Neural Information Retrieval

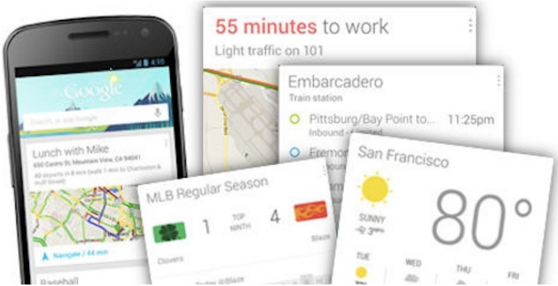
[ELE680] Deep Neural Networks

Krisztian Balog
University of Stavanger

Week 37, 2021

Information Retrieval (IR)

"Making the **right information** available to the **right person** at the **right time** in **the right form**."



Classic IR problem

Ad hoc document retrieval

- Given a collection of documents D and a search query q
- Score all documents $d \in D$ in the collection by computing $score(d, q)$
- Return the top-scoring documents as results

Traditional text representation

Bag-of-words text representation

- Simplified representation of text as a bag (multiset) of words
- Disregards word ordering, but keeps multiplicity

Example: *"the dog ate my homework and my shoes"*

	ate				dog		my	
0	1	0	...	0	1	0	2	...

Traditional retrieval models

Common form of a retrieval function

$$score(d, q) = \sum_{t \in q} w_{t,d} \times w_{t,q}$$

- Note: we only consider terms in the query, $t \in q$
- $w_{t,d}$ is the term's weight in the document
- $w_{t,q}$ is the term's weight in the query

$score(d, q)$ is (in principle) to be computed for every document in the collection

Fundamental challenge

Vocabulary mismatch

q:	dog		first		aid				
	1		1		1				

vs.

d:						vet		clinic	
						1		1	

Word embeddings

- Static embeddings (Word2vec, GloVe)
- Contextual embeddings (GPT, ELMO, BERT, RoBERTa)

Ranking using static embeddings

Word2vec

- Words are represented as dense, continuous vectors of lesser dimensionality:

$$\mathbf{v}_{\text{hotel}} = \begin{pmatrix} 0.19 & 0.2 & -0.9 & 0.4 \end{pmatrix}$$
$$\mathbf{v}_{\text{motel}} = \begin{pmatrix} 0.27 & 0.01 & -0.7 & 0.3 \end{pmatrix}$$

Word2vec

- Words are represented as dense, continuous vectors of lesser dimensionality:

$$\mathbf{v}_{\text{hotel}} = \begin{pmatrix} 0.19 & 0.2 & -0.9 & 0.4 \end{pmatrix}$$
$$\mathbf{v}_{\text{motel}} = \begin{pmatrix} 0.27 & 0.01 & -0.7 & 0.3 \end{pmatrix}$$

- Straightforward way of measuring document-query similarity (unsupervised):
  - Create vector-based representations of queries and documents, $\mathbf{v}_q$ and $\mathbf{v}_d$, by taking the centroid of their word vectors
  - Score documents based on the cosine similarity of their embeddings vectors to that of the query:

$$score(d,q) = \cos(\mathbf{v}_d, \mathbf{v}_q)$$

Word2vec

- Words are represented as dense, continuous vectors of lesser dimensionality:

$$\mathbf{v}_{\text{hotel}} = \begin{pmatrix} 0.19 & 0.2 & -0.9 & 0.4 \end{pmatrix}$$
$$\mathbf{v}_{\text{motel}} = \begin{pmatrix} 0.27 & 0.01 & -0.7 & 0.3 \end{pmatrix}$$

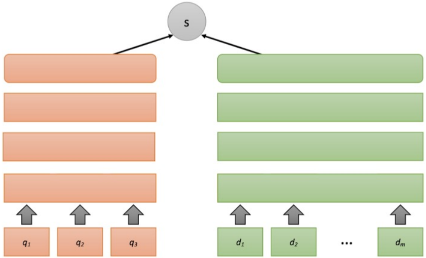
- Straightforward way of measuring document-query similarity (unsupervised):
  - Create vector-based representations of queries and documents, $\mathbf{v}_q$ and $\mathbf{v}_d$, by taking the centroid of their word vectors
  - Score documents based on the cosine similarity of their embeddings vectors to that of the query:

$$score(d, q) = \cos(\mathbf{v}_d, \mathbf{v}_q)$$

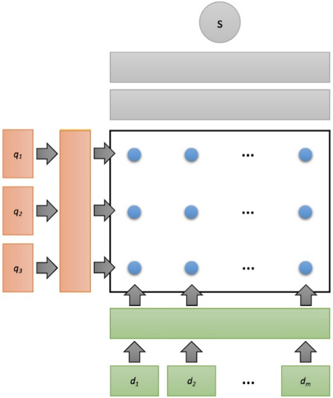
- **What about supervised ranking (i.e., learning the score function)?**

Neural ranking models[1]

Representation-based



Interaction-based



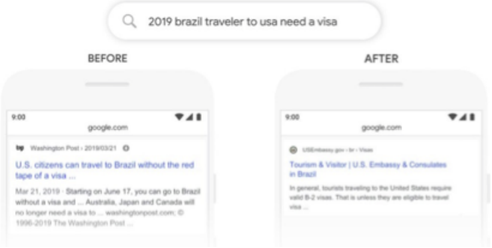
[1]Mitra and Craswell. An Introduction to Neural Information Retrieval. FnTIR 2017. https://arxiv.org/abs/1705.01509

Ranking using contextual embeddings (BERT)

Adoption by commercial search engines[2,3]

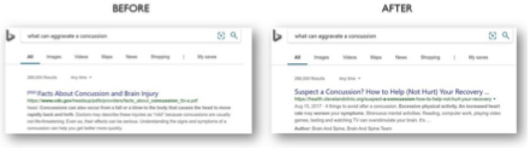
Google

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.[1]



Microsoft 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.[2]

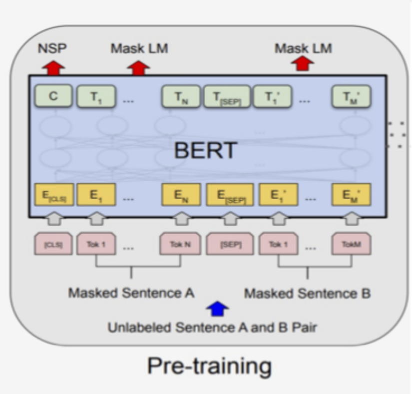


---

[2] `https://blog.google/products/search/search-language-understanding-bert/`

[3] `https://azure.microsoft.com/en-us/blog/bing-delivers-its-largest-improvement-in-search-experience-using-azure-gpus/`

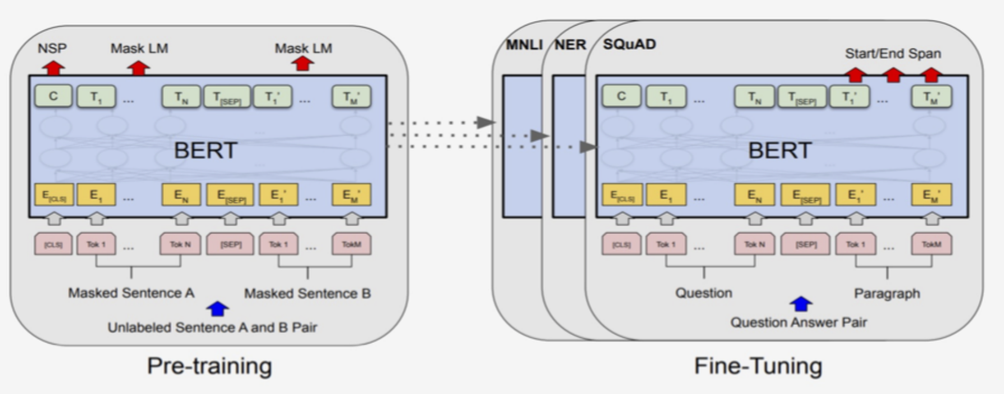
BERT Recap[4]



Self-supervised: ∞ training data

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

BERT Recap[5]

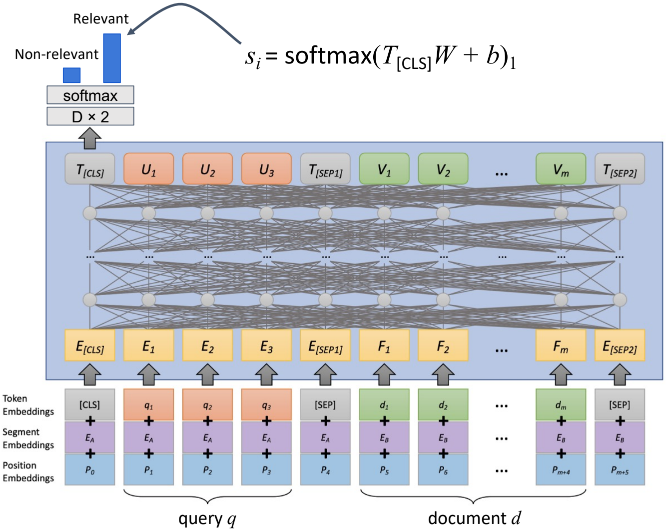


Self-supervised: ∞ training data

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

BERT for relevance classification (MonoBERT)

$$score(d, q) = P(Relevant = 1|d, q)$$



Training MonoBERT

Loss:

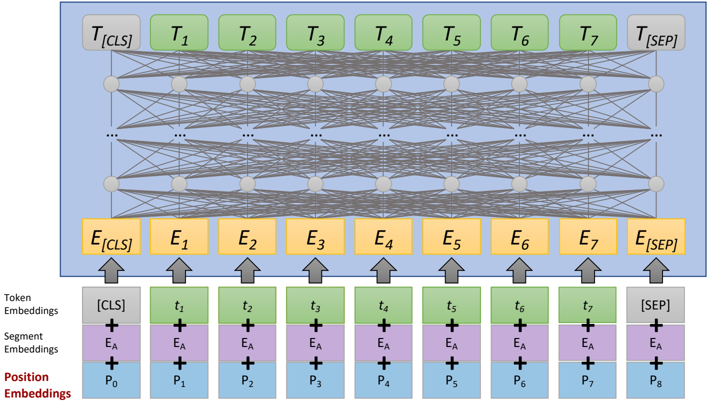
$$L = -\sum_{d \in D^+} \log score(d, q) - \sum_{d \in D^-} \log(1 - score(d, q))$$

- D^+: human-annotated data
- D^-: sampled from top-k ranked documents by traditional ranker

BERT's limitations

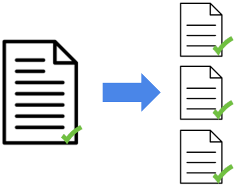
Cannot input entire documents!

Need separate embedding for every possible position (restricted to 512)

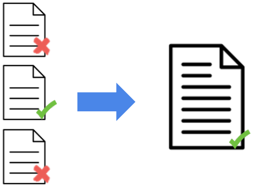


From documents to passages

Training time
Transfer labels

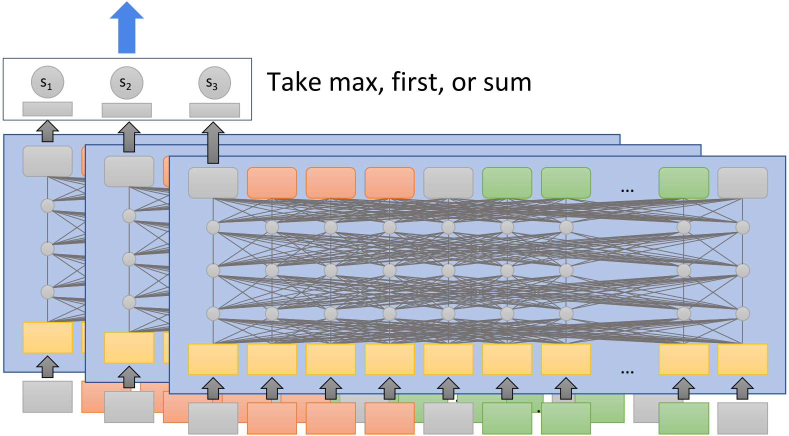


Inference time
Aggregate evidence



Aggregating passage scores (BERT-MaxP, FirstP, SumP)[6]

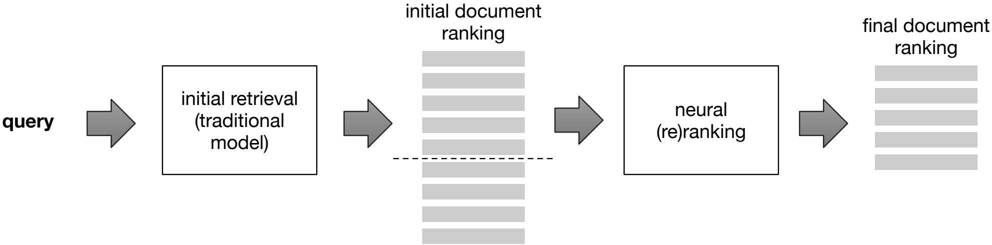
Document Score



Take max, first, or sum

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

Neural ranking in practice



Further reading

- ECIR 2021 tutorial by MacAvaney, Macdonald, and Tonellotto
  `https://github.com/terrier-org/ecir2021tutorial`
- WSDM 2021 tutorial by Yates, Nogueira, and Lin
  `https://t.co/jjhMnMm0wb`
- Pretrained Transformers for Text Ranking: BERT and Beyond
  `https://arxiv.org/abs/2010.06467`