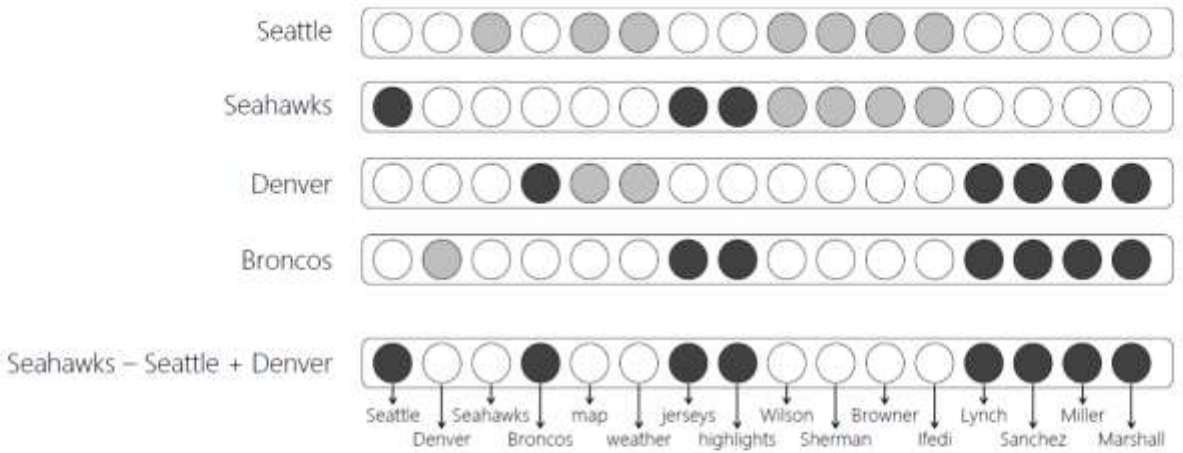


Figure 3.5: A visual demonstration of term analogies via simple vector algebra. The shaded circles denote non-zero values. Darker shade is used to highlight the non-zero values along the vector dimensions for which the output of $\vec{v}_{Seahawks} - \vec{v}_{Seattle} + \vec{v}_{Denver}$ is positive. The output vector is closest to $\vec{v}_{Broncos}$ as shown in this toy example.

Levy, Goldberg and Ramat-Gan. Linguistic Regularities in Sparse and Explicit Word Representations. *CoNLL* 2014

EMBEDDINGS

An embedding is a representation of items in a new space such that the properties of, and the relationships between, the items are preserved from the original representation.

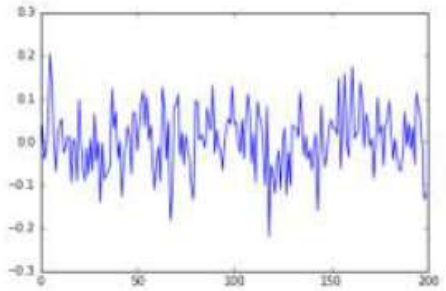
Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. MIT Press, 2016.

EMBEDDINGS

```
In [67]: print(vec['banana'])
         plt.plot(vec['banana'])
```

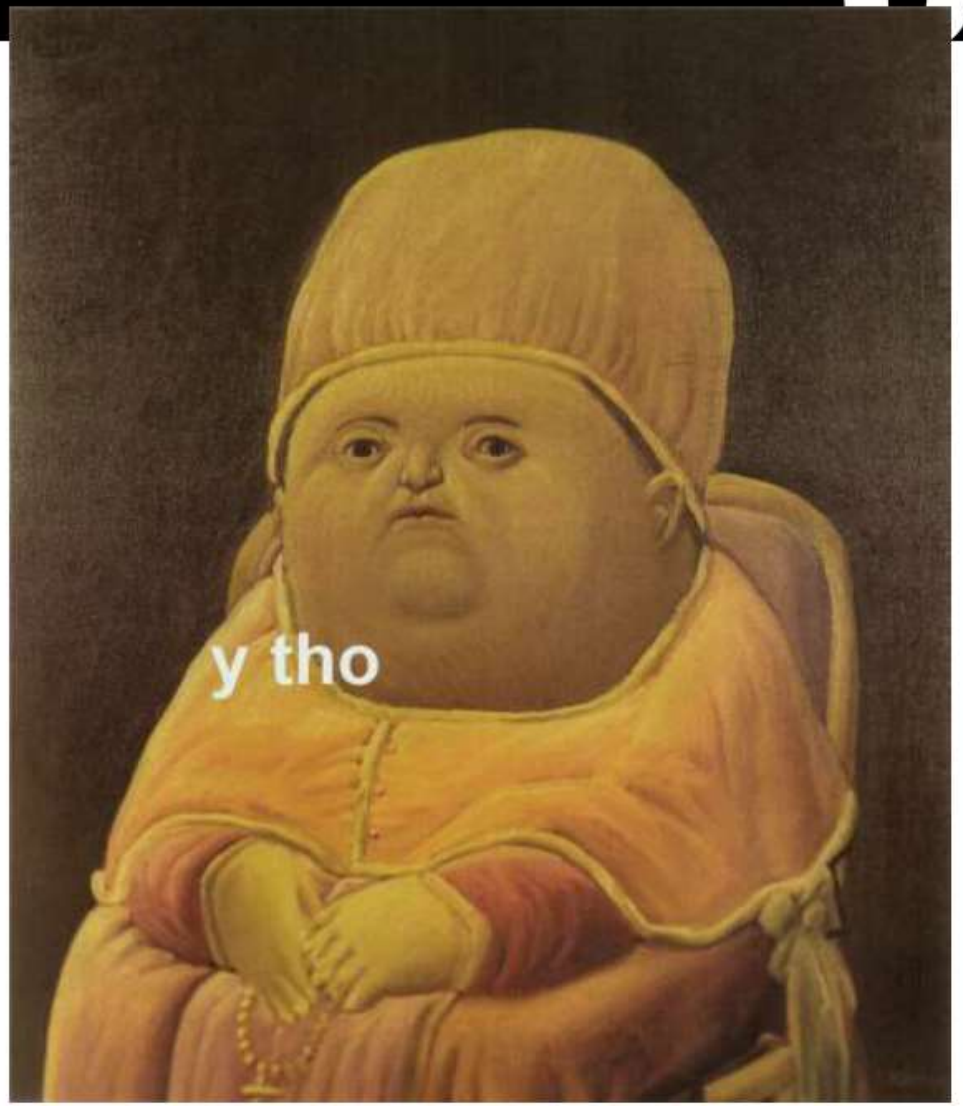
```
[-0.065091, 0.037847, -0.040299, -0.022862, 0.046481, 0.204306, 0.122157, 0.000275, -0.069716, 0.014626, 0.038425, 0.053029, -
0.024947, -0.013991, 0.010317, 0.012735, -0.094237, 0.007101, -0.007268, -0.091869, 0.097138, -0.002357, -0.065102, -0.089856,
 -0.013727, -0.074923, 0.007938, -0.066188, 0.064525, -0.0436, -0.001177, -0.140017, -0.003096, -0.086315, -0.0763, -0.071214,
 -0.051458, 0.123467, 0.031151, 0.068839, -0.039029, 4e-06, -0.127185, -0.049415, -0.007708, 0.035502, 0.009538, -0.075545, 0.0
09583, 0.002794, -0.021556, 0.031155, 0.087352, 0.117663, 0.034883, 0.104613, 0.004534, 0.037999, -0.058016, -0.110670, -0.0353
5, -0.012488, -0.0924, 0.126315, 0.080949, -0.040334, 0.047046, -0.182169, -0.1268, 0.082376, 0.082963, 0.110873, -0.031732, 0.
022219, -0.054332, 0.015394, -0.019853, -0.04169, -0.106969, -0.134253, 0.093094, 0.094716, 0.002643, 0.017417, 0.00309, -0.014
145, 0.078464, 0.041464, 0.026328, 0.12988, -0.02715, 0.027002, -0.014312, -0.017305, -0.066002, 0.002747, 0.033995, 0.053829,
 0.040628, 0.127369, 0.040216, 0.045803, -0.003395, -0.024843, 0.052411, -0.039267, 0.043378, 0.110868, 0.067947, -0.050505, 0.
019753, -0.094825, 0.094058, 0.057547, 0.045447, -0.016258, -0.102323, 0.080506, -0.219069, -0.053595, -0.069609, -0.120579, -
0.048799, -0.019837, -0.109987, -0.002571, 0.031825, -0.124037, -0.024646, -0.102276, 0.038512, 0.035166, 0.031713, 0.008979,
 0.114415, 0.0421, -0.034152, 0.014497, -0.04199, -0.018534, -0.065822, -0.020059, 0.019961, -0.159393, -0.03374, 0.083666, -0.
025234, -0.058921, -0.014924, 0.035292, 0.050979, 0.031609, 0.0322, 0.015638, 0.146793, -0.062475, 0.042192, 0.157084, 0.00237
1, -0.035507, 0.08275, 0.173776, 0.007175, 0.016044, 0.025942, 0.137863, 0.094541, -0.013125, 0.065621, 0.040823, -0.010574, 0.
007796, -0.085031, -0.003617, 0.102267, 0.018047, 0.037613, -0.056187, 0.036693, 0.053867, 0.094616, 0.015941, -0.041536, 0.005
796, -0.03094, -0.063241, -0.067706, -0.026023, 0.069142, -0.008786, 0.042428, -0.017718, 0.03318, -0.052277, 0.114012, 0.00154
2, 0.063282, -0.012149, -0.134274, -0.118431]
```

```
Out[67]: [<matplotlib.lines.Line2D at 0x12a60774e48>]
```



e.g., 200-dimensional term embedding for “banana”

What's the advantage of latent vector spaces over observed features spaces?



LET'S TAKE AN IR EXAMPLE

In Salton's vector space, both these passages are equidistant from the query "Albuquerque"

A latent feature representation may put the first passage closer to the query because of terms like "population" and "area"

Query: "Albuquerque"

Albuquerque is the most populous city in the U.S. state of New Mexico. The high-altitude city serves as the county seat of Bernalillo County, and it is situated in the central part of the state, straddling the Rio Grande. The city population is 557,169 as of the July 1, 2014, population estimate from the United States Census Bureau, and ranks as the 32nd-largest city in the U.S. The Metropolitan Statistical Area (or MSA) has a population of 902,797 according to the United States Census Bureau's most recently available estimate for July 1, 2013.

Passage *about* Albuquerque

Allen suggested that they could program a BASIC interpreter for the device; after a call from Gates claiming to have a working interpreter, MITS requested a demonstration. Since they didn't actually have one, Allen worked on a simulator for the Altair while Gates developed the interpreter. Although they developed the interpreter on a simulator and not the actual device, the interpreter worked flawlessly when they demonstrated the interpreter to MITS in Albuquerque, New Mexico in March 1975; MITS agreed to distribute it, marketing it as Altair BASIC.

Passage *not about* Albuquerque

Embeddings

- Context independent
  - There is just one vector representation for each word
  - Different senses of the word are combined into one single vector
  - Examples: **Word2Vec**, GloVe
- Context dependent
  - Can have multiple vector representations for the same word based on the context
  - Examples: **BERT**, RoBERTA, XLNet

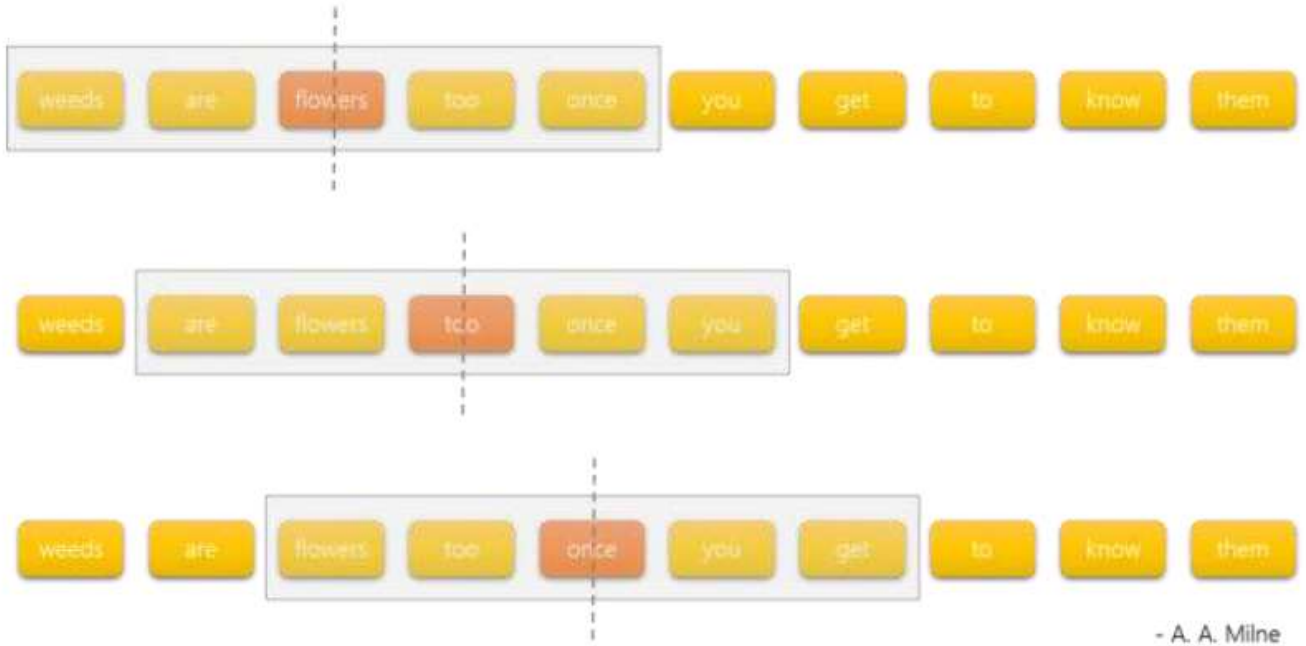
WORD2VEC

Goal: simple (shallow) neural model learning from billion words scale corpus

Predict middle word from neighbors within a fixed size context window

Two different architectures:

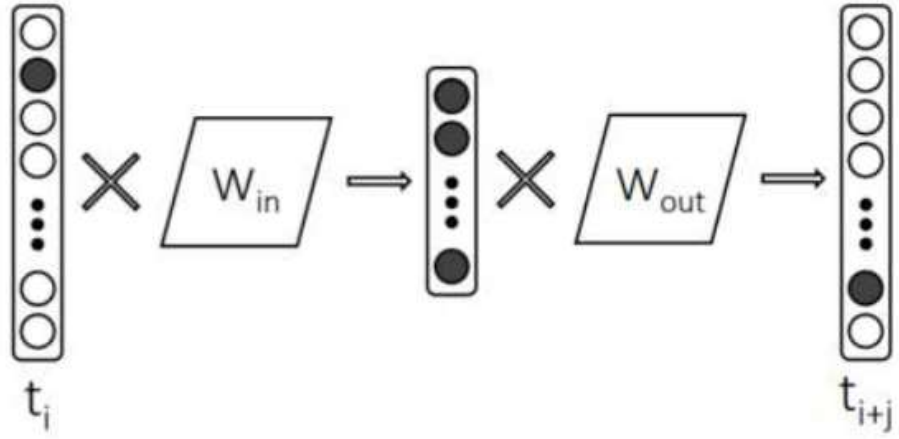
1. Skip-gram
2. CBOW



Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In NIPS, 2013.

SKIP-GRAM

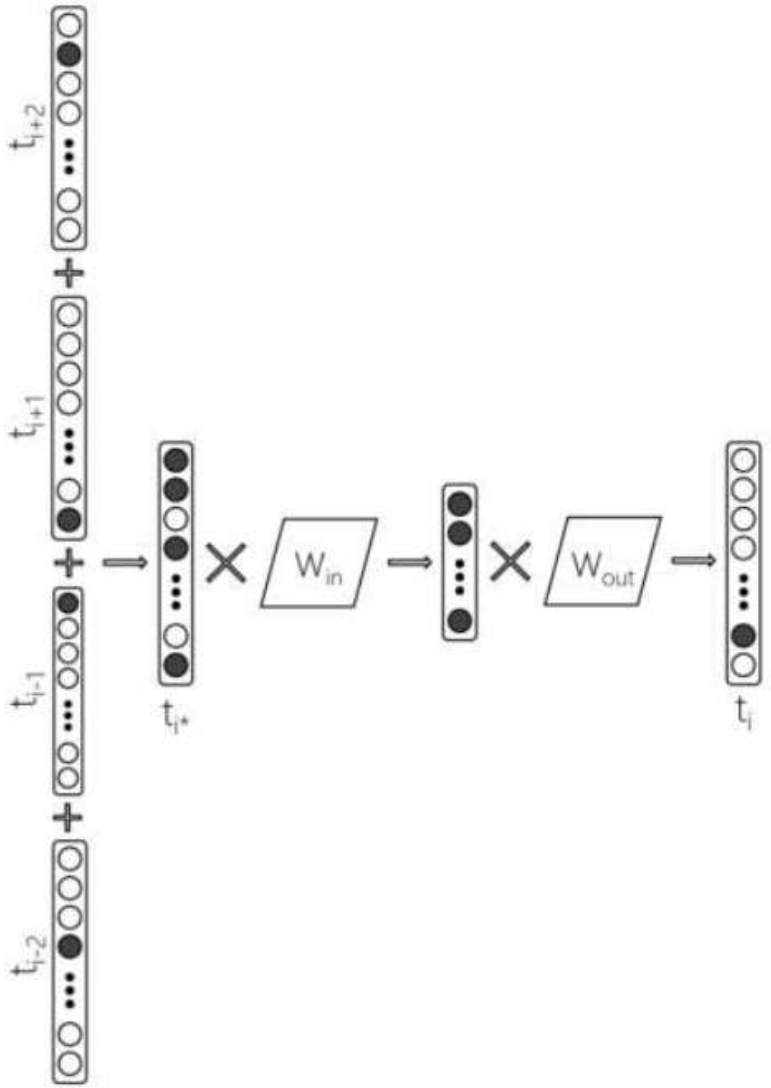
Predict neighbor t_{i+j} given term t_i



(a) Skip-gram

CONTINUOUS BAG-OF-WORDS (CBOW)

Predict the middle term t_i given $\{t_{i-c}, \ldots, t_{i-1}, t_{i+1}, \ldots, t_{i+c}\}$



(b) Continuous bag-of-words (CBOW)

WORD ANALOGIES WITH WORD2VEC

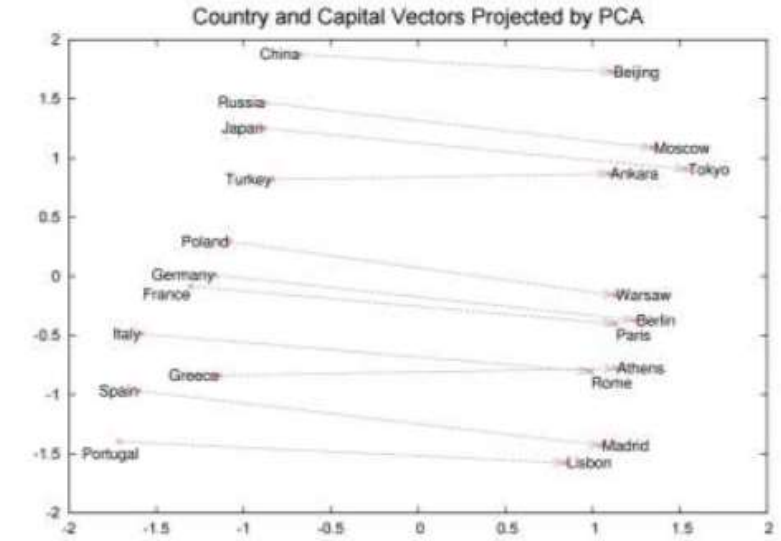
W2v is popular for word analogy tasks

$$\vec{v}_{king} - \vec{v}_{man} + \vec{v}_{woman} \approx \vec{v}_{queen}$$

But remember the same relationships *also* exist in the observed feature space, as we saw earlier

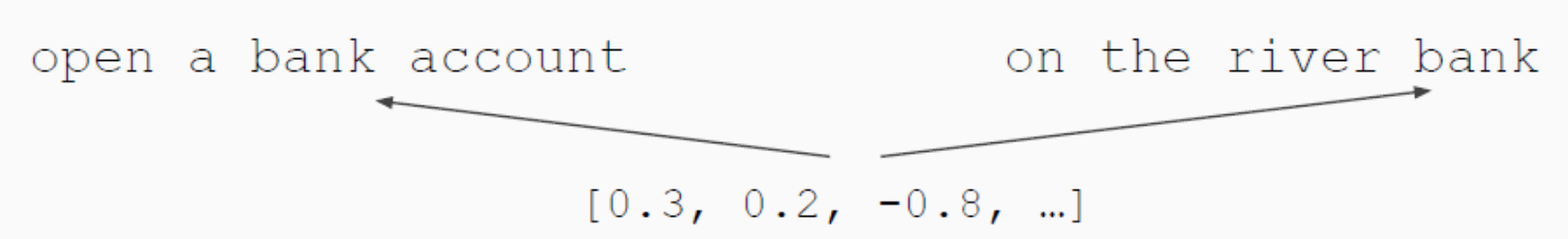
Table 1: *Examples of five types of semantic and nine types of syntactic questions in the Semantic-Syntactic Word Relationship test set.*

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

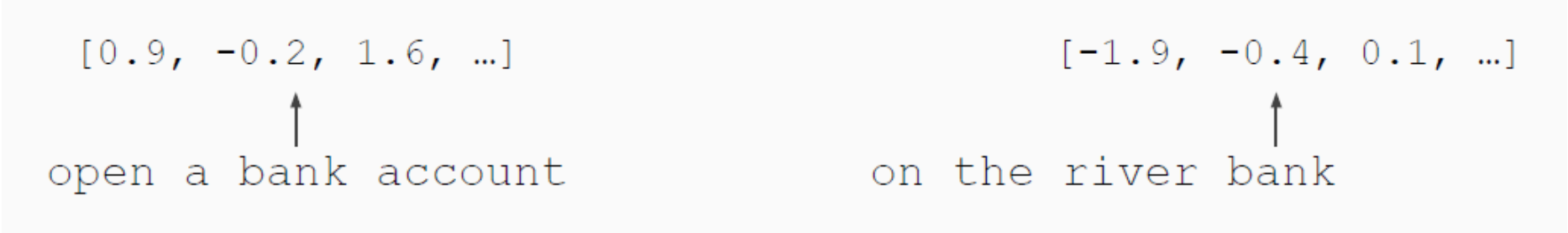


Contextual Representations

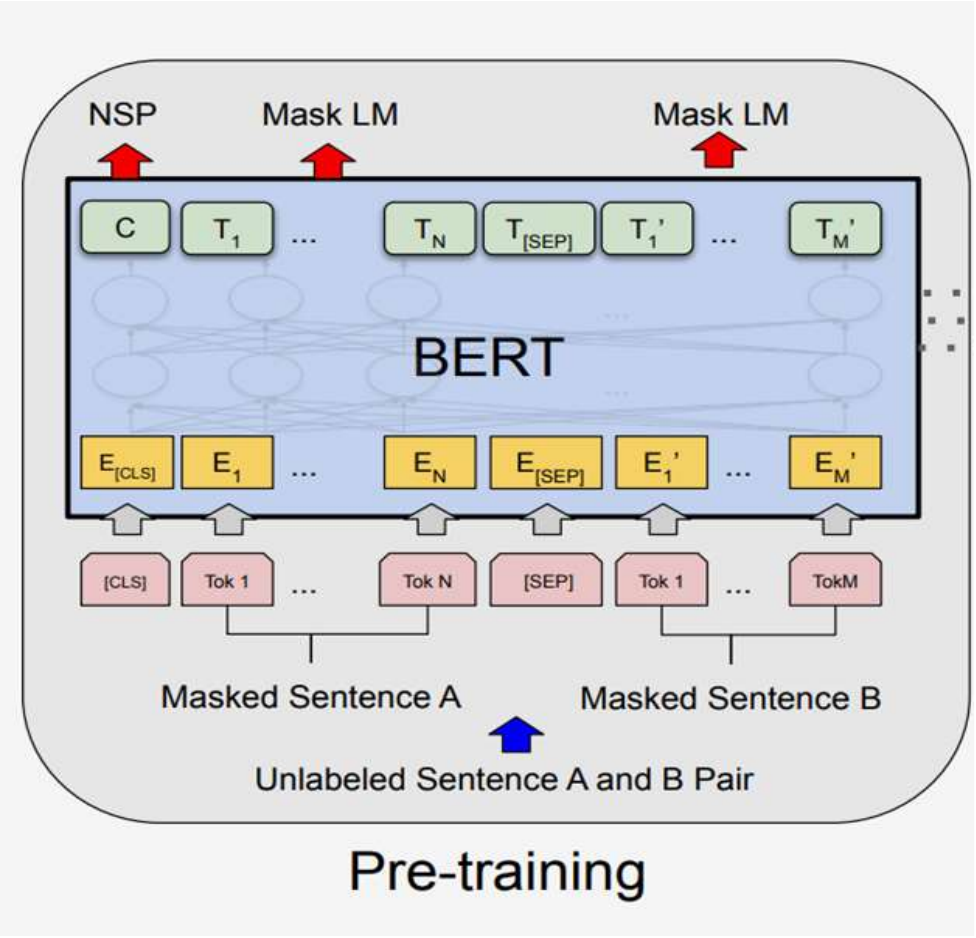
- **Problem:** Word embeddings are applied in a context free manner



- **Solution:** Train *contextual* representations on text corpora



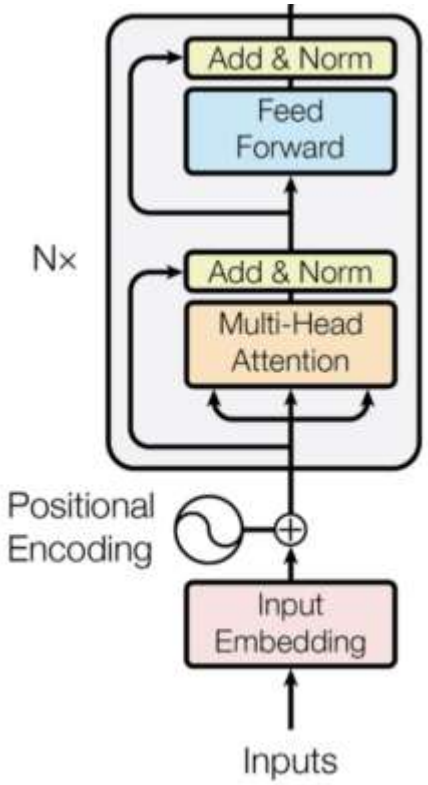
What is BERT?



Self-supervised: ∞ training data

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

BERT's Pretraining Ingredients

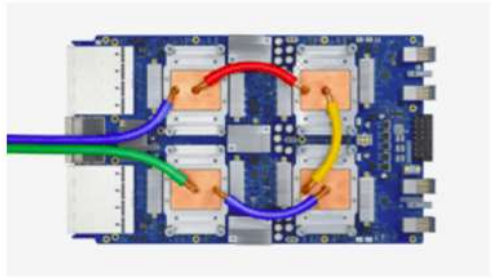


Transformer (encoder-only) with lots of parameters

+

Lots of texts

+



Cloud TPU v3

420 teraflops

128 GB HBM

Cloud TPU v3 Pod

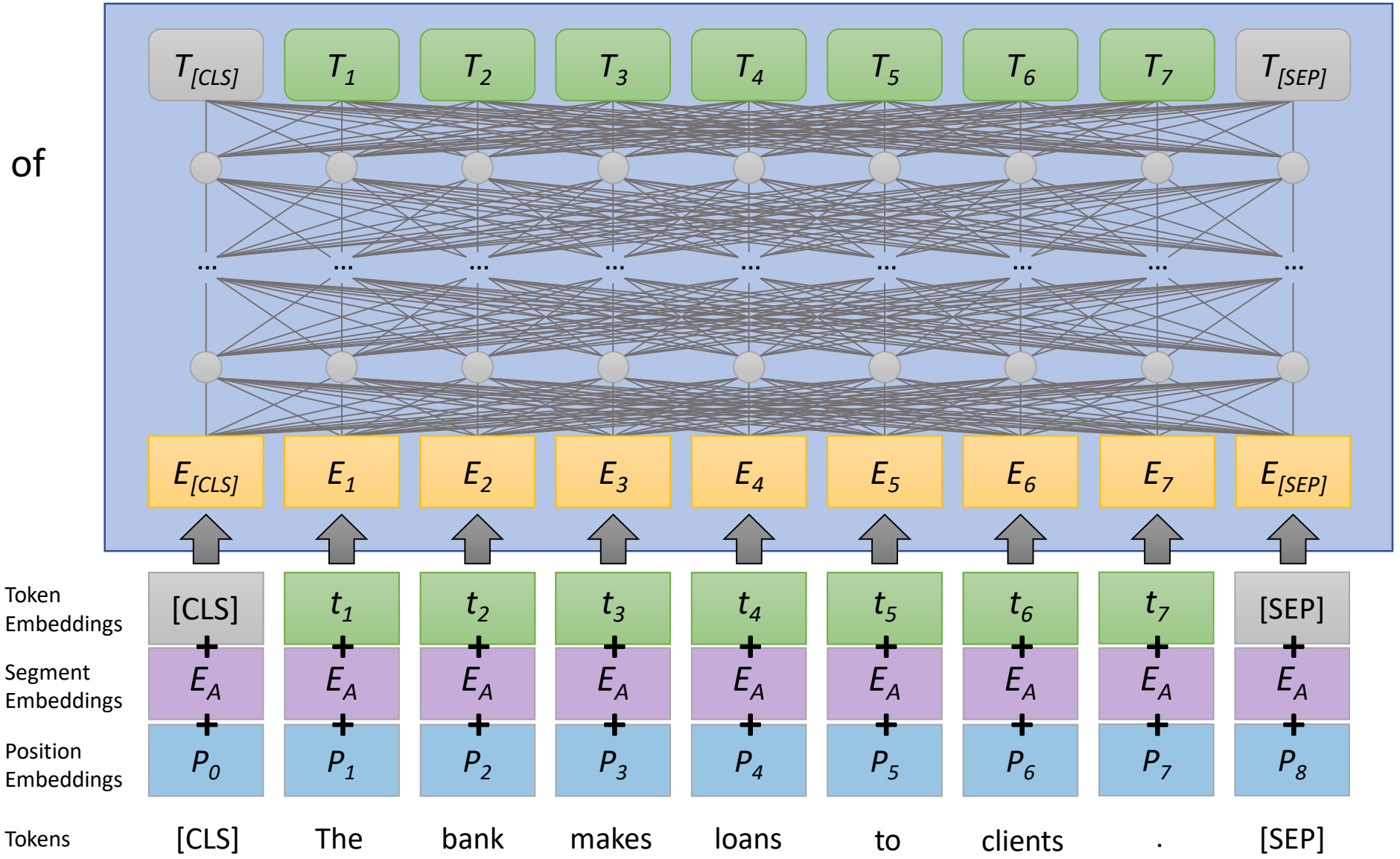
100+ petaflops

32 TB HBM

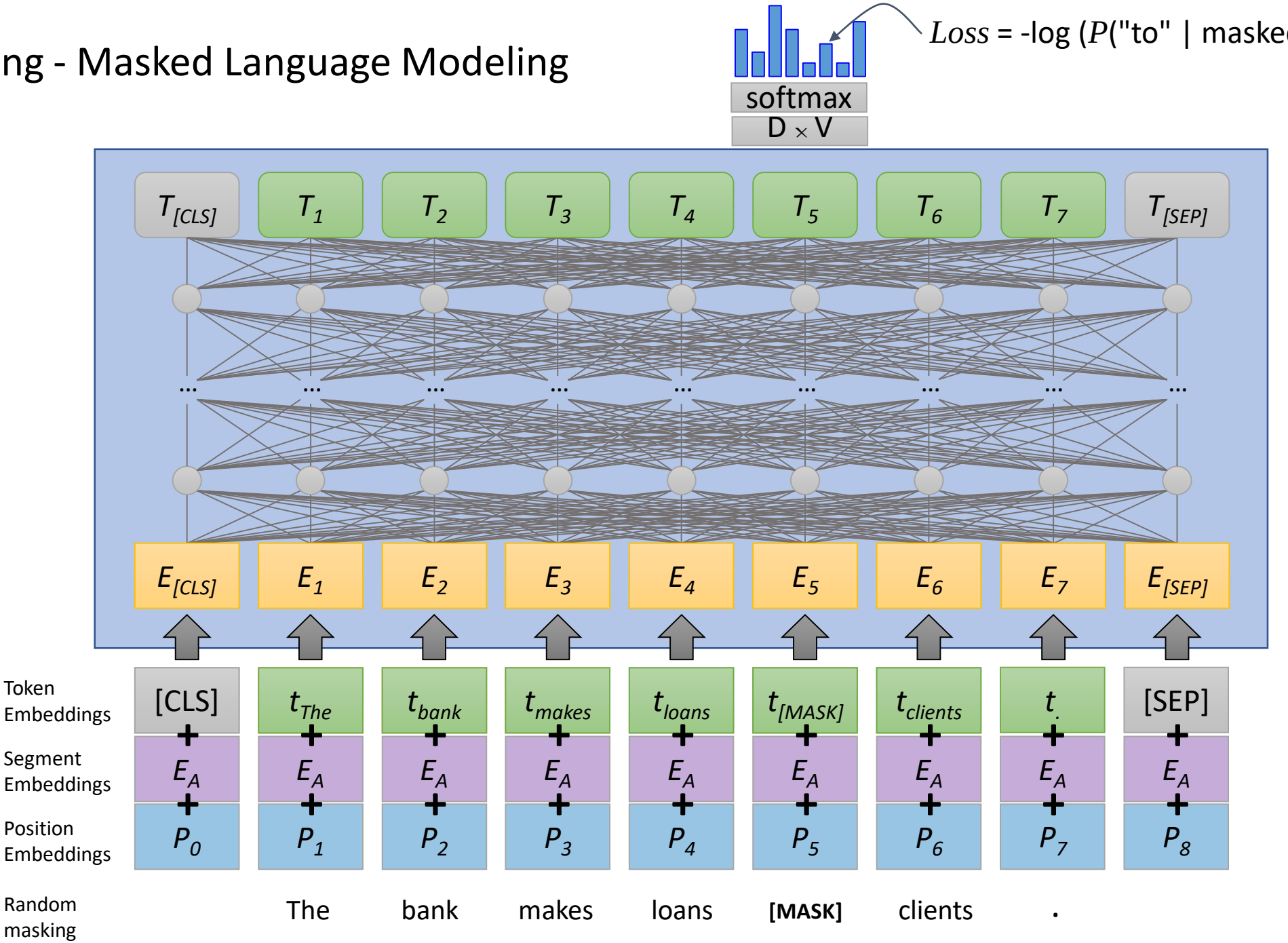
Lots of Compute

BERT

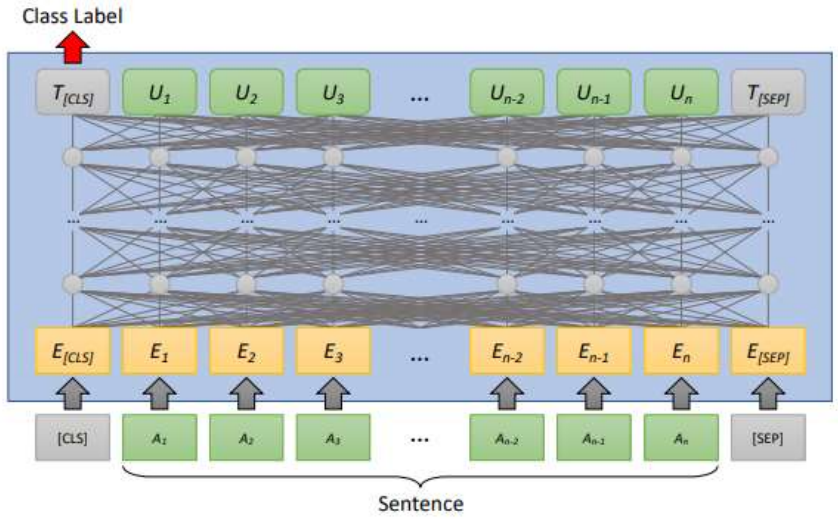
string → sequence of vectors



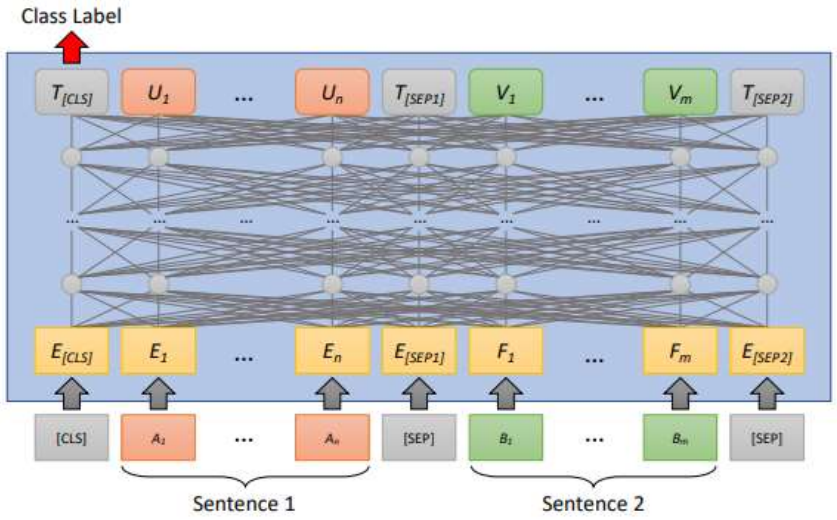
Pretraining - Masked Language Modeling



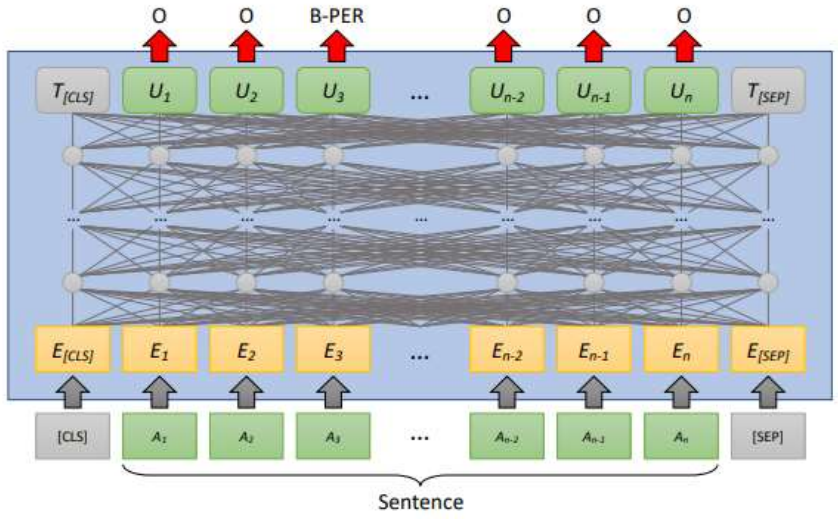
Finetuning



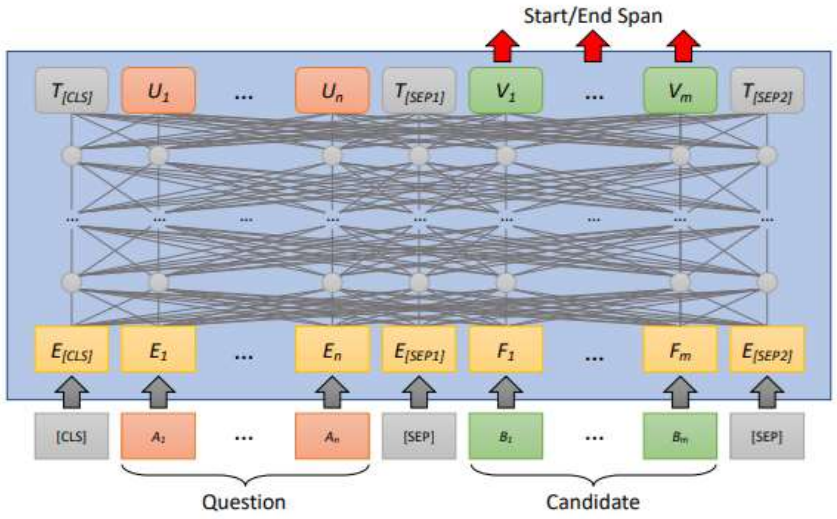
(a) Single-Input Classification Tasks



(b) Two-Input Classification Tasks



(c) Single-Input Sequence Labeling Tasks



(d) Two-Input Sequence Labeling Tasks

Post BERT Pretraining Advancements

- *RoBERTa: A Robustly Optimized BERT Pretraining Approach* (Liu et al, University of Washington and Facebook, 2019)
- *XLNet: Generalized Autoregressive Pretraining for Language Understanding* (Yang et al, CMU and Google, 2019)
- *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations* (Lan et al, Google and TTI Chicago, 2019)
- *T5:Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer* (Raffel et al, Google, 2019)
- *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators* (Clark et al, 2020)
- *GPT2: Language Models are Unsupervised Multitask Learners* (Radford et al, OpenAI, 2019)

PTMs in Information Retrieval

- Pre-BERT models
- BERT based models and more

RECAP: RETRIEVAL USING VECTOR REPRESENTATIONS

Generate vector representation of query

Generate vector representation of document

Estimate relevance from q-d vectors

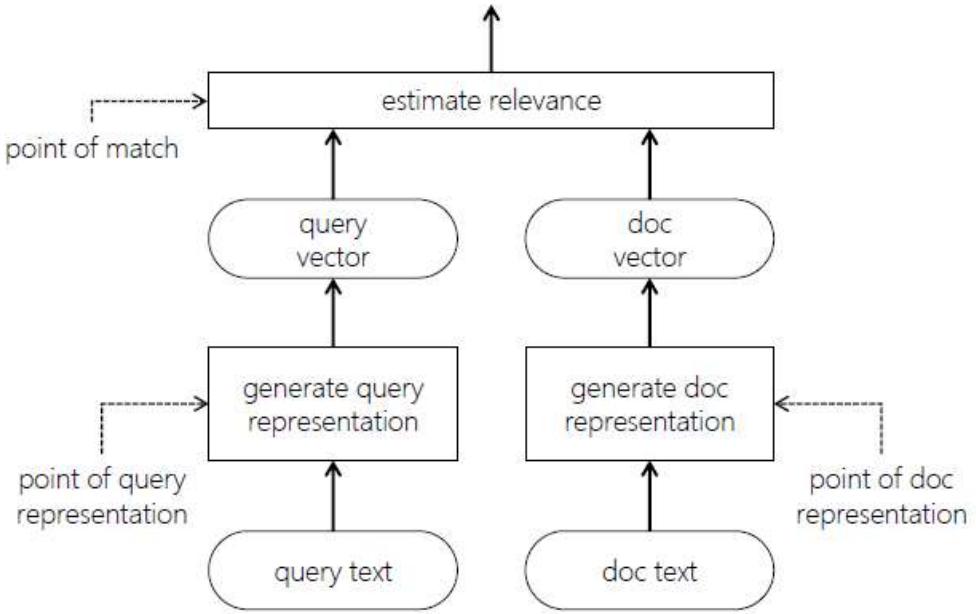
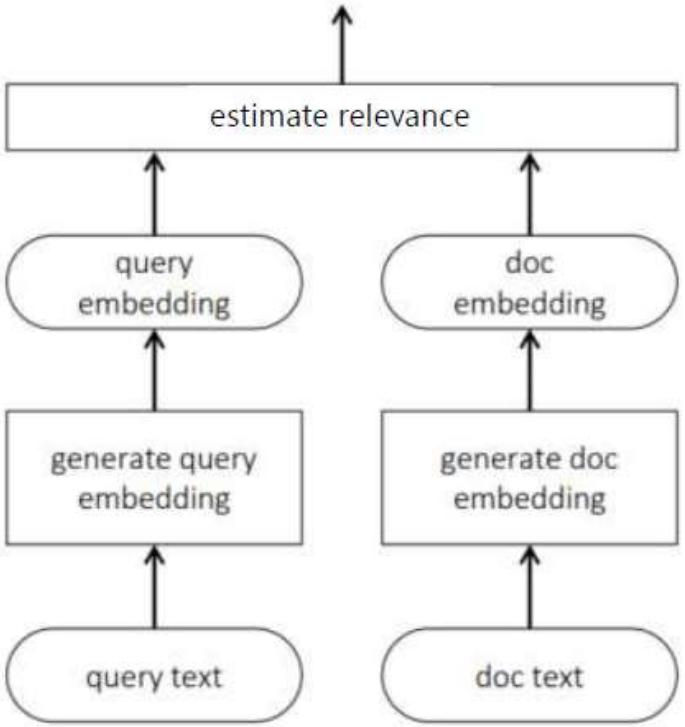
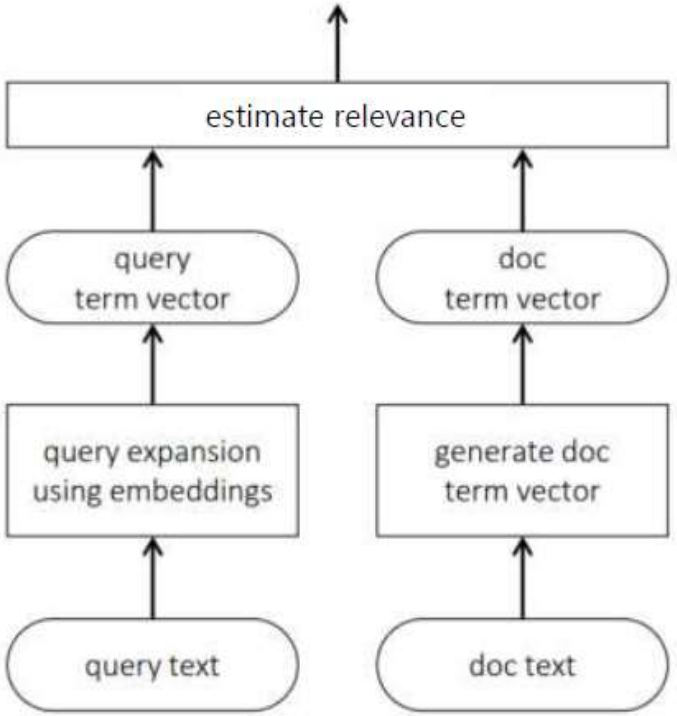


Figure 2.3: Document ranking typically involves a query and a document representation steps, followed by a matching stage. Neural models can be useful either for generating good representations or in estimating relevance, or both.

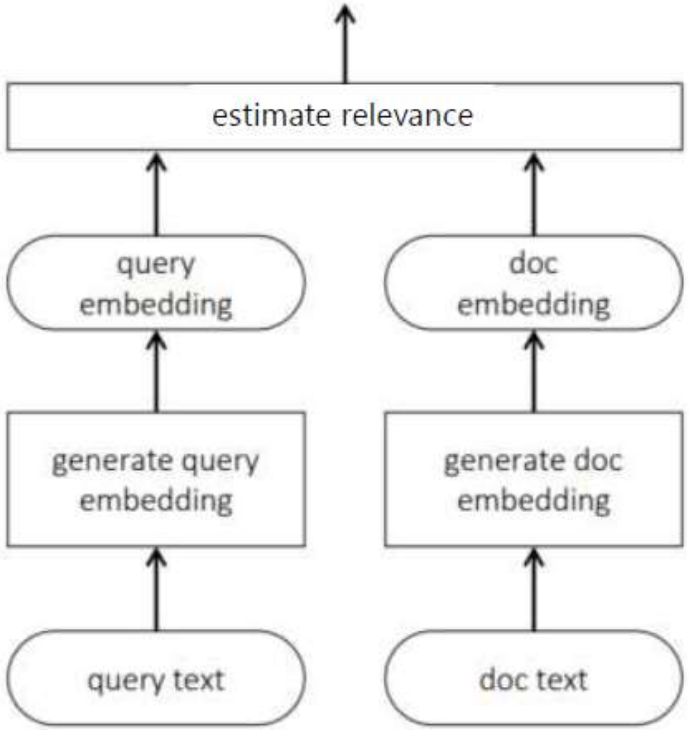
POPULAR APPROACHES TO INCORPORATING TERM EMBEDDINGS FOR MATCHING



Compare query and document directly in the embedding space



Use embeddings to generate suitable query expansions



Compare query and document directly in the embedding space

E.g.,

Generalized Language Model [Ganguly et al., 2015]

Neural Translation Language Model [Zuccon et al., 2015]

Average term embeddings [Le and Mikolov, 2014, Nalisnick et al., 2016, Zamani and Croft, 2016, and others]

Word mover's distance [Kusner et al., 2015, Guo et al., 2016]

GENERALIZED LANGUAGE MODEL

Traditional language modeling based IR approach may estimate q-d relevance as follows,

$$
\begin{aligned}
p(d|q) &= \frac{p(q|d).p(d)}{\sum_{\bar{d}\in D} p(q|\bar{d}).p(\bar{d})} \\
&\propto p(q|d).p(d) \\
&= p(q|d) \qquad \text{, assuming p(d) is uniform} \\
&= \prod_{t_q \in q} p(t_q|d)
\end{aligned}
$$

where, $p(t_q|d)$ is the probability of generating term t_q from document d

GENERALIZED LANGUAGE MODEL

Traditional language modeling based IR approach may estimate q-d relevance as follows,

$$
\begin{aligned}
p(d|q) &= \prod_{t_q \in q} p(t_q|d) \\
&= \prod_{t_q \in q} \left(\lambda \hat{p}(t_q|d) + (1-\lambda)\hat{p}(t_q|D) \right) \\
&= \prod_{t_q \in q} \left(\lambda \frac{tf(t_q,d)}{|d|} + (1-\lambda) \frac{\sum_{\bar{d} \in D} tf(t_q,\bar{d})}{\sum_{\bar{d} \in D} |\bar{d}|} \right)
\end{aligned}
$$

$\hat{p}(t_q|d)$ and $\hat{p}(t_q|D)$ are the probabilities of randomly sampling term t_q from document d and the full collection D, respectively

$\hat{p}(t_q|D)$ has a smoothing effect on the $p(t_q|d)$ estimation

GENERALIZED LANGUAGE MODEL

GLM includes additional smoothing based on term similarity in the embedding space

$$p(d|q) = \prod_{t_q \in q} \left(\lambda \frac{tf(t_q, d)}{|d|} + (1 - \lambda) \frac{\sum_{\bar{d} \in D} tf(t_q, \bar{d})}{\sum_{\bar{d} \in D} |\bar{d}|} \right)$$

Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth JF Jones. Word embedding based generalized language model for information retrieval. In SIGIR, 2015.

GENERALIZED LANGUAGE MODEL

GLM includes additional smoothing based on term similarity in the embedding space

$$p(d|q) = \prod_{t_q \in q} \left(\lambda \frac{tf(t_q, d)}{|d|} + (1 - \lambda) \frac{\sum_{\bar{d} \in D} tf(t_q, \bar{d})}{\sum_{\bar{d} \in D} |\bar{d}|} \right)$$

Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth JF Jones. Word embedding based generalized language model for information retrieval. In SIGIR, 2015.

GENERALIZED LANGUAGE MODEL

GLM includes additional smoothing based on term similarity in the embedding space

$$p(d|q) = \prod_{t_q \in q} \left(\lambda \frac{tf(t_q, d)}{|d|} + \alpha \frac{\sum_{t_d \in d} (sim(\vec{v}_{t_q}, \vec{v}_{t_d}) \cdot tf(t_d, d))}{\sum_{t_{d_1} \in d} \sum_{t_{d_2} \in d} sim(\vec{v}_{t_{d_1}}, \vec{v}_{t_{d_2}}) \cdot |d|^2} + \beta \frac{\sum_{\bar{t} \in N_t} (sim(\vec{v}_{t_q}, \vec{v}_{\bar{t}}) \cdot \sum_{\bar{d} \in D} tf(\bar{t}, \bar{d}))}{\sum_{t_{d_1} \in N_t} \sum_{t_{d_2} \in N_t} sim(\vec{v}_{t_{d_1}}, \vec{v}_{t_{d_2}}) \cdot \sum_{\bar{d} \in D} |\bar{d}| \cdot |N_t|} + (1 - \alpha - \beta - \lambda) \frac{\sum_{\bar{d} \in D} tf(t_q, \bar{d})}{\sum_{\bar{d} \in D} |\bar{d}|} \right)$$

Probability of generating the term from the document based on similarity in the embedding space

Probability of generating the term from the full collection based on similarity in the embedding space

Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth JF Jones. Word embedding based generalized language model for information retrieval. In SIGIR, 2015.

NEURAL TRANSLATION LANGUAGE MODEL

TLM estimates $p(t_q|t_d)$ from q-d paired data similar to statistical machine translation

NTLM uses term-term similarity in the embedding space to estimate $p(t_q|t_d)$

NTLM - skipgram			
w = insider		w = trading	
u	$p(w\|u)$	u	$p(w\|u)$
insider	0.169	trading	0.164
fraud	0.102	traders	0.103
drexel	0.099	futures	0.099
securities	0.096	stock	0.097
racketeering	0.093	exchange	0.094
bribery	0.091	market	0.093

Translation Language Model:

$$p(t_q|d) = \sum_{t_d \in d} p(t_q|t_d) \cdot p(t_d|d)$$

Neural Translation Language Model:

$$p(t_q|t_d) = \frac{cos(\vec{v}_{t_q}, \vec{v}_{t_d})}{\sum_{t \in T} cos(\vec{v}_t, \vec{v}_{t_d})}$$

Guido Zuccon, Bevan Koopman, Peter Bruza, and Leif Azzopardi. Integrating and evaluating neural word embeddings in information retrieval. In ADCS, 2015.

AVERAGE TERM EMBEDDINGS

Q-D relevance estimated by computing cosine similarity between centroid of q and d term embeddings

$$sim(q,d) = cos(\vec{v}_q, \vec{v}_d) = \frac{\vec{v}_q^{\intercal}\vec{v}_d}{\|\vec{v}_q\|\|\vec{v}_d\|}$$

$$\text{where,} \quad \vec{v}_q = \frac{1}{|q|}\sum_{t_q \in q}\frac{\vec{v}_{t_q}}{\|\vec{v}_{t_q}\|}$$

$$\vec{v}_d = \frac{1}{|d|}\sum_{t_d \in d}\frac{\vec{v}_{t_d}}{\|\vec{v}_{t_d}\|}$$

Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. Improving document ranking with dual word embeddings. In WWW, 2016.
Bhaskar Mitra, Eric Nalisnick, Nick Craswell, and Rich Caruana. A dual embedding space model for document ranking. arXiv preprint arXiv:1602.01137, 2016.

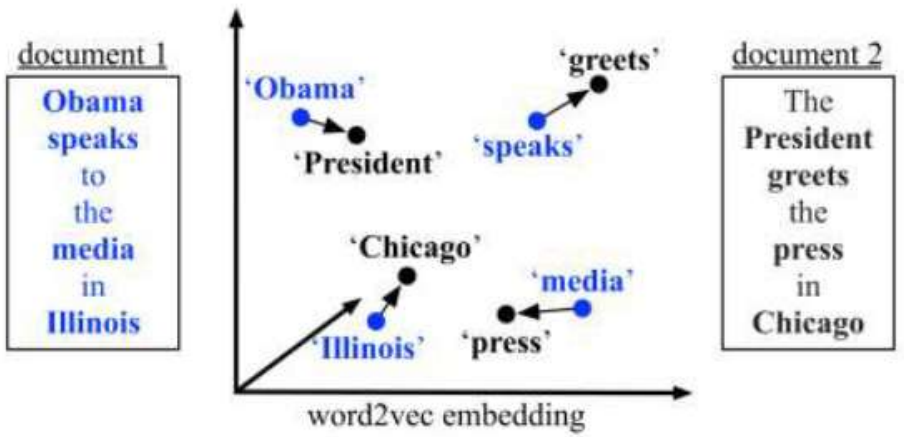
WORD MOVER'S DISTANCE

Based on the Earth Mover's Distance (EMD)
[Rubner et al., 1998]

Originally proposed by Wan et al. [2005, 2007], but used WordNet and topic categories

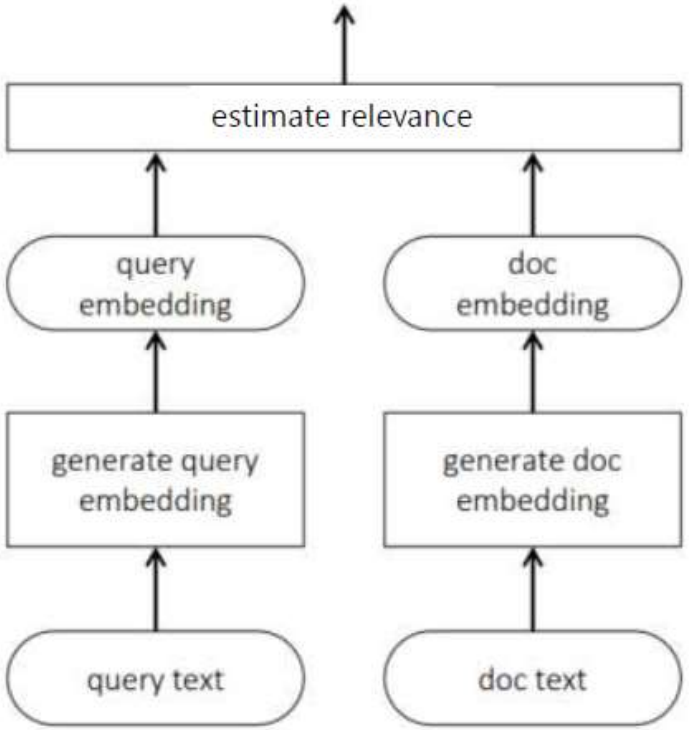
Kusner et al. [2015] incorporated term embeddings

Adapted for q-d matching by Guo et al. [2016]

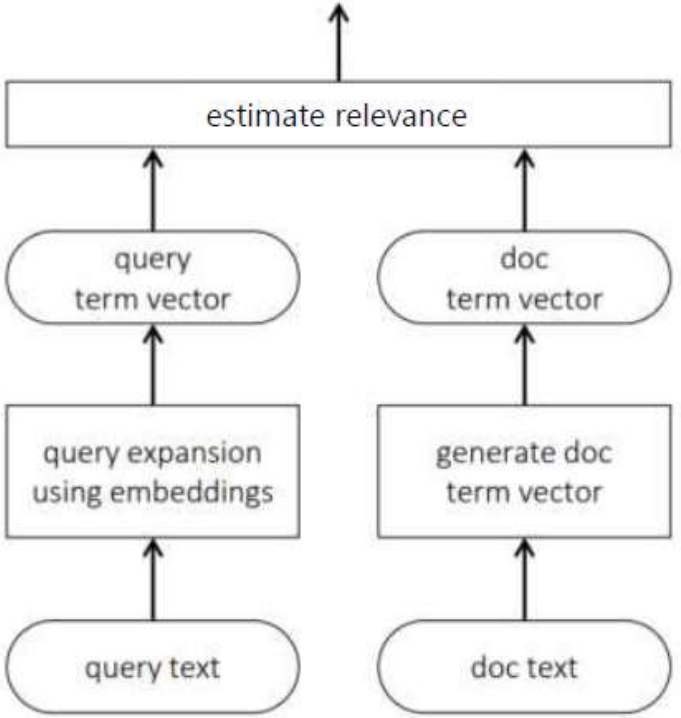


Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. A metric for distributions with applications to image databases. In CV, 1998.
Xiaojun Wan and Yuxin Peng. The earth mover's distance as a semantic measure for document similarity. In CIKM, 2005.
Xiaojun Wan. A novel document similarity measure based on earth mover's distance. Information Sciences, 2007.
Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In ICML, 2015.
Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. Semantic matching by non-linear word transportation for information retrieval. In CIKM, 2016.

POPULAR APPROACHES TO INCORPORATING TERM EMBEDDINGS FOR MATCHING



Compare query and document directly in the embedding space



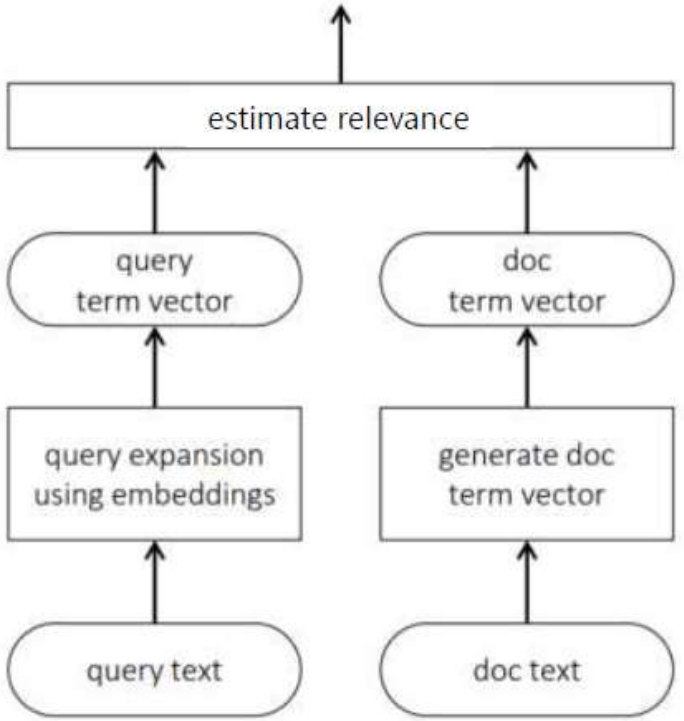
Use embeddings to generate suitable query expansions

QUERY EXPANSION USING TERM EMBEDDINGS

Find good expansion terms based on nearness in the embedding space

$$score(t_c, q) = \frac{1}{|q|} \sum_{t_q \in q} cos(\vec{v}_{t_c}, \vec{v}_{t_q})$$

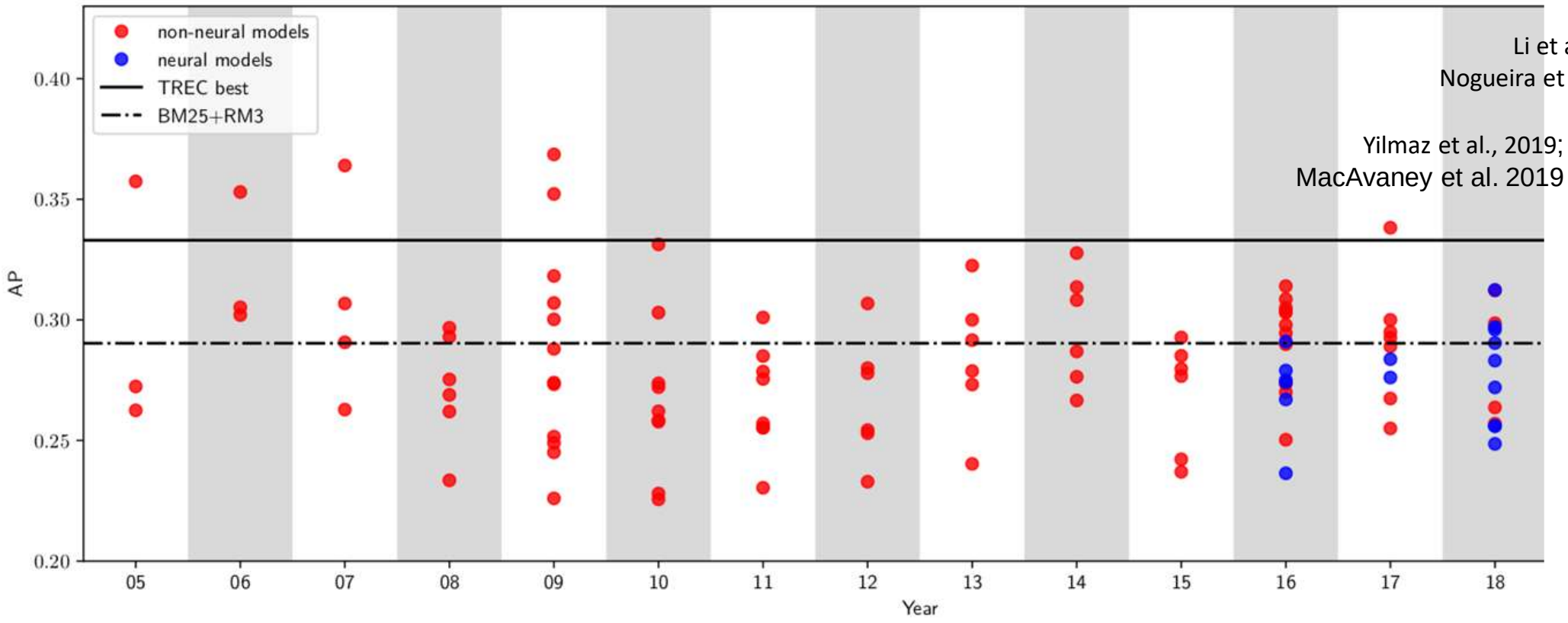
Better retrieval performance when combined with pseudo-relevance feedback (PRF) [Zamani and Croft, 2016] and if we learn query specific term embeddings [Diaz et al., 2016]



Use embeddings to generate suitable query expansions

Fernando Diaz, Bhaskar Mitra, and Nick Craswell. Query expansion with locally-trained word embeddings. In ACL, 2016.
Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. Using word embeddings for automatic query expansion. arXiv preprint arXiv:1606.07608, 2016.
Hamed Zamani and W Bruce Croft. Embedding-based query language models. In ICTIR, 2016.

Progress in Information Retrieval - Robust04



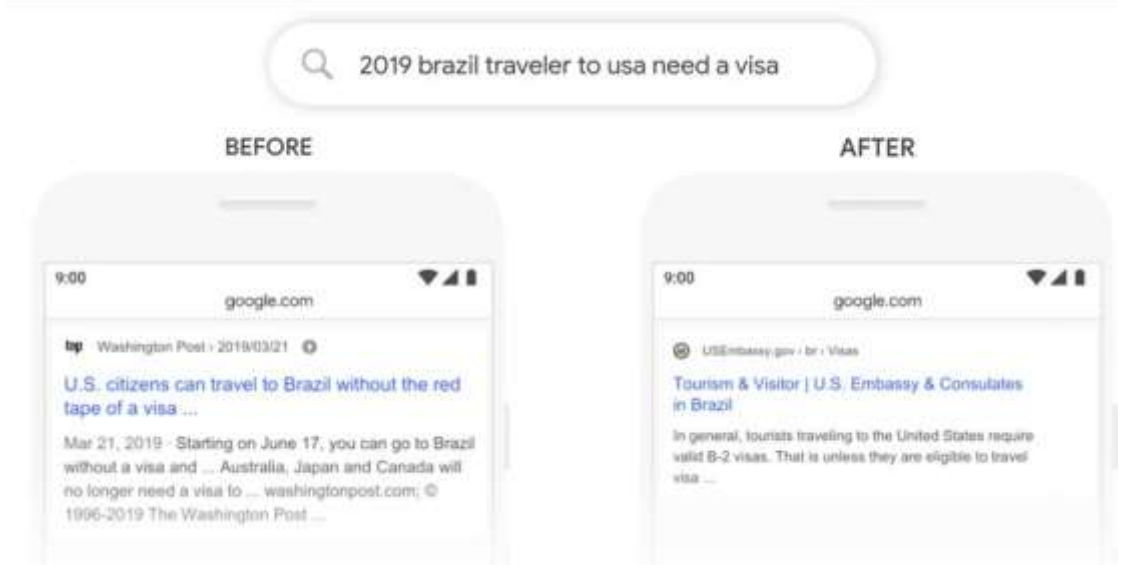
Li et al., 2020;
Nogueira et al., 2020

Yilmaz et al., 2019;
MacAvaney et al. 2019

Source: [Yang et al., (2019)]

Adoption by Commercial Search Engines

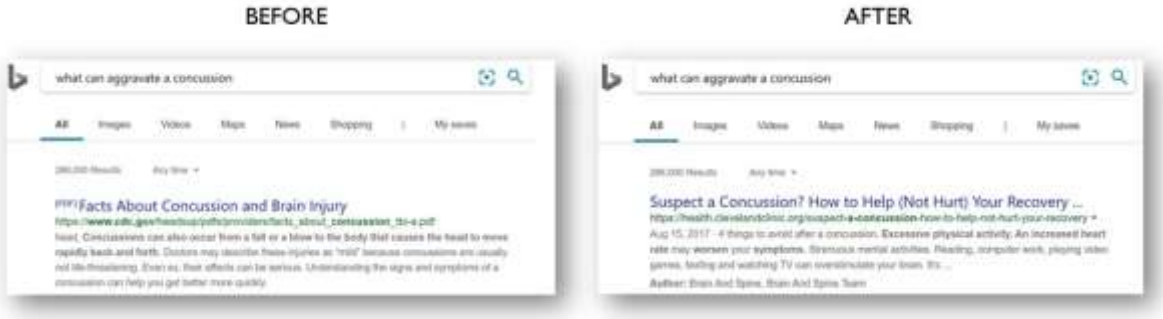
Google Search



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



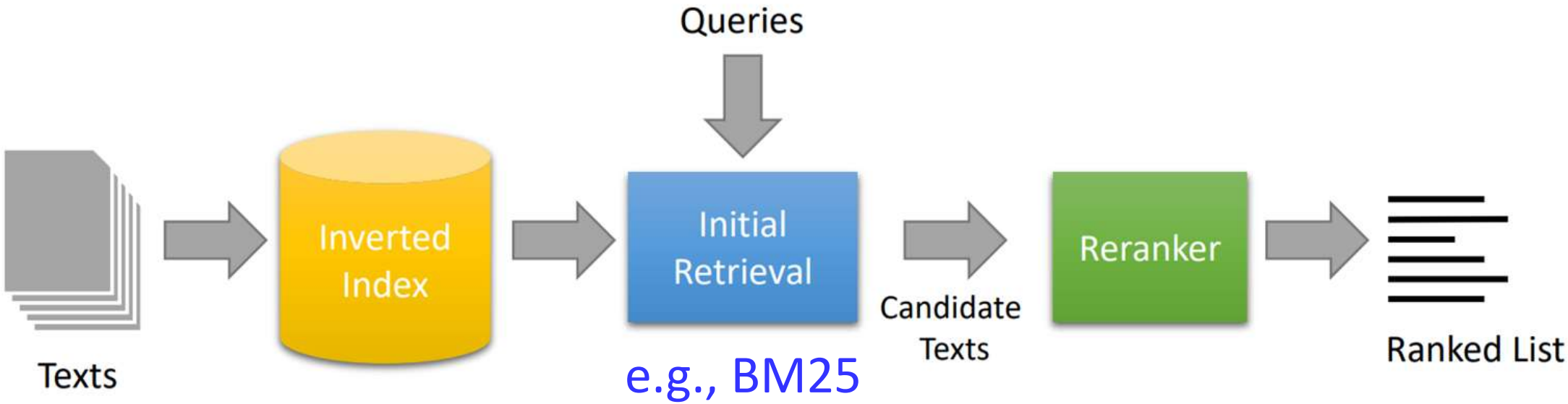
Starting from April of this year (2019), we used large transformer models to deliver the largest quality improvements to our Bing customers in the past year.

source

BERT
for Relevance Classification

(aka monoBERT)

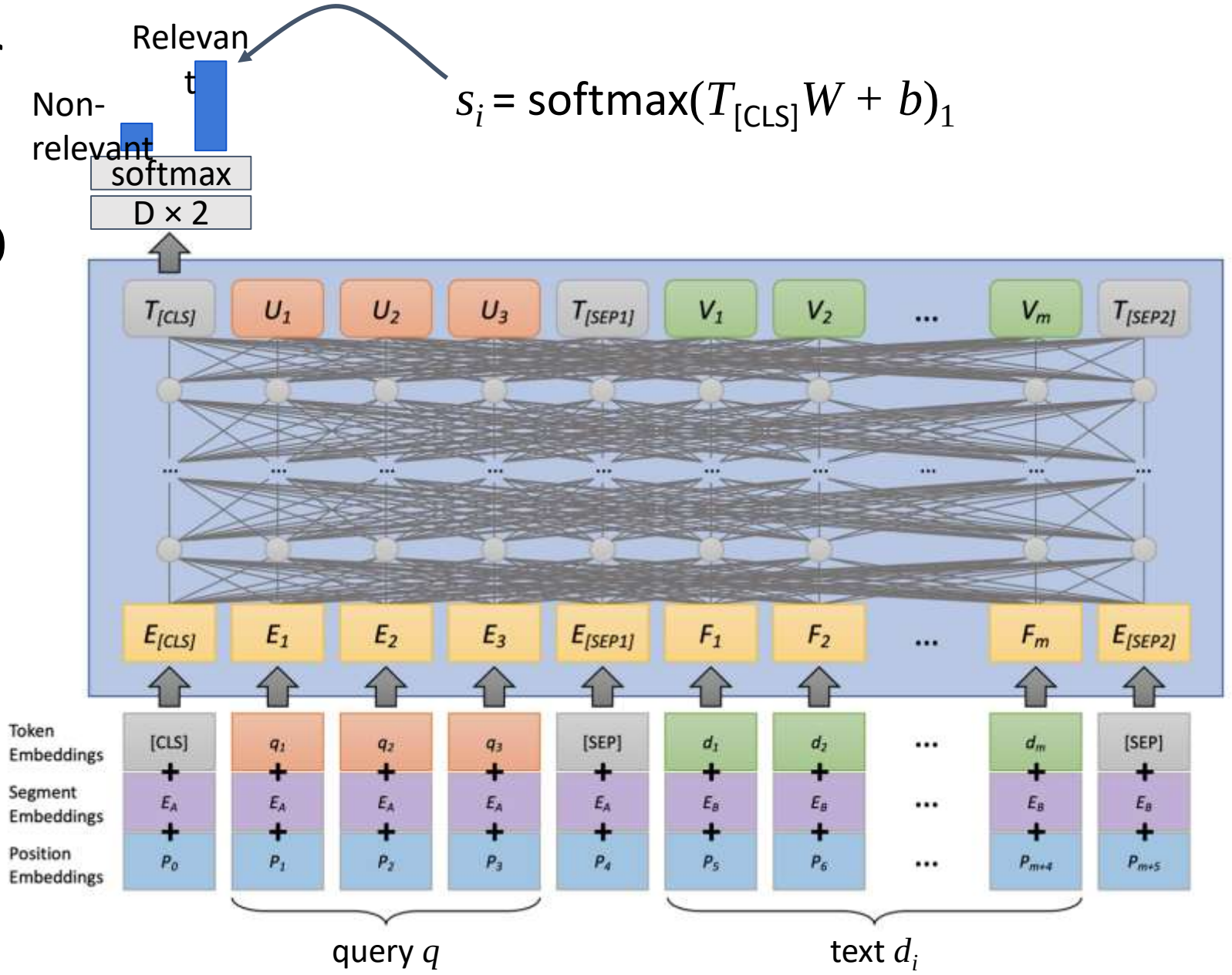
Retrieve-and-Rerank Architecture



monoBERT: BERT reranker

We want:

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



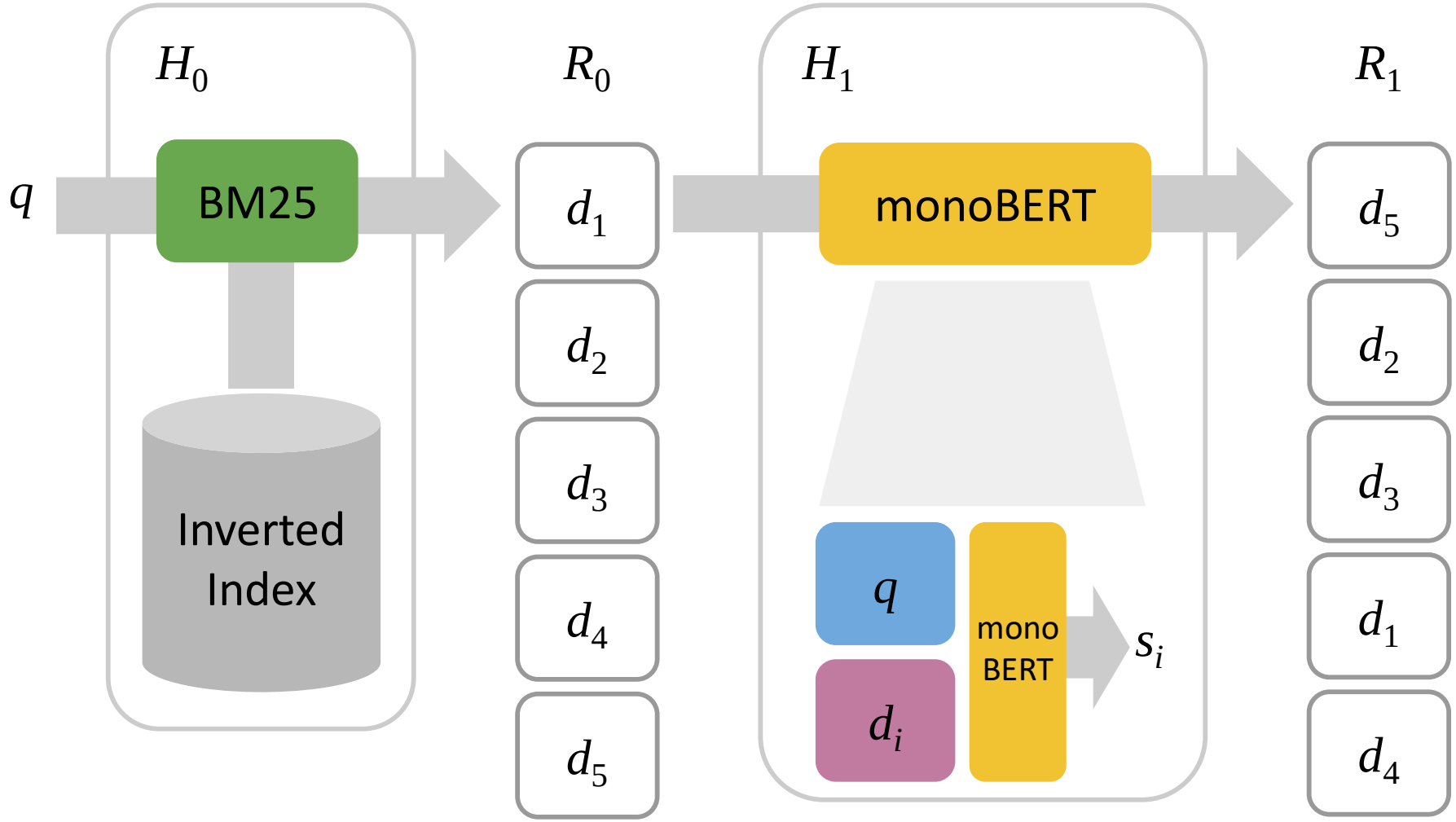
Training monoBERT

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

Humans

BM25

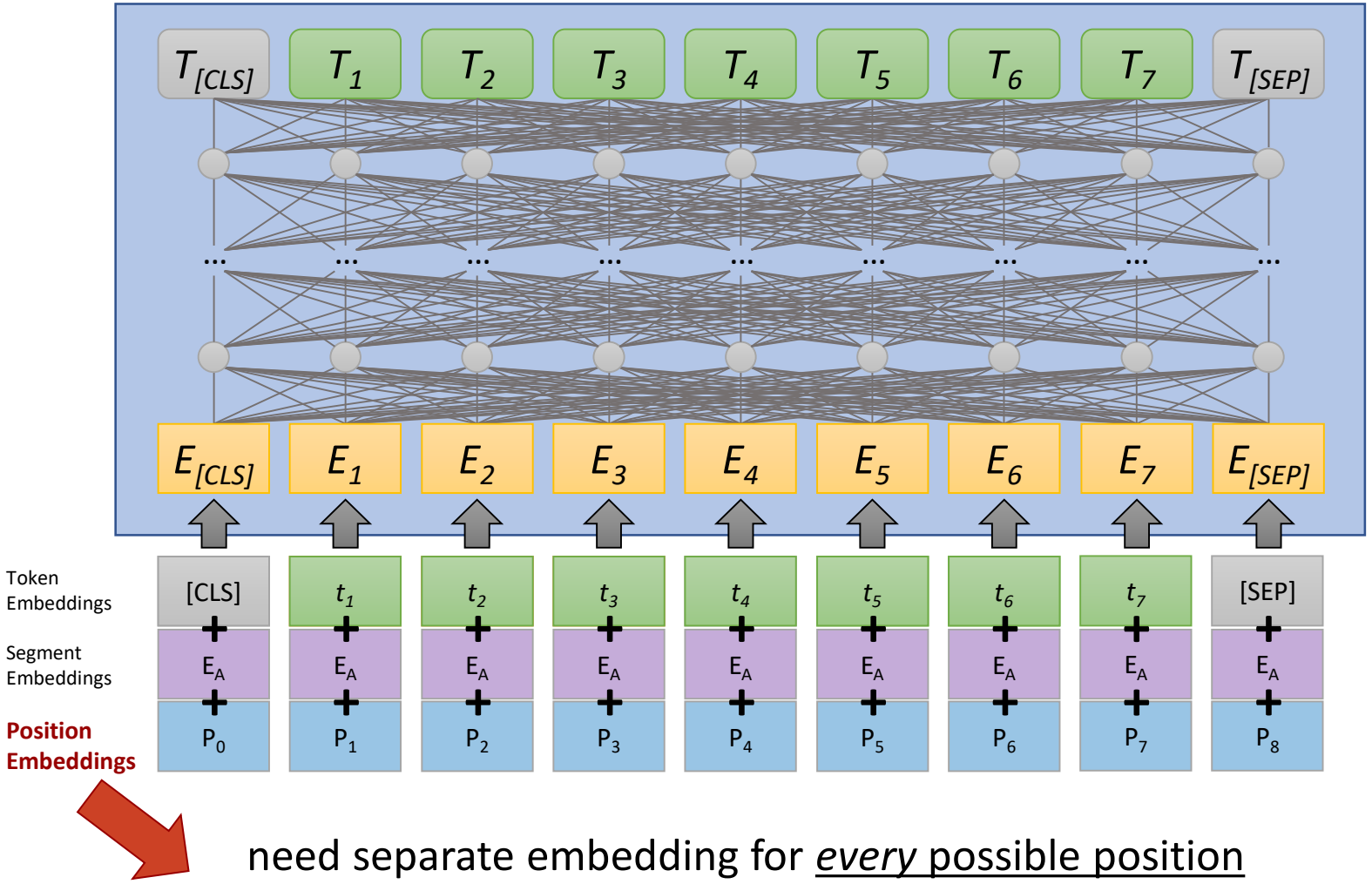
Once monoBERT is trained…



TREC 2019 - Deep Learning Track - Passage

	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

BERT's Limitation



need separate embedding for *every* possible position

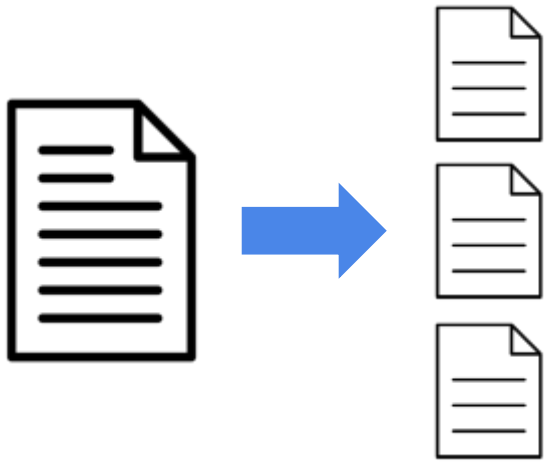
➔ restricted to indices 0-511

Cannot input entire documents

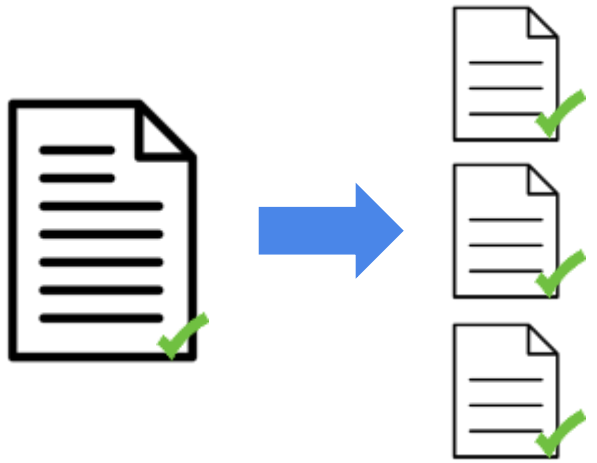
- what do we input?
- & how do we label it?

From Passages to Documents

Handling Length Limitation: Training

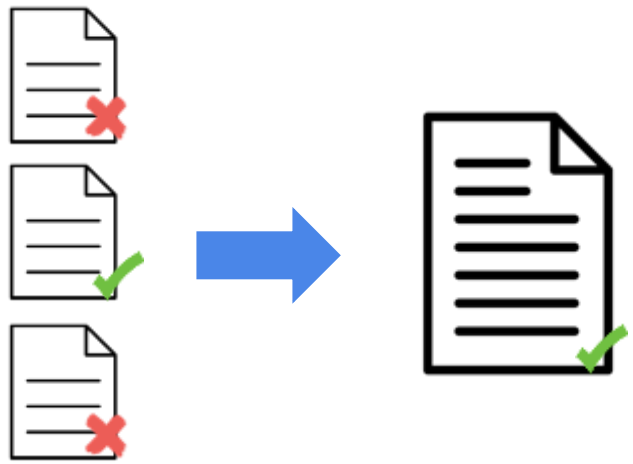


Chunk documents



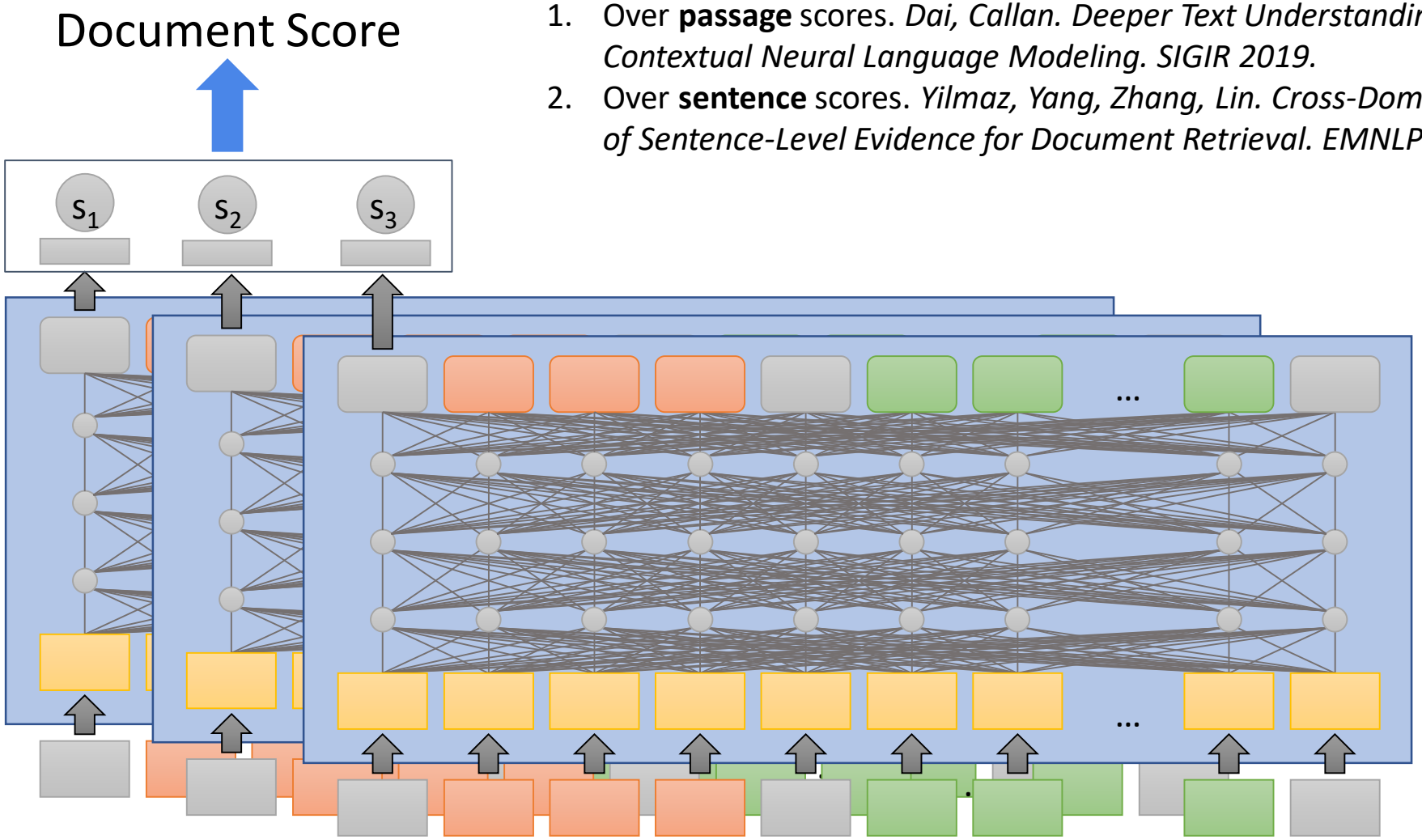
Transfer labels
(approximation)

Handling Length Limitation: Inference



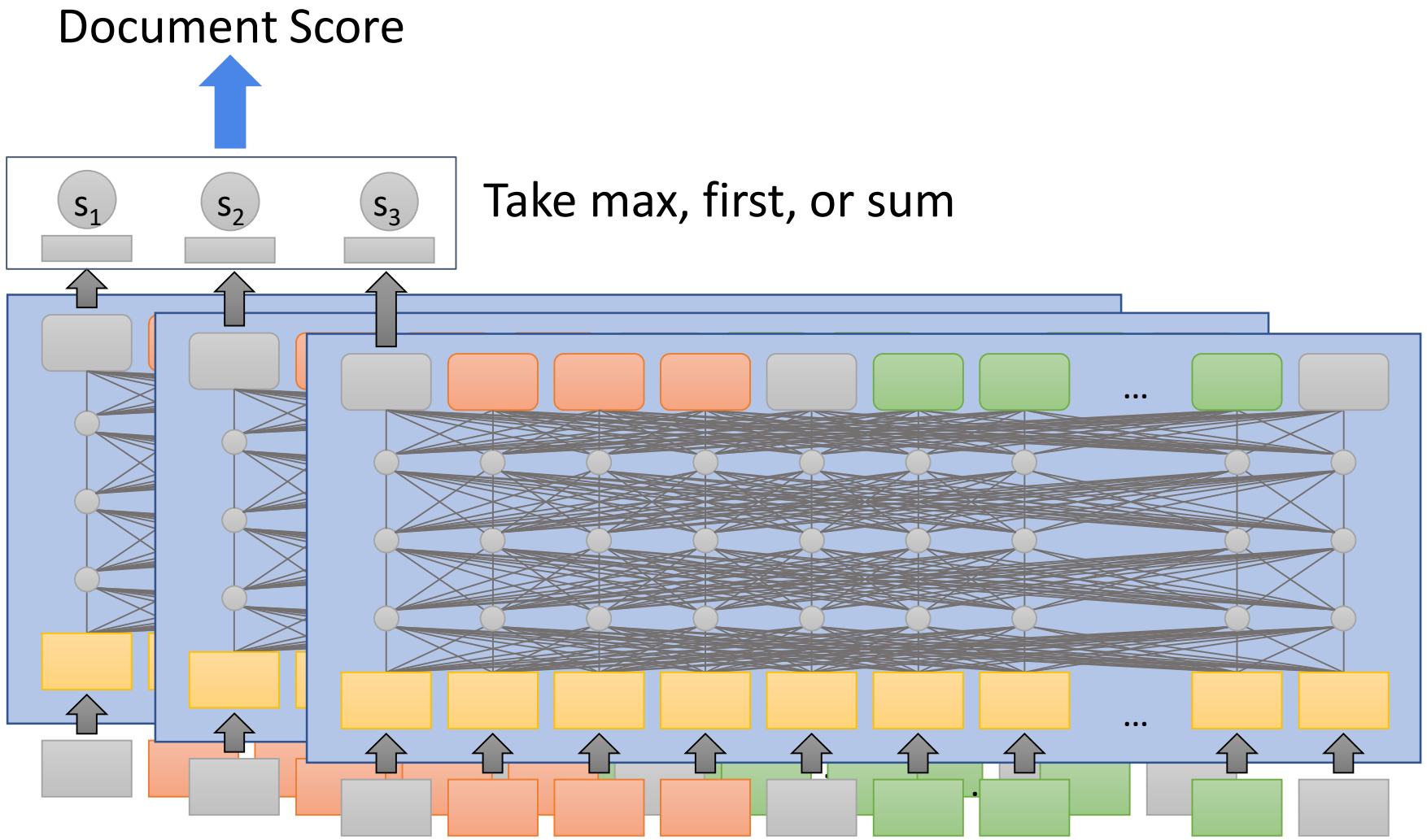
Aggregate Evidence

Approach #1: Score Aggregation



1. Over **passage** scores. *Dai, Callan. Deeper Text Understanding for IR with Contextual Neural Language Modeling. SIGIR 2019.*
2. Over **sentence** scores. *Yilmaz, Yang, Zhang, Lin. Cross-Domain Modeling of Sentence-Level Evidence for Document Retrieval. EMNLP '19.*

Over Passage Scores: BERT-MaxP, FirstP, SumP



Dai, Callan. Deeper Text Understanding for IR with Contextual Neural Language Modeling. SIGIR 2019.

Over Passage Scores: Results

Model		Robust04 nDCG@20 Title	Robust04 nDCG@20 Description
(1)	BOW	0.417	0.409
(2)	SDM	0.427	0.427
(3)	LTR	0.427	0.441
(4a)	BERT–FirstP	0.444†	0.491†
(4b)	BERT–MaxP	**0.469**†	**0.529**†
(4c)	BERT–SumP	0.467†	0.524†

Dai, Callan. Deeper Text Understanding for IR with Contextual Neural Language Modeling. SIGIR 2019.

Over Sentence Scores: Birch

$$s_f \triangleq \alpha \cdot s_d + (1 - \alpha) \cdot \sum_{i=1}^{n} w_i \cdot s_i$$

First-stage retrieval score

Sentence scores

- Trained on sentence-level judgments like tweets
- Interpolation weights are tuned on target dataset

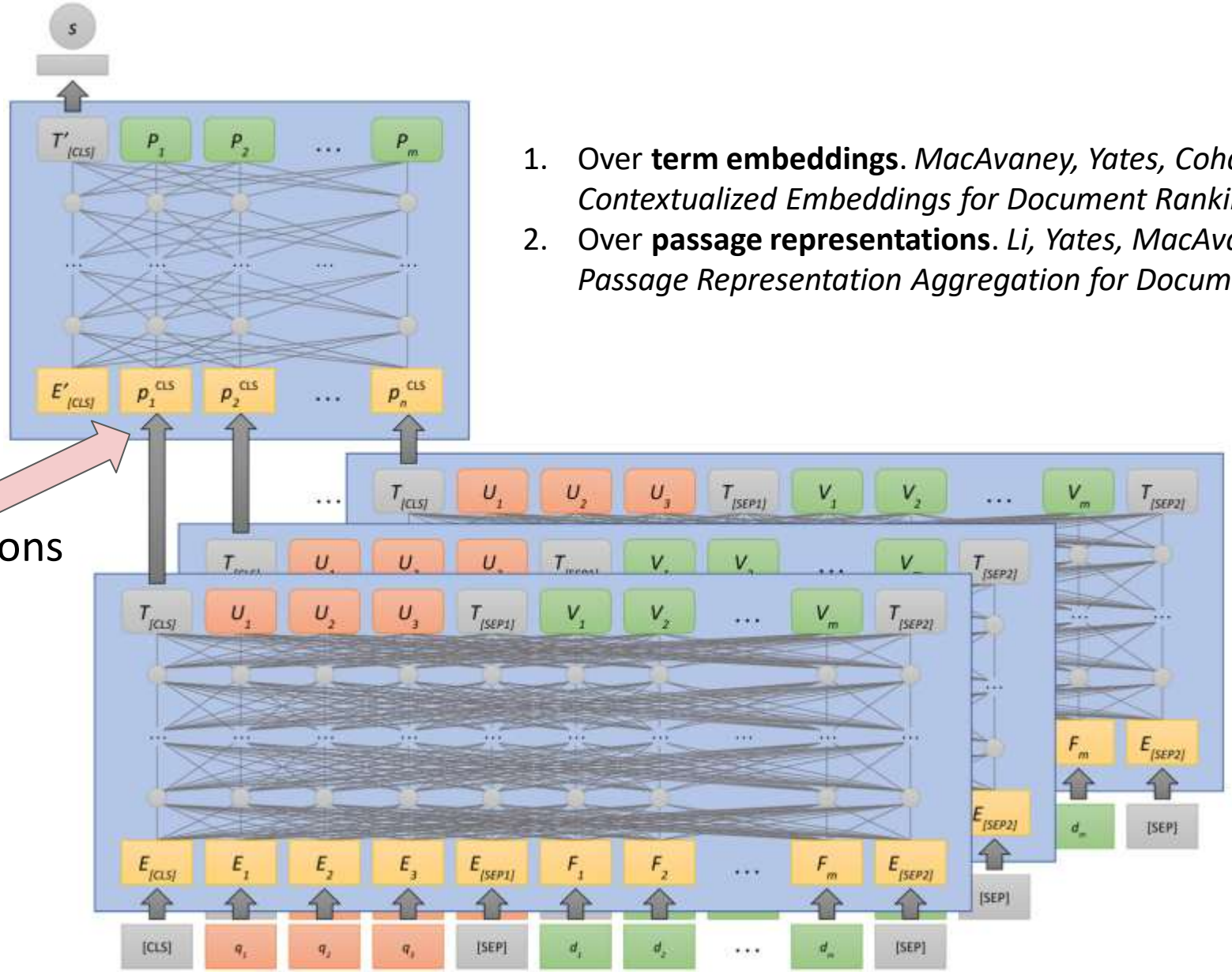
Yilmaz, Yang, Zhang, Lin. Cross-Domain Modeling of Sentence-Level Evidence for Document Retrieval. EMNLP '19.

Over Sentence Scores: Results

Method		Robust04	
		MAP	nDCG@20
(1)	BM25 + RM3	0.2903	0.4407
(2a)	1S: BERT(MB)	0.3408†	0.4900†
(2b)	2S: BERT(MB)	0.3435†	0.4964†
(2c)	3S: BERT(MB)	0.3434†	0.4998†

Yilmaz, Yang, Zhang, Lin. Cross-Domain Modeling of Sentence-Level Evidence for Document Retrieval. EMNLP '19.

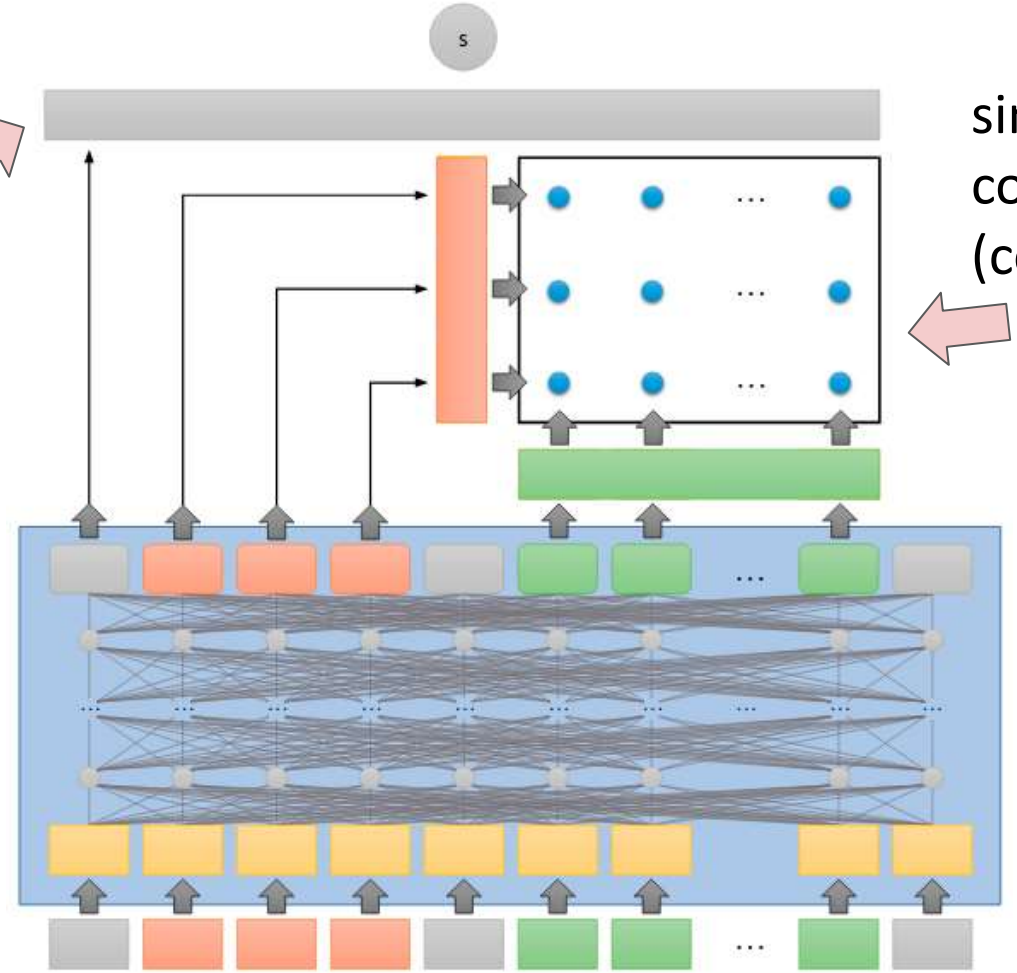
Approach #2: Representation Aggregation



1. Over **term embeddings**. *MacAvaney, Yates, Cohan, Goharian. CEDR: Contextualized Embeddings for Document Ranking. SIGIR 2019.*
2. Over **passage representations**. *Li, Yates, MacAvaney, He, Sun. PARADE: Passage Representation Aggregation for Document Reranking. arXiv 2020.*

Over Term Embeddings: CEDR

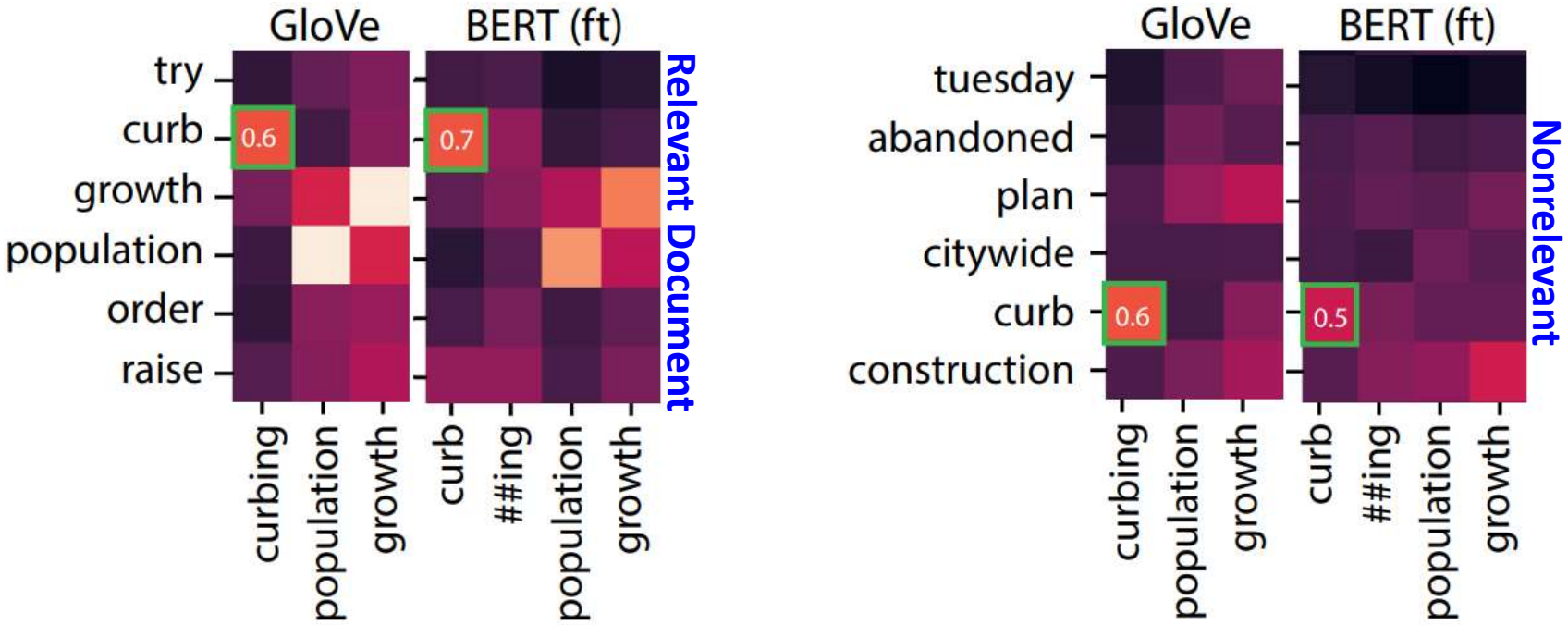
interaction-based pre-BERT model (PACRR, KNRM)



similarity matrix using contextualized embeddings (concatenate passages)

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

Over Term Embeddings: CEDR



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

Over Term Embeddings: Results

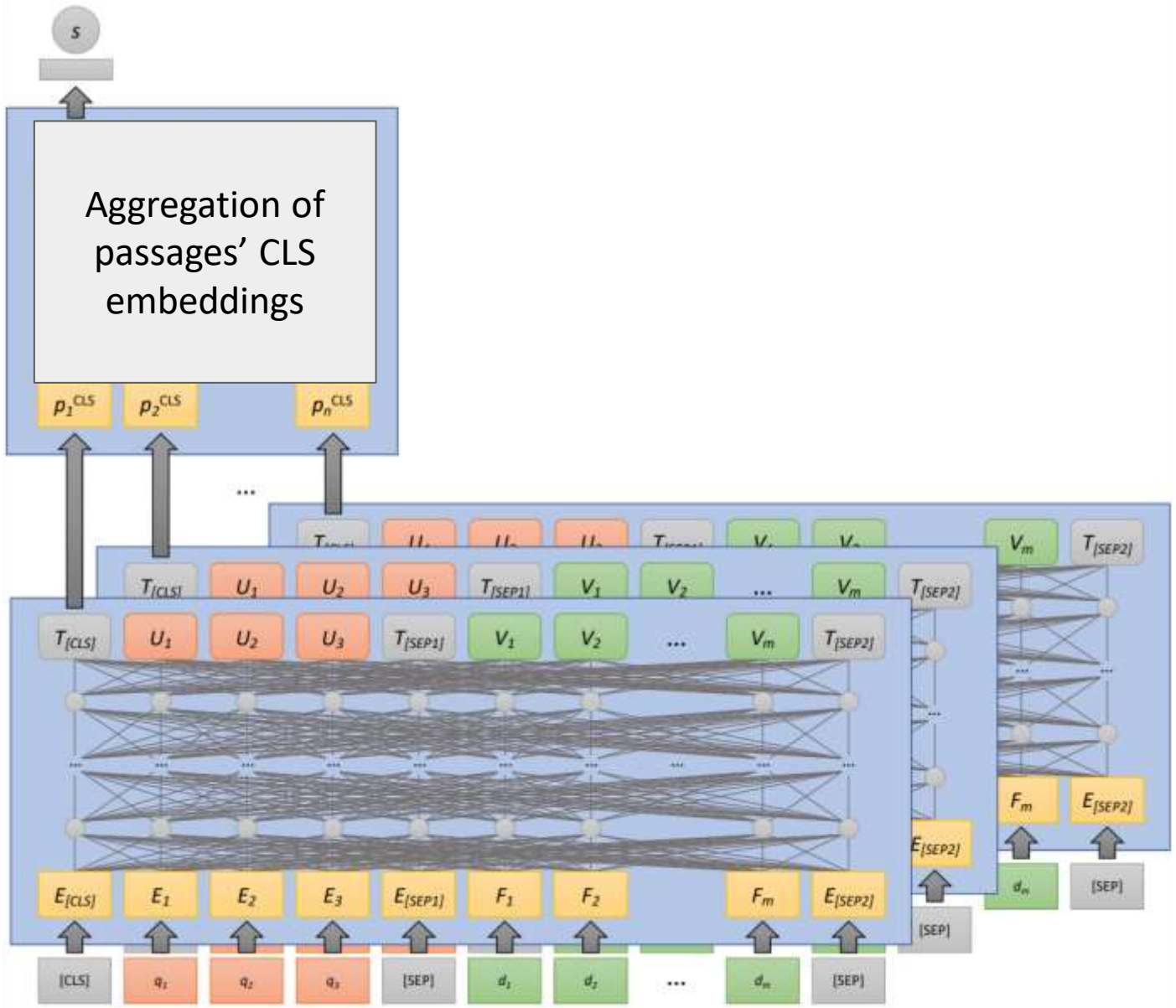
	Method	Input Representation	Robust04 nDCG@20
(1)	BM25	n/a	0.4140
(2)	Vanilla BERT	BERT (fine-tuned)	[B] 0.4541
(3a)	PACRR	GloVe	0.4043
(3b)	PACRR	BERT	0.4200
(3c)	PACRR	BERT (fine-tuned)	[BVG] 0.5135
(3d)	CEDR–PACRR	BERT (fine-tuned)	**[BVG] 0.5150**
(4a)	KNRM	GloVe	0.3871
(4b)	KNRM	BERT	[G] 0.4318
(4c)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

Over Passage Representations: PARADE

Aggregation approaches:
(increasing complexity)

- Average feature value
- Max feature value
- Attn-weighted average
- Two Transformer layers



Li, Yates, MacAvaney, He, Sun. PARADE: Passage Representation Aggregation for Document Reranking. arXiv 2020.

Over Passage Representations: Results

Method		Robust04 nDCG@20 Title	Robust04 nDCG@20 Description
(1)	BM25	0.4240	0.4058
(2)	BM25 + RM3	0.4407	0.4255
(3a)	Birch (MS)	0.4227	0.4053
(3b)	Birch (MS→MB)	0.5137	0.5069
(4)	BERT–MaxP (MS)	0.4931	0.5453
(5a)	PARADE $_{\text{Avg}}$	0.4917†	0.5324†‡
(5b)	PARADE $_{\text{Max}}$	0.5115†§	0.5487†‡
(5c)	PARADE $_{\text{Attn}}$	0.5134†§	0.5517†‡
(5d)	PARADE	**0.5252**†§	**0.5605**†‡§
(6)	PARADE (with BERT $_{\text{Large}}$)	0.5243	-

Li, Yates, MacAvaney, He, Sun. PARADE: Passage Representation Aggregation for Document Reranking. arXiv 2020.

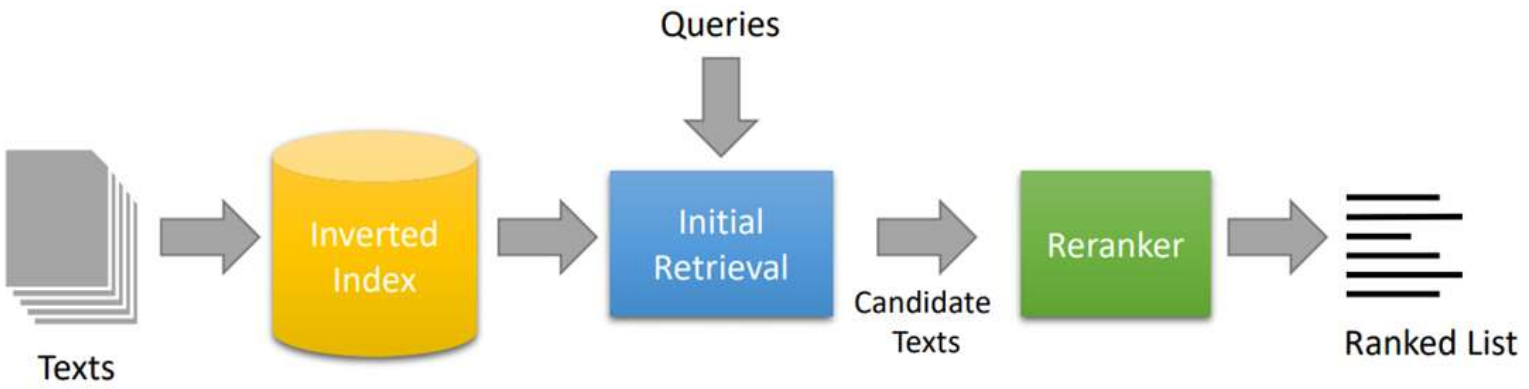
Multi-stage rerankers

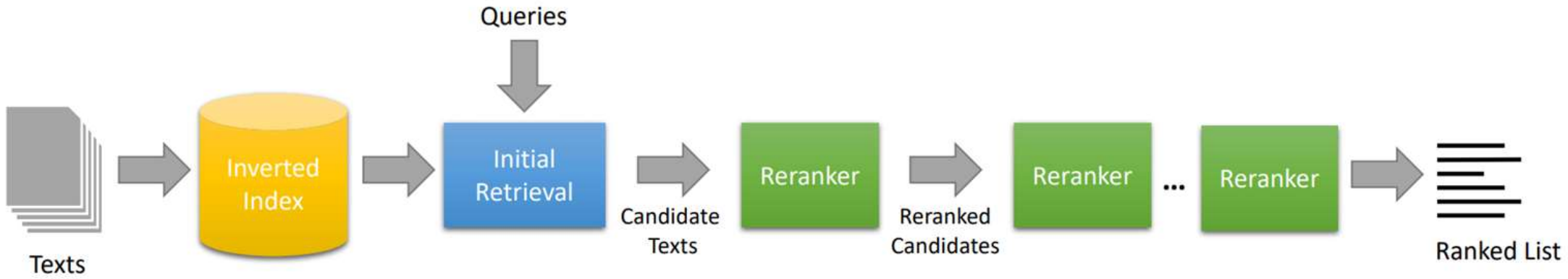
why multi-stage?
duoBERT

Multi-stage rerankers

why multi-stage?
duoBERT

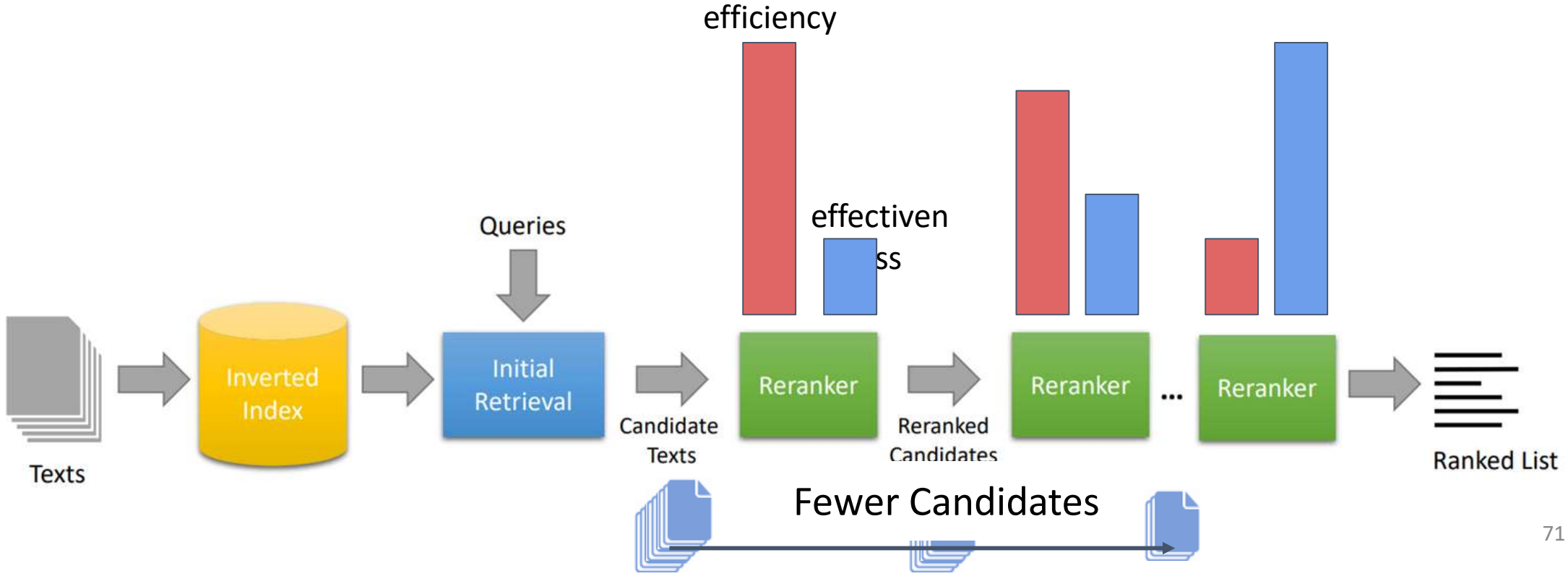
From Single to Multiple Rerankers





Why Multi-stage?

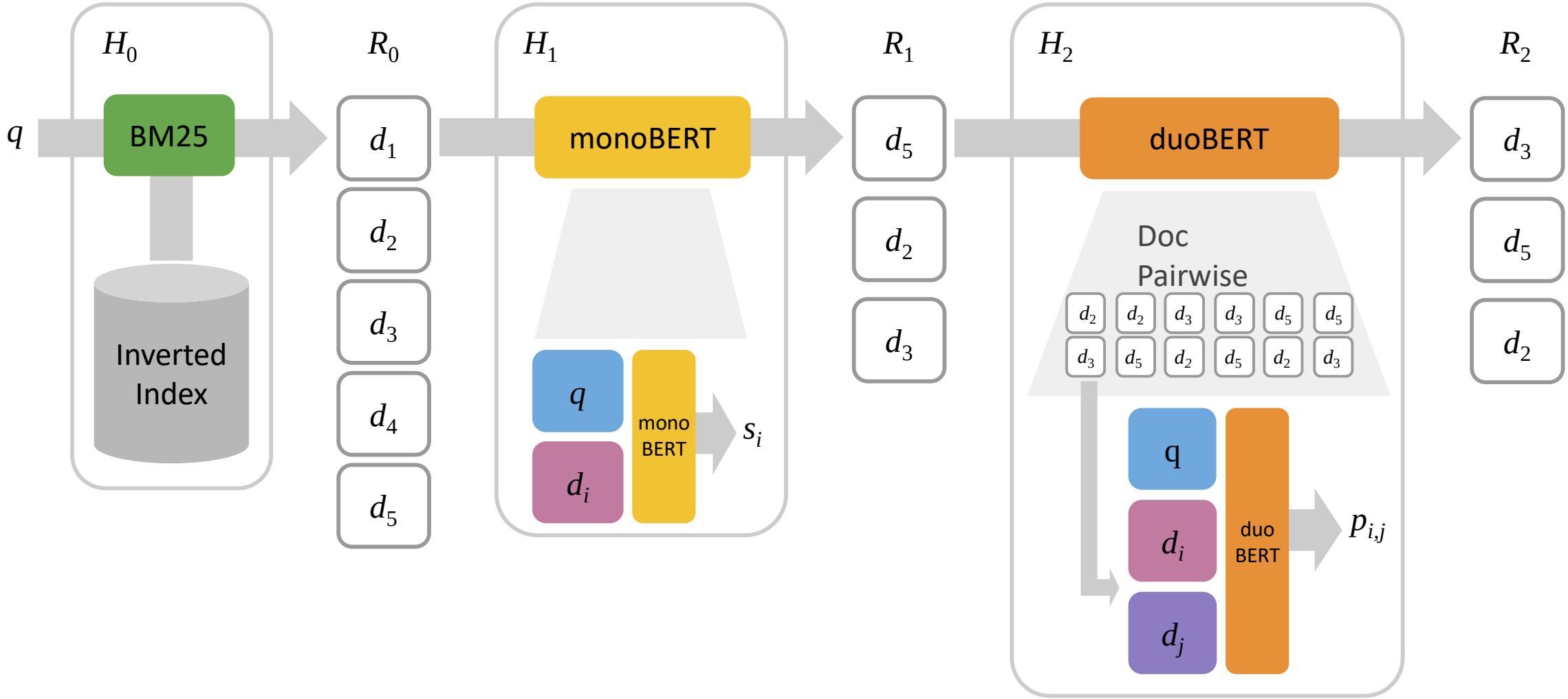
- Trade-off between effectiveness (quality of the ranked lists) and efficiency (retrieval latency)



Multi-stage Rerankers

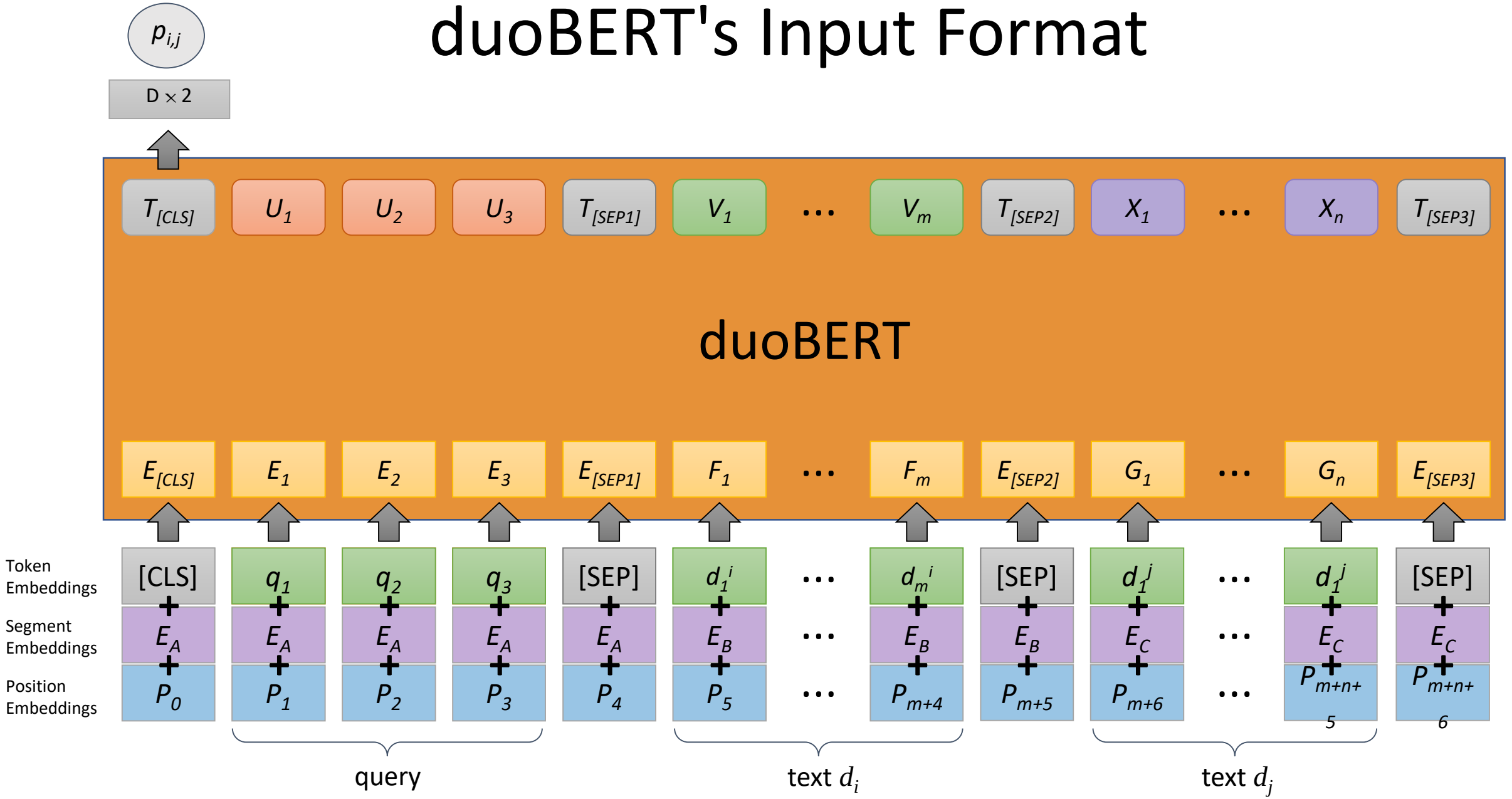
why multi-stage?
duoBERT

Multi-stage with duoBERT



Nogueira, Yang, Cho, Lin. Multi-stage document ranking with bert. 2019.

duoBERT's Input Format



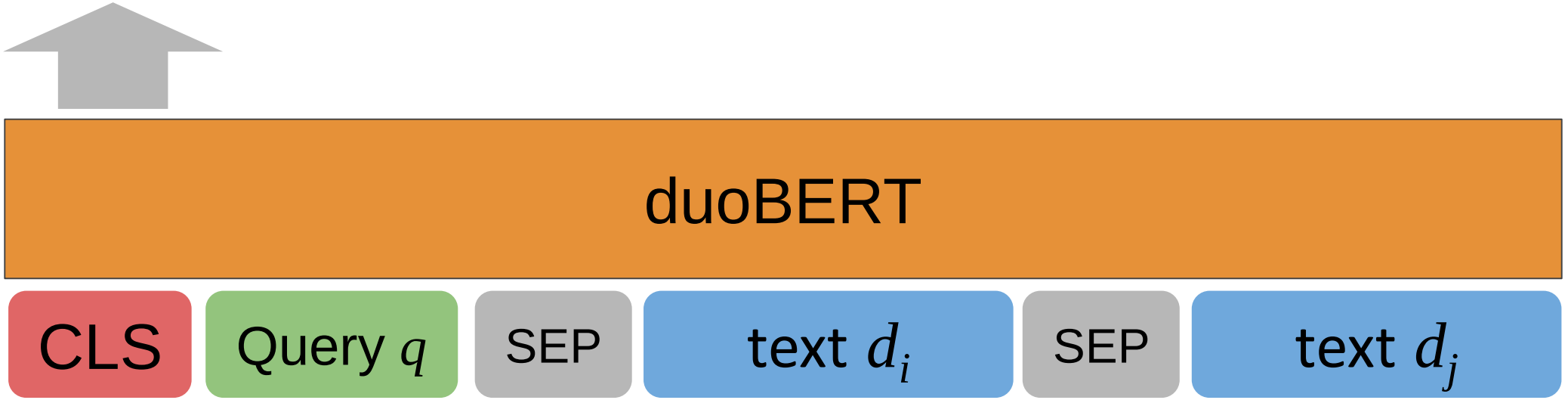
Training duoBERT

Is doc d_i more relevant than doc d_j to the query q?

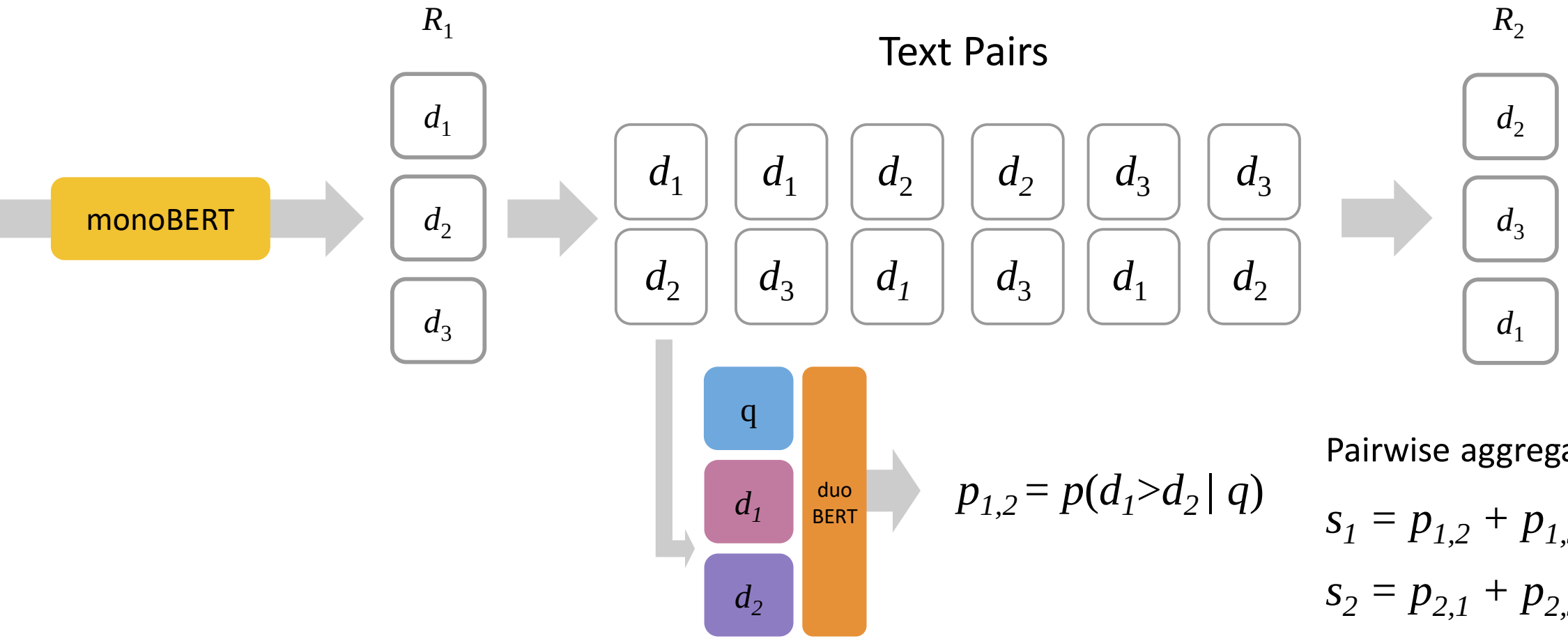
Loss:

$$L_{\text{duo}} = -\sum_{i \in J_{\text{pos}}, j \in J_{\text{neg}}} \log(p_{i,j}) - \sum_{i \in J_{\text{neg}}, j \in J_{\text{pos}}} \log(1 - p_{i,j})$$

$p_{i,j} = p(d_i > d_j \mid q)$



Inference with duoBERT



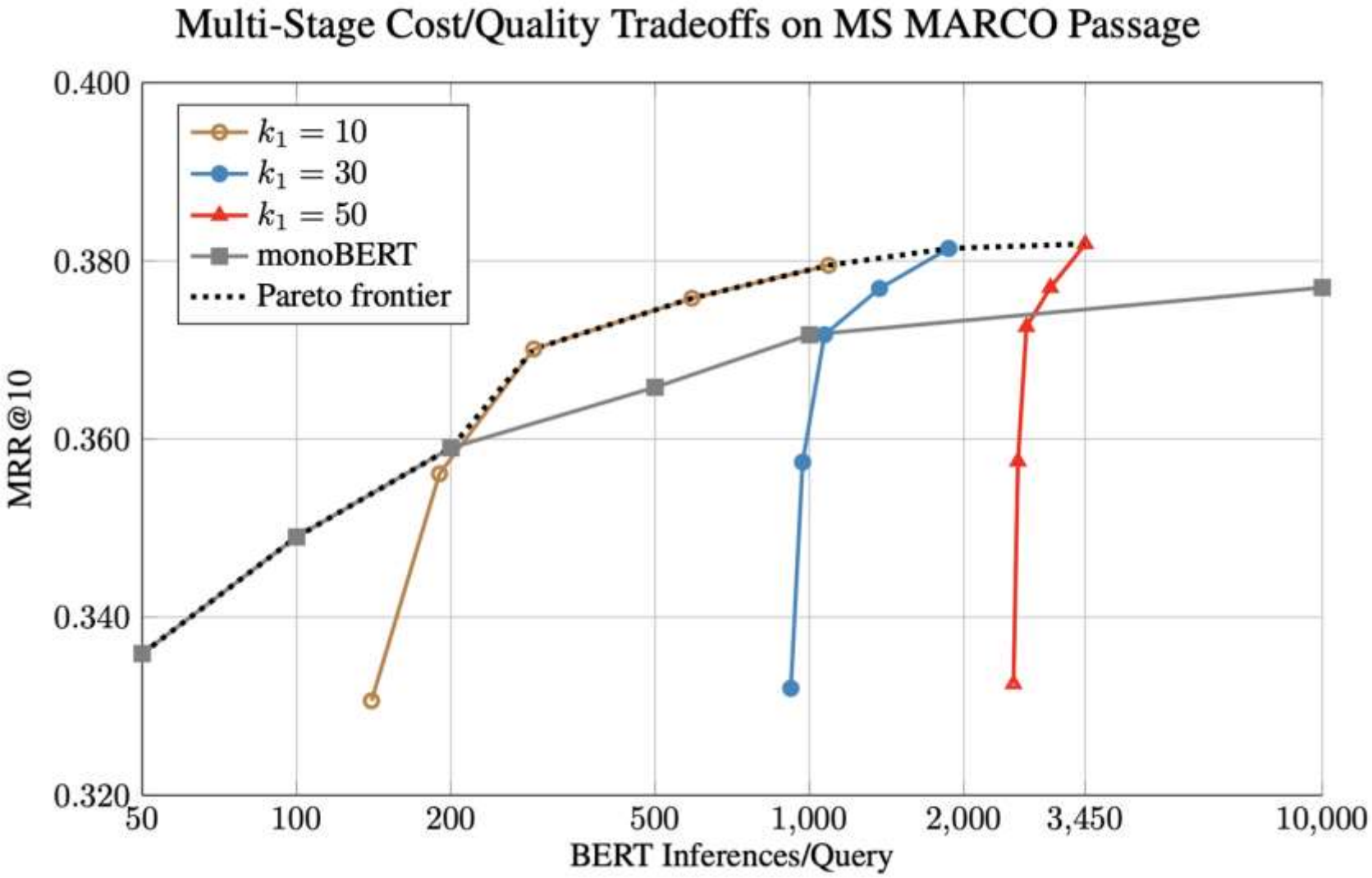
Pairwise aggregation:

$$s_1 = p_{1,2} + p_{1,3}$$

$$s_2 = p_{2,1} + p_{2,3}$$

$$s_3 = p_{3,1} + p_{3,2}$$

monoBERT vs. duoBERT



Takeaways of Multi-stage Rerankers

Advantage:

- more tuning knobs → more flexibility in effectiveness/efficiency tradeoff space

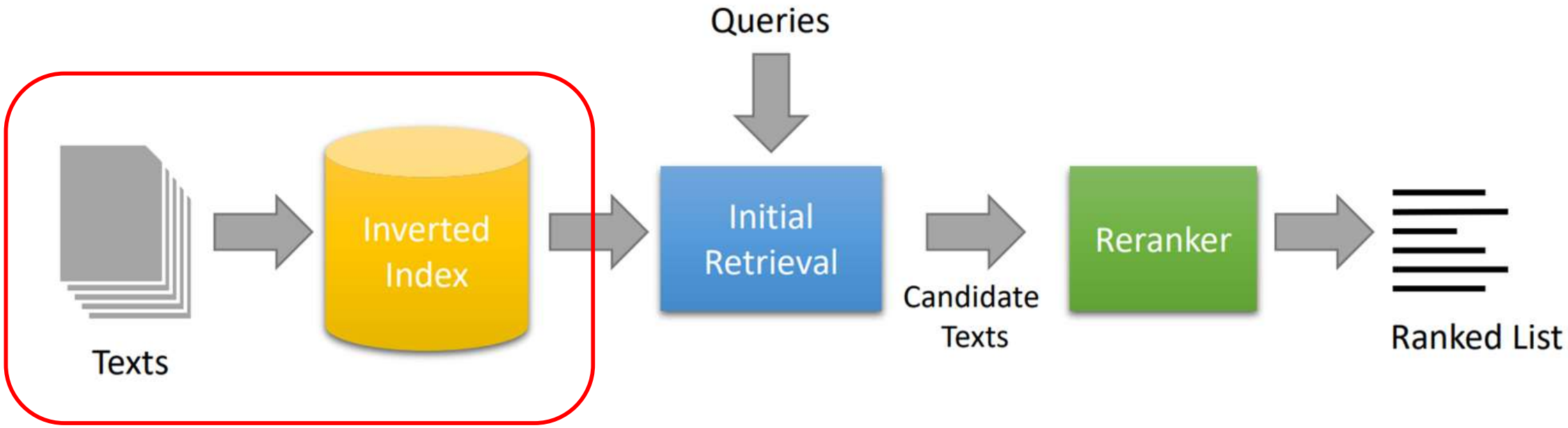
Disadvantage:

- more tuning knobs → more complexity

Document Preprocessing Techniques

Query vs document expansion
doc2query
DeepCT
DeepImpact

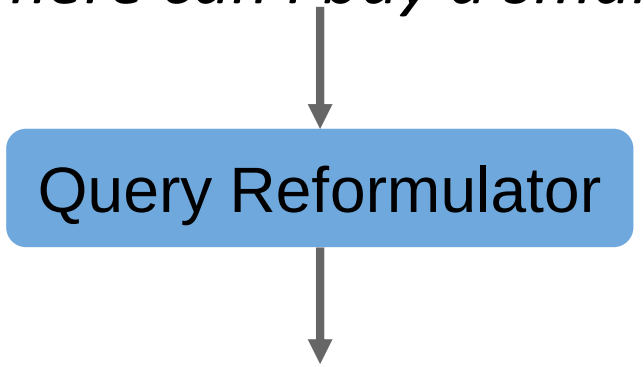
A Simple Search Engine



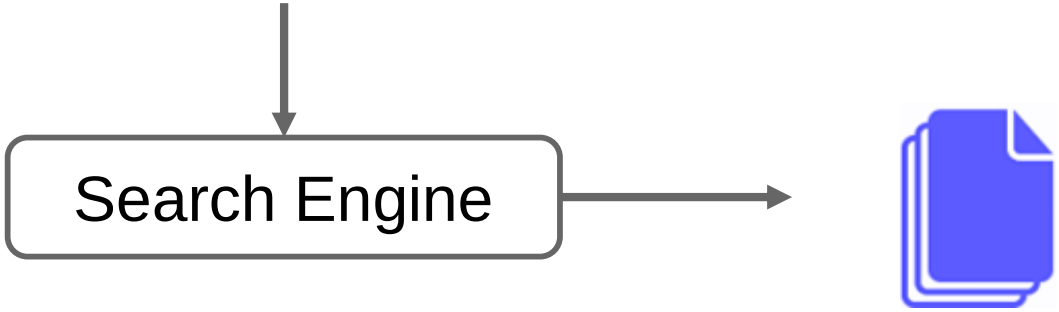
This section

Query Reformulation (aka query expansion)

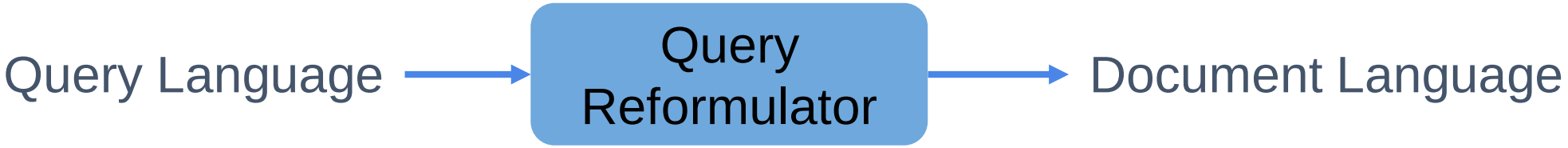
query: "*Where can I buy a small car?*"



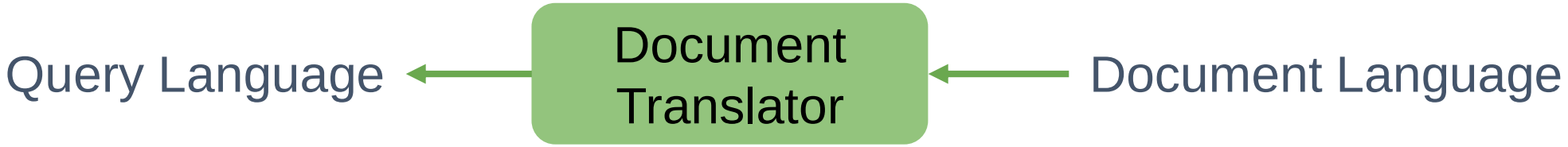
reformulated query: "*compact car sales store*"



Query reformulation as a translation task



Hard: Input has little information



Easier: Input has a lot of information

Document Preprocessing Techniques

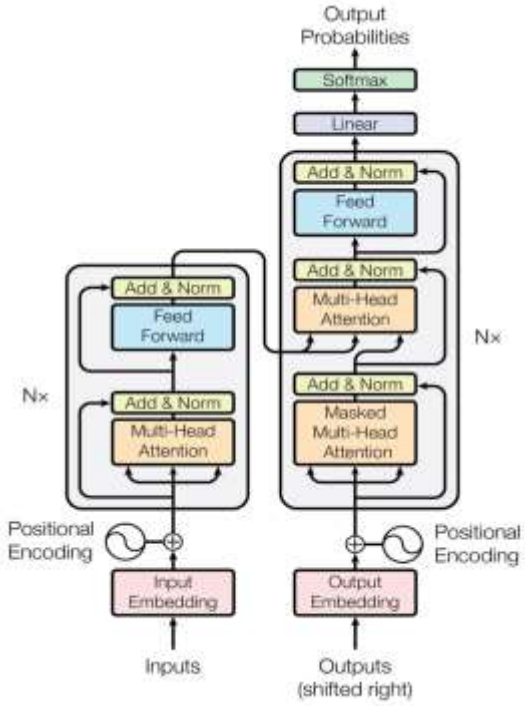
Query vs document expansion
doc2query
DeepCT
DeepImpact

doc2query

Document → seq2seq Transformer → Query

Supervised training:

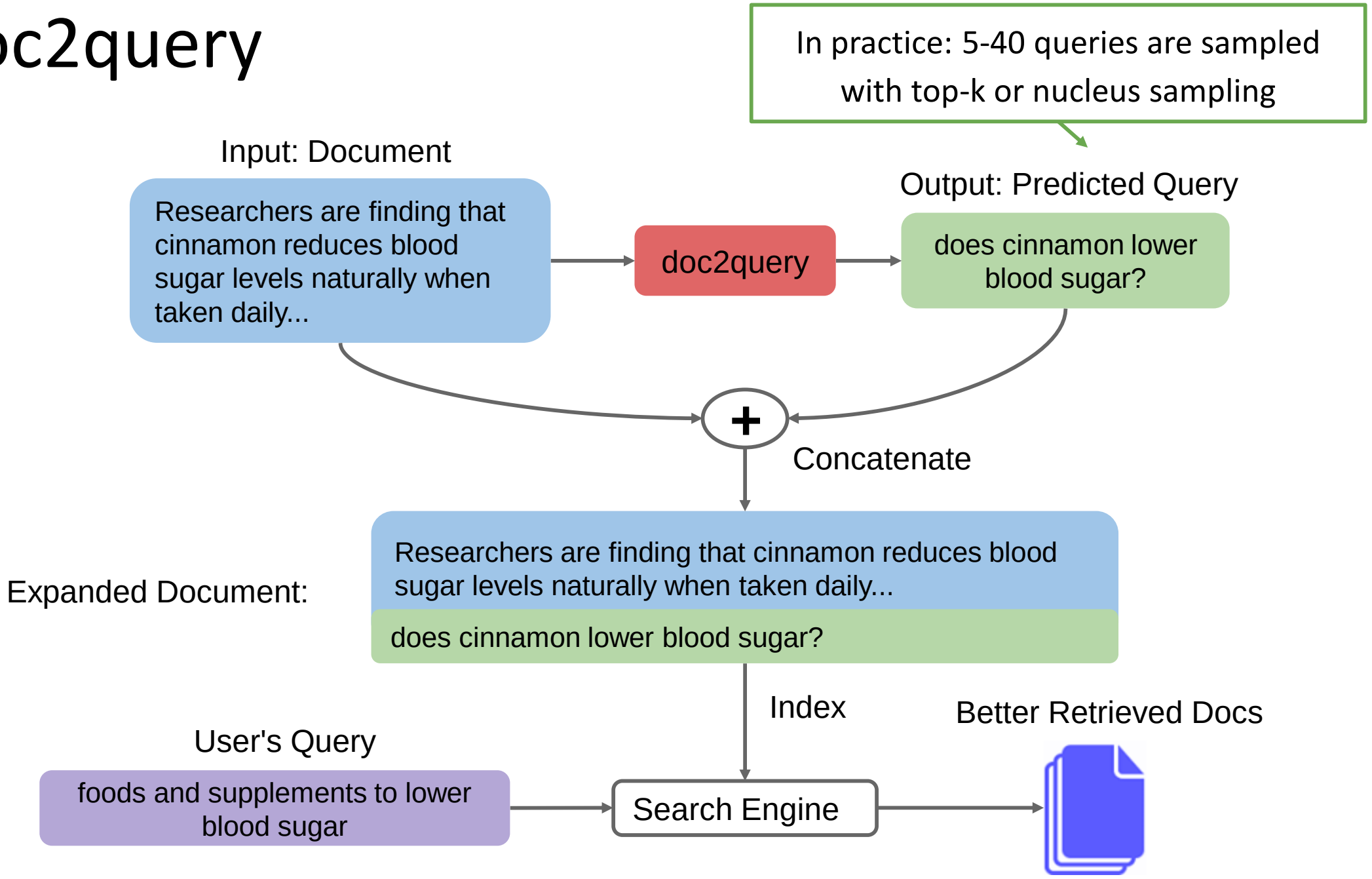
pairs of <query, relevant document>



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

Source: Vaswani et al., 2017

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

Results

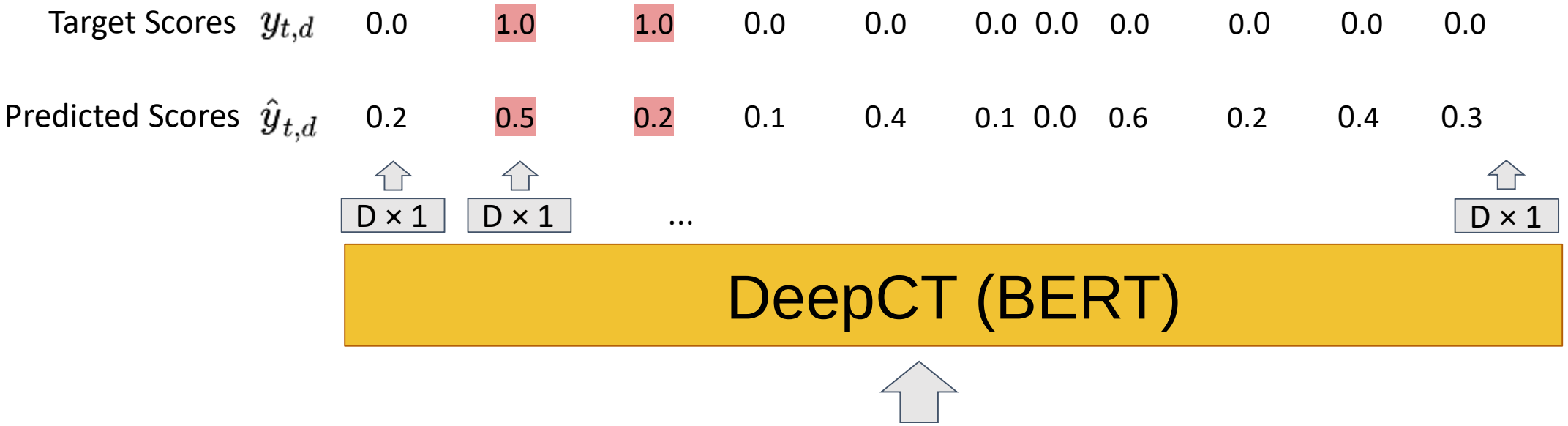
	MARCO Passage (MRR@10)	TREC-DL 19 (nDCG@10)
BM25	0.184	0.506
+ doc2query	**0.277**	**0.642**

Document Preprocessing Techniques

Query vs document expansion
doc2query
DeepCT
DeepImpact

DeepCT

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

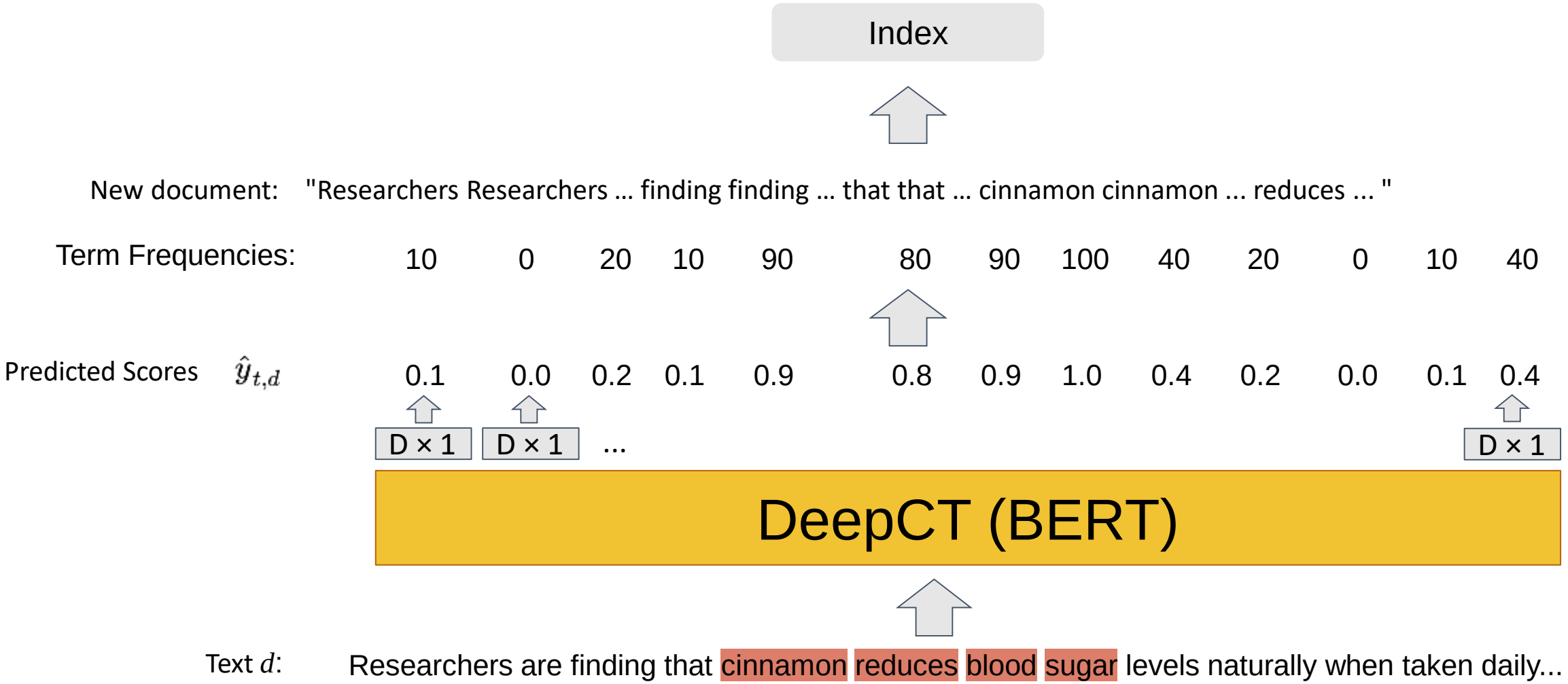


Text d: The Geocentric Theory was proposed by the greeks under the guidance...

Relevant query q: "who proposed the geocentric theory"

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

Once DeepCT is trained...



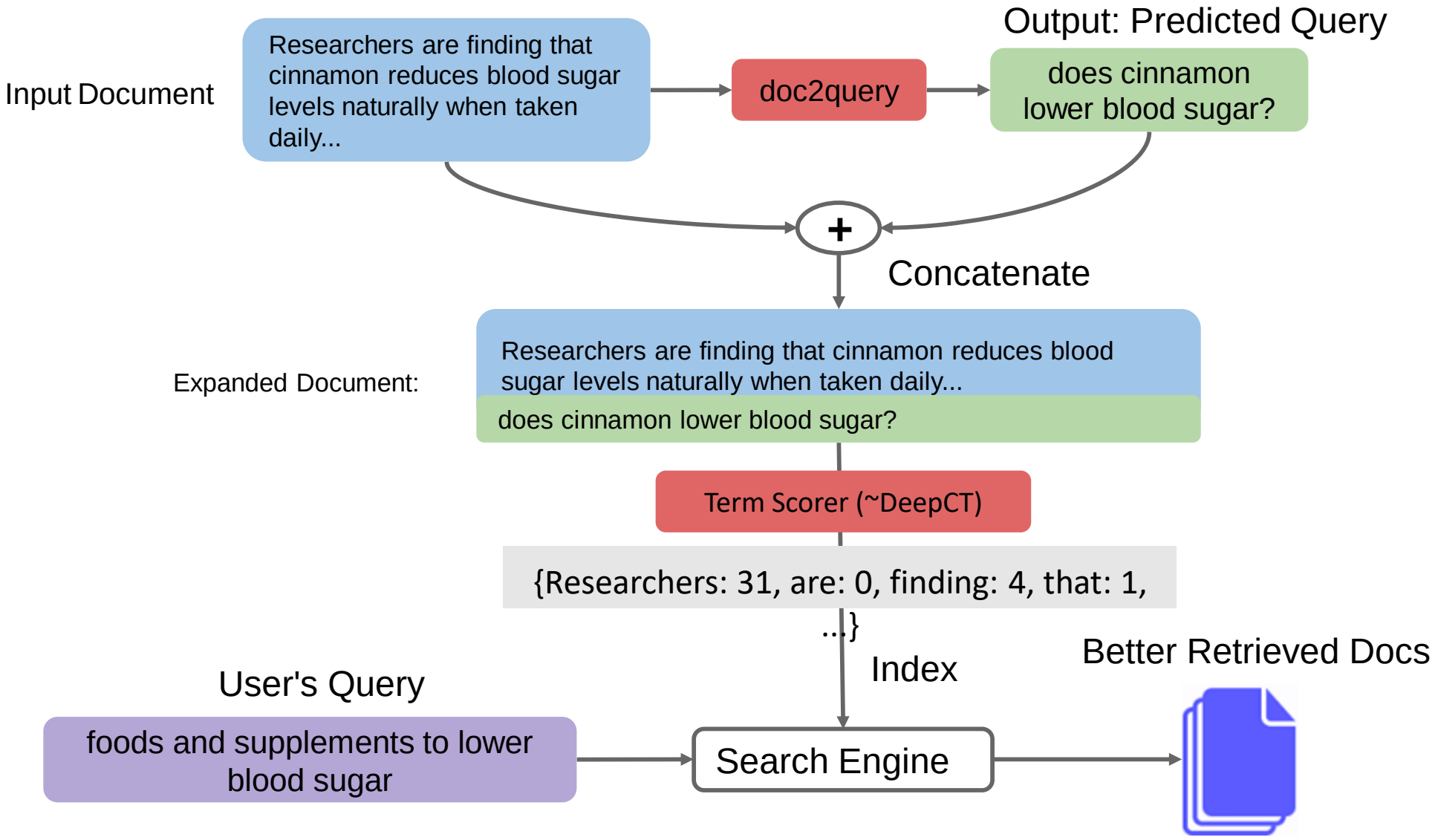
Results on MS MARCO Passage Dev Set

Model	MRR@10	R@1000	BERT Inferences per doc
BM25	0.184	0.853	-
+ doc2query	0.229	0.907	1
+ doc2query	0.277	0.944	40
DeepCT	0.243	0.913	1

Document Preprocessing Techniques

Query vs document expansion
doc2query
DeepCT
DeepImpact

DeepImpact: combining doc2query with DeepCT

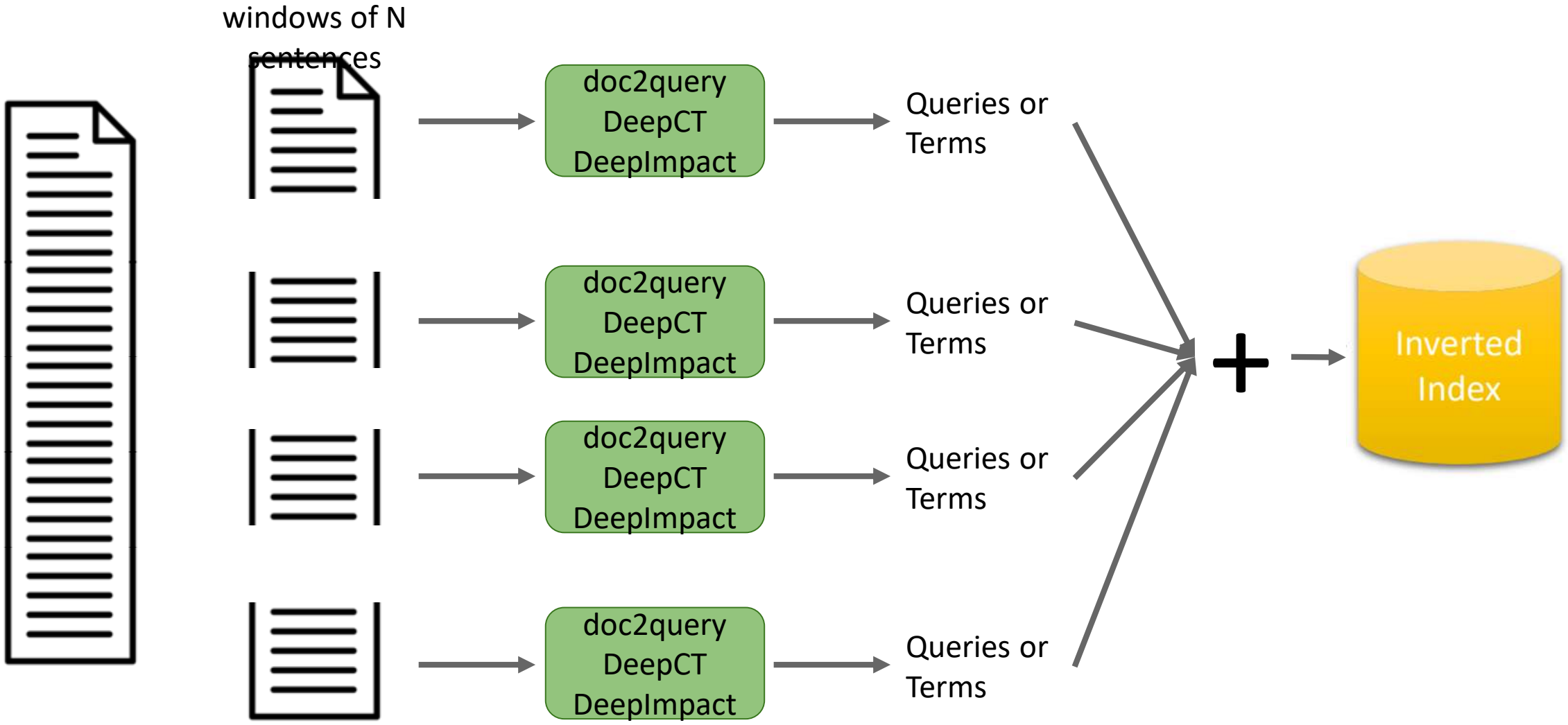


Mallia, Khattab, Tonellotto, and Suel. Learning Passage Impacts for Inverted Indexes. 2021

Results on MS MARCO Passage Dev Set

Model	MRR@10	R@1000	Latency (ms/query)
BM25	0.184	0.853	13
DeepCT	0.243	0.913	11
doc2query	0.278	0.947	12
DeepImpact	0.326	0.948	58
BM25 + monoBERT	0.355	0.853	(GPU) 10,700

How to Expand Long Documents?



Takeaways of Document Expansion

Advantages:

- Documents have more context than queries →easy prediction task
- Documents can be processed offline *and* in parallel
- Run on CPU at query time

Disadvantages:

- Have to iterate over the entire collection
- Not as effective as rerankers (yet)

Resources of Pre-training Methods in IR

Popular data repositories for the pre-training and fine-tuning process of PTMs in IR

Datasets for Pre-Training

- Pre-training objectives for IR are mostly based on large-scale collections
- Collections for pre-training tasks in IR must have the following properties:

1. **Large collection size**:
   - A necessity for pre-training tasks in any deep learning fields
2. **Structured documents**:
   - Title, passages, sub-title, html structure, etc. can be exploited to capture inter-page semantic relations
   - Hyperlinks between the pages provide intra-page semantic relations

- The second property is not always necessary for IR pre-training tasks
- But if a collection owns these properties, the collection might be better for IR pre-training tasks

General Text Corpora

- General text corpora are widely used in NLP researches for different tasks in different domains
- Contain large amounts of documents, suitable for classic pre-training tasks, e.g., masked language modeling and next sentence prediction

Dataset	Source	#Docs	Language	Latest crawl date
Books[1]	Book	74M	ENG	2015
C4[2]	web extracted text	0.3B	ENG	2019
Wikipedia[3]	Wiki text	10M	Multi-lang	monthly update
RealNews[4]	News	120GB	ENG	2019
Amazon[5]	reviews	11GB	ENG	2003

General Text Corpora: Books, C4

- *Books*:
  - Align books with the corresponding movie releases by associating the visual information with descriptive text
  - The text conveys both visual content (how a character, an object or a scene looks like) and high-level semantics (what someone is thinking, feeling and how these states evolve through a story)
- *C4*: (Colossal Clean Crawled Corpus)
  - More than 300 GBs clean English text scraped from the web
  - Can be used to pretrain language models and word representations

General Text Corpora: Wikipedia, RealNews, Amazon Reviews

- *Wikipedia*:
  - Wikimedia wikis
  - Well-organized document structures, entity links, and rich information
- *RealNews*:
  - News articles from Common Crawl
  - More than 5000 news domains indexed by Google News
  - Train data: December 2016 to March 2019; Evaluation: April 2019
- *Amazon Reviews*:
  - Amazon shopping reviews
  - Spans a period of 18 years, more than 35 million reviews up to March 2013
  - Reviews include user and product information, ratings, and a plaintext review

IR Related Corpora

- Contain documents similar to downstream IR tasks
- Minimize the gap between pre-training and downstream IR tasks to achieve better ranking performance

Dataset	Source	#Docs	Language	Latest crawl date
WT10G[6]	web pages	1.7M	ENG	1997
GOV2[7]	pages in .GOV	25M	ENG	2004
CWP200T	Chinese web pages	7B	CHN	2015
SogouT[8]	Sogou web pages	1.17B	CHN	2016
ClueWeb09[9]	web pages	1.04B	Multi-lang	2009
ClueWeb12[10]	web pages	0.73B	ENG	2012
MS MARCO[11]	Bing web pages	3.2M	ENG	2018

IR Related Corpora: WT10G, GOV2

- *WT10G* (Web Track 10Gigabytes)
  - Collected by CSIRO in Australia
  - A crawl of web pages in 1997 and applied in many web-based experiments
  - Retains the properties of the 1997 web content which includes: the graph structure of web links, server size distribution, inclusion of inter-domain links and web pages on various subjects
  - The page content and hyperlinks can be used in pre-training tasks
- *GOV2*:
  - Crawl of .gov sites in the early 2004
  - Includes html, text and the extracted text of pdf, word and postscript
  - About 426 GB and contains 25 million documents
  - Has potential for pre-training tasks with text-based self-supervised learning objectives

IR Related Copora: SogouT, Clueweb

- *SogouT*:
  - A Chinese web page collection,
  - Sampled from Sogou search engines's indexed documents considering both quality and diversity
  - Auxiliary resources including hyperlinks, query logs, and word embeddings
  - Suitable for pre-training tasks in Chinese IR
- *Clueweb*:
  - A large-scale web document collection provided by CMU
  - Clueweb09 contains about 1 billion web pages in 10 languages collected in Jan and Feb 2009
  - Clueweb12 was created based on Clueweb09 with several data cleaning strategies

IR Related Copora: MS MARCO

- *MS MARCO*
  - A large-scale document collection consisting of 3.2 million documents from the Bing search engine
  - Includes 1 million non-question queries for different retrieval tasks

Datasets for Fine-Tuning

- Document-oriented tasks
- Query-oriented tasks

Document-oriented Tasks

- *First stage retrieval* (**FSR**)
  - Retrieval stage from the full collection
- *Ad-hoc ranking* (**AR**)
  - Ranking a candidate list given a query
- *Session search* (**SS**):
  - Ranking a candidate list given a query and historical interactions
- *Multi-modal ranking* (**MMR**):
  - Given a query, rank the candidate list where each item contains multiple heterogeneous information such as text, picture and html structure
- *Personalized Search* (**PS**):
  - User-specific Ranking

Query-oriented Tasks

- *Query reformulation* (**QR**)
  - Iteratively modifying a query to improve the quality of search engine results in order to enhance user's search satisfaction
- *Query suggestion* (**QS**)
  - Providing a suggestion which may be a reformulated query to better represent a user's search intent
- *Query clarification* (**QC**)
  - Identifying user's search intent during a session

Other Tasks

- *Document summarization* (**DS**)
  - Shortening a document to create a subset (or a summary) that represents the most important information in this document
- *Snippet generation* (**SG**)
  - Query-specific document summarization
- *Keyphrase extraction* (**KE**)
  - Also known as Keyword Extraction
  - Aims to automatically extract the most used and most important terms in a document

Fine-Tuning Collections: Robust, TREC MQ

- Robust
  - Robust track is a classic ad-hoc retrieval task in TREC which focuses on poorly performing topics
  - Only includes 250 queries in Robust04 and 50 queries in Robust05
  - Used for evaluation in most experimental settings
- TREC Million Query (MQ)
  - Ad-hoc retrieval over a large-scale collection of queries and documents
  - A four-level relevance judgement for each query-document pair

Fine-Tuning Collections: Clueweb, TREC web track

- Clueweb
  - A large-scale web search dataset provided by CMU
  - The “Category B” data set consists of the English pages, which is roughly the first 50 million pages of the entire data set
- TREC web track
  - Exploits the documents from Clueweb
  - The goal is to explore and evaluate specific aspects of Web retrieval, including traditional ad-hoc retrieval task, risk-sensitive task and diversity search task

Fine-Tuning Collections: TREC DL track, AOL

- TREC Deep Learning Track
  - Studies IR in a large training data regime
  - Contains two tasks: Passage ranking and document ranking;
    - Two subtasks are included in each case: full ranking and reranking
  - Usually used as an evaluation set by training a retrieval model on a large-scale dataset such as MS MARCO
- AOL
  - A public available query log released by the internet company AOL
  - Contains the query sessions, anonymized user ids and clicked documents, suitable for ad-hoc ranking, session search, personalized search, query reformulation and suggestion

Fine-Tuning Collections: Sogou-QCL, Sogou-SRR, Tiangong-ST

- All Created from Sogou search engine to support research on IR
- Sogou-QCL:
  - Consists of 537,366 queries, more than 9 million Chinese web pages, and five kinds of relevance labels assessed by click models
  - Also includes 2,000 queries with four-level human assessed relevance labels
- Sogou-SRR (Search Result Relevance) :
  - Consists of 6,338 queries and corresponding top 10 search results
  - Each search result contains the screenshot, title, snippet, HTML source code, parse tree, url as well as a four-grade relevance score (1-4) and the result type
  - The heterogeneous information provides opportunity for multi-modal ranking
- Tiangong-ST:
  - 147,155 refined Web search sessions, 40,596 unique queries, 297,597 web pages, and six kinds of weak relevance labels assessed by click models.
  - Different from Sogou-QCL and Sogou-SRR, the session information provided in this dataset is able to be used in session search ranking

Fine-Tuning Collections: Qulac

- Qulac
  - Collected through crowdsourcing in terms of the topics in the TREC Web Track 2009-2012
  - A dataset on asking Questions for Lack of Clarity in open-domain information-seeking conversations
  - Contains 198 topics where each topic is recognized as either “ambiguous” or “faceted”
  - The clarifying questions are collected based on each topic through crowdsourcing
  - Based on each topic-facet pair, the answers to each clarifying question are collected
  - The average number of facets per topic is 3.85 ± 1.05
  - The facets and topics in this collection can be used for query clarification task

Fine-Tuning Collections: BEIR, MS MARCO

- BEIR (Benchmarking IR)
  - A new heterogeneous benchmark containing different IR tasks
  - The benchmark contains 18 datasets covering 9 IR tasks (Fact Checking, Citation Prediction, Duplicate Question Retrieval, Argument Retrieval, News Retrieval, Question Answering, Tweet Retrieval, Biomedical IR, Entity Retrieval) from 17 different datasets
  - Used to systematically study the zero-shot generalization capabilities of several neural retrieval methods
- MS MARCO
  - A popular large-scale document collection with about 3.2 million available documents, which are from the Bing search engine
  - 1 million non-question queries are also included in this dataset for different retrieval tasks

Fine-Tuning Collections: TREC CAR, CNN/Daily Mail

- TREC Complex Answer Retrieval (CAR):
  - Uses topics, outlines, and paragraphs extracted from English Wikipedia
  - Wikipedia articles split into sections outlines and the contained paragraphs
  - The complex topics are selected from articles on open information needs
  - Contains a passage task and an entity task, where the latter can be used in keyphrase extraction tasks
- CNN/Daily Mail:
  - A large-scale collection of news articles and further modified for summarization
  - Consists of more than 280,000 training samples and 11,490 test set samples
  - Train set documents have 29.74 sentences with 766 words on average
  - Summaries consist of 53 words and 3.72 sentences on average

Fine-Tuning Collections: NYT, Debatepedia

- New York Times (NYT)
  - A large-scale document summarization dataset
  - Contains articles from *NYT* between 1987 and 2007
  - Summaries were written by library scientists
  - Particularly useful as an extractive summarization dataset
- Debatepedia
  - Collected from *debatepedia.org*
  - An encyclopedia of pro and con arguments and quotes on critical debate topics
  - There are totally 663 debates in the corpus, which belong to 53 overlapping categories such as Politics, Law, Crime, Environment, Health, Morality, Religion, etc.
  - 5 queries per debate and 4 documents per query on average

Fine-Tuning Collections: DUC, WIKIREF

- DUC:
  - A dataset for document summarization.
  - In most experiments, it is used for testing only
  - It consists of 500 news articles, each of the article is paired with four human written summaries
- WIKIREF
  - A query-focused summarization dataset from Wikipedia, to generate summarization with a given query
  - Contains more than 280,000 examples

Dataset	Subdata	Size	Source	Potential Tasks
Robust	Robust04	0.5M docs, 250 queries	TREC Robust Track	FSR, AR, QR
	Robust05	1M docs, 50 queries		
TREC MQ	MQ2007	6.5K docs, 1.7K queries	TREC Million Query track	FSR, AR, QR
	MQ2008	1.4K docs, 784 queries		
Clueweb	09-CatB	50M docs, 150 queries	Web pages	FSR, AR, QR, KE
	12-CatB	50M docs		
TREC web track	99-2014	See [13]	TREC web track	FSR, AR, QR
TREC DL track	2019-2021	See [14]	TREC Deep Learning track	FSR, AR
AOL	\	6M queries	AOL Query logs	AR,SS,PS,QR,QS
Sogou-QCL	\	9M docs, 0.5M queries	Sogou Query logs	AR, QR
Sogou-SRR	\	63K results, 6K queries	Sogou Query logs	AR, MMR, QR
Tiangong-ST	\	0.3M docs, 40K queries	Sogou Query logs	AR, SS, QR, QS
Qulac	\	10K question-answer pairs	TREC Web Track	AR, QR, QC
BEIR	7 IR tasks	Vary from tasks	Wiki, Quaro, Twittter, News and etc.	FSR,AR, etc.
MS MARCO	2019-20	1M queries, 8.8M passages, 3.2M docs	TREC Deep Learning Track	FSR, AR, QR
TREC CAR	\	30M paras, 2M queries	TREC Complex answer retrieval	AR, QR, KE
CNN / Daily Mail	\	0.3M docs	Human generated abstracts	DS
New York Times (NYT)	\	1.8M docs	News articles	DS
Debatepedia	\	1,303 debates	Debate key points	SG, DS
DUC	2001-07	300 clusters, See [15]	Doc understanding conference	SG, DS
WIKIREF	\	0.3M samples	QFS benchmark	SG, DS

Leaderboards

1. MS MARCO (Passage retrieval and document retrieval task): https://microsoft.github.io/msmarco/
2. DuReader (Machine Reading Comprehension task): https://ai.baidu.com/broad/leaderboard?dataset=dureader
3. Robust04 (Document retrieval task): https://paperswithcode.com/sota/ad-hoc-information-retrieval-on-trec-robust04
4. CNN/Mail (Documents summarization task): https://paperswithcode.com/sota/document-summarization-on-cnn-daily-mail
5. Baidu DuIE (Entity extraction task): https://ai.baidu.com/broad/leaderboard?dataset=dureader
6. Benchmarking IR (BEIR) (Passage retrieval and document retrieval task): https://github.com/UKPLab/beir

Questions?