h2oloo at TREC 2020: Deep Learning Track

Ronak Pradeep, Hang Cui, Ruizhou Xu,
Rodrigo Nogueira, and Jimmy Lin

Slides adapted from Xinyu Zhang

h2oloo at TREC 2019

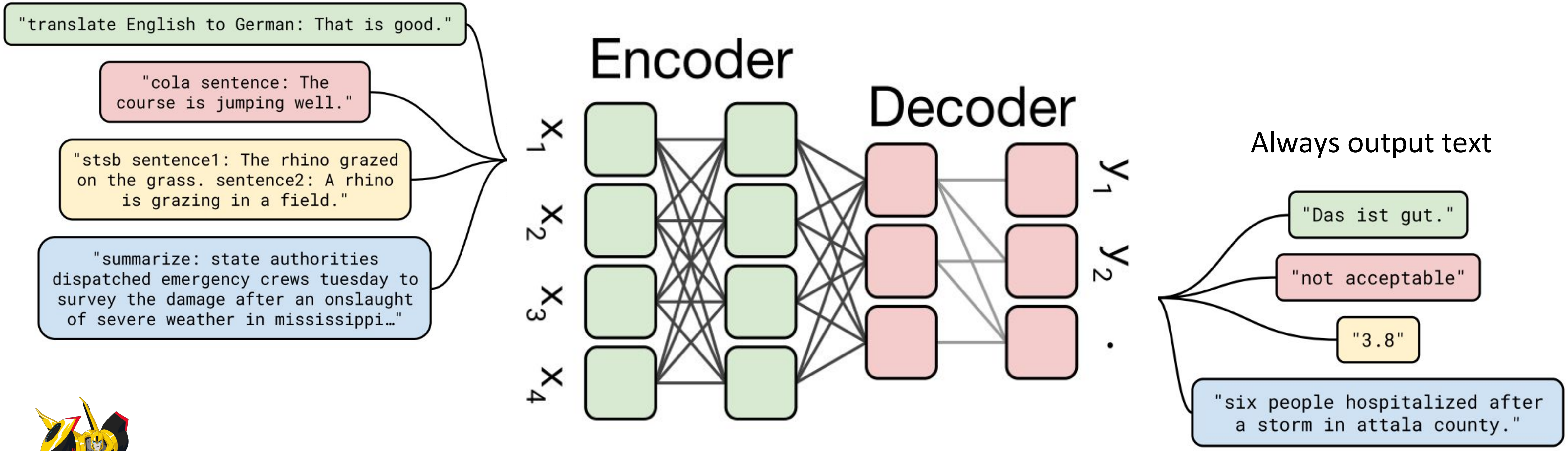
h2oloo at TREC 2020

Welcome T5

<u>Pretraining Dataset</u>

C4 (Colossal Clean Crawled Corpus)

Multitask Pretraining Mixture



Raffel, Colin, et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." *JMLR 2020*.

How did we rank: Multi-Stage Ranking Expando-Mono-Duo-T5

H_{-1}

docT5query

→

H_0

Anserini Retriever

top-k_0 passages →

H_1

monoT5

top-k_1 passages →

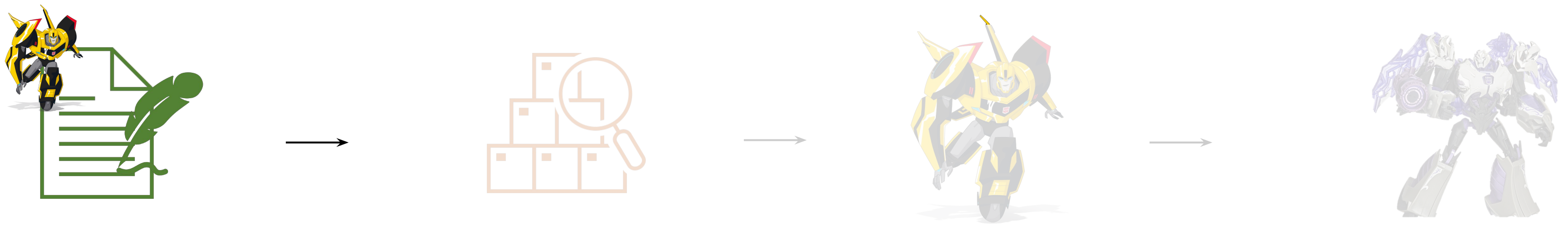
H_2

duoT5

Nogueira, Rodrigo, et al. "Multi-stage Document Ranking with BERT." *arXiv:1910.14424* (2019).
Zhang, Edwin, et al. "Covidex: Neural Ranking Models and Keyword Search Infrastructure for the COVID-19 Open Research Dataset." *EMNLP SDP 2020*.

Passages

How did we rank: (H_{-1}) docT5query



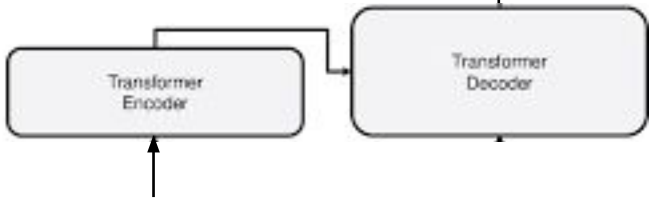
H_{-1} H_0 H_1 H_2

Generated Queries (N = 2):

what is the weather in washington dc?

when is the hottest month in washington dc?

top-k sampling (k=10)



Passage:

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.

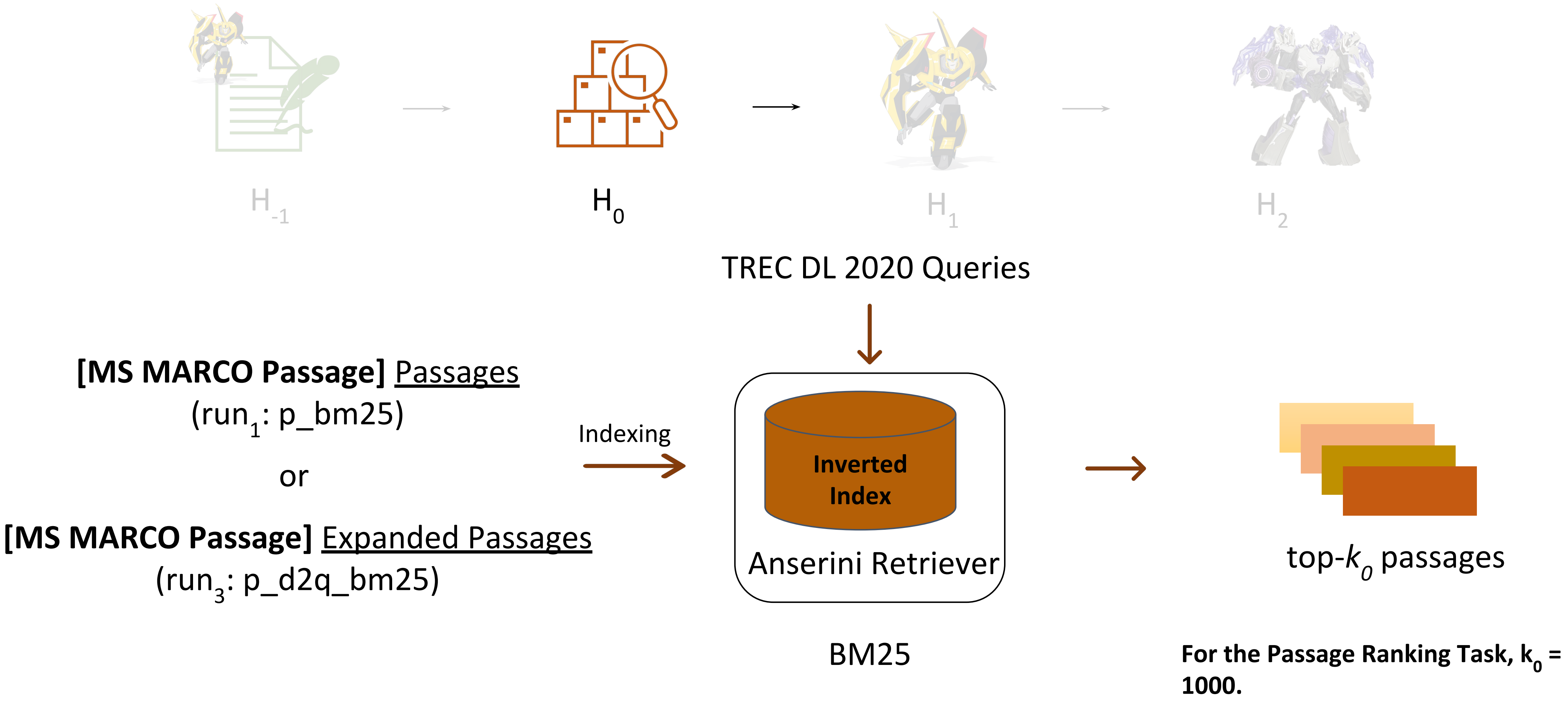
Expanded Passage:

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. *what is the weather in washington dc? when is the hottest month in washington dc?*

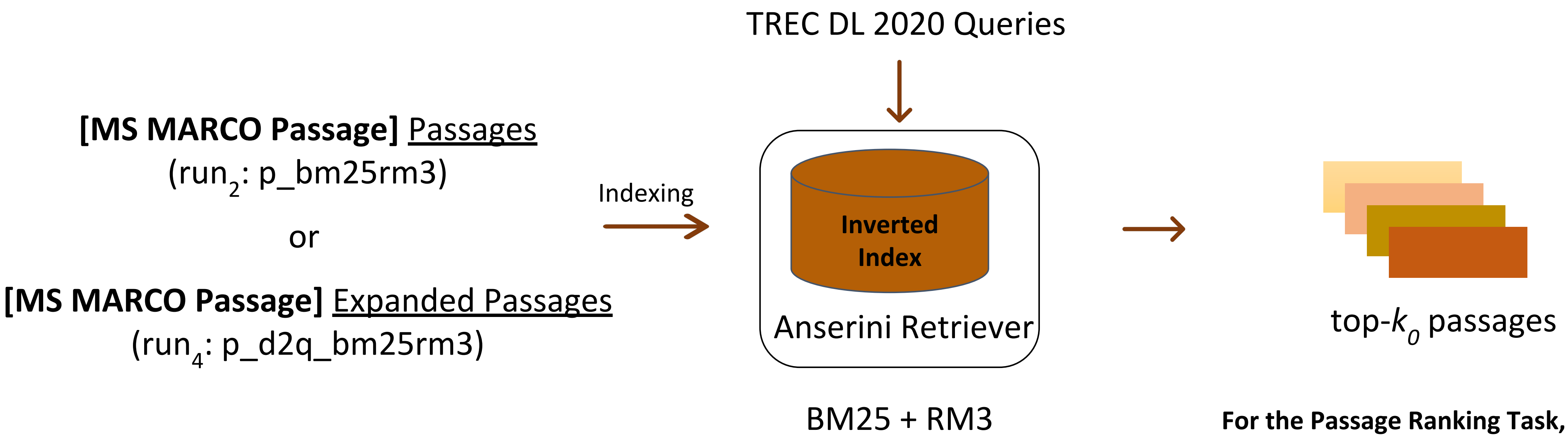
For the MS MARCO Passage Corpus, we generate N = 40 queries

Nogueira, Rodrigo, et al. "Document Expansion by Query Prediction." *arXiv:1904.08375* (2019).
Nogueira, Rodrigo, et al. "From doc2query to docTTTTTquery." (2019).

How did we rank: (H_0) Anserini Retriever

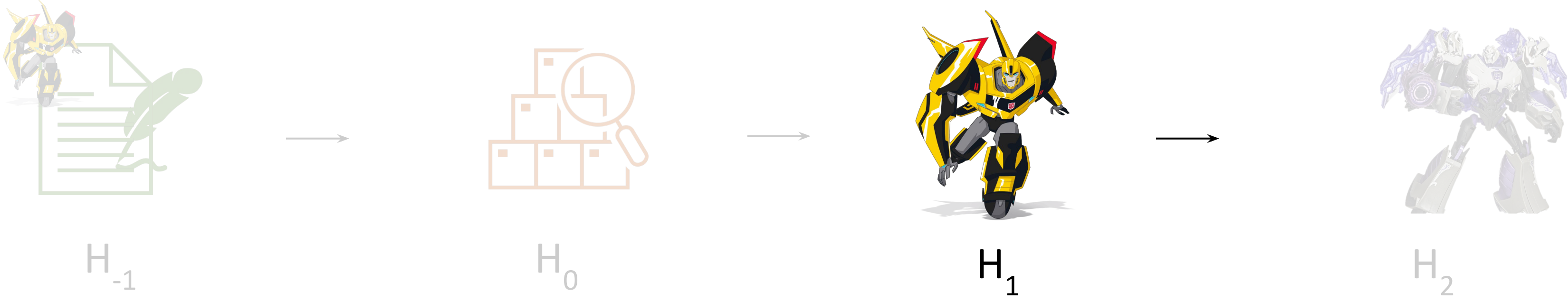


How did we rank: (H_0) Anserini Retriever

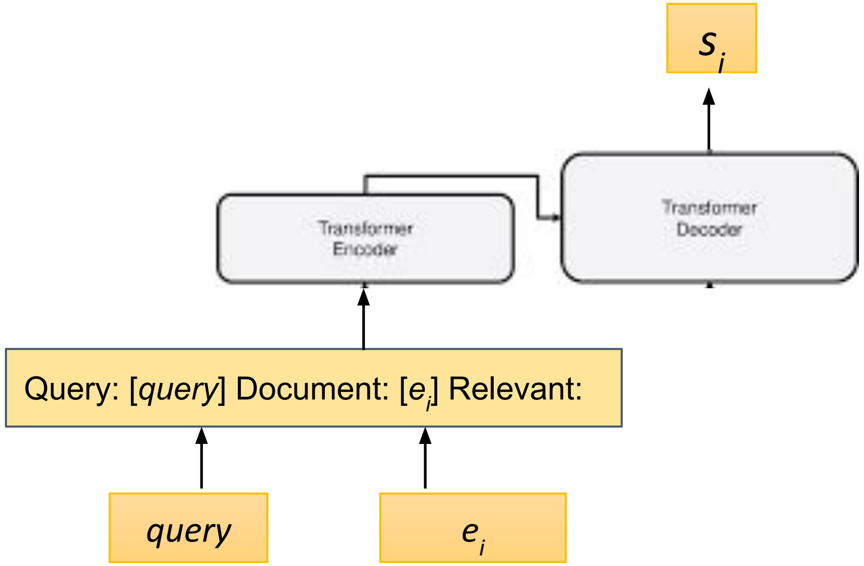


For the Passage Ranking Task, k_0 = 1000.

How did we rank: (H_1) monoT5



top-k_0 passages



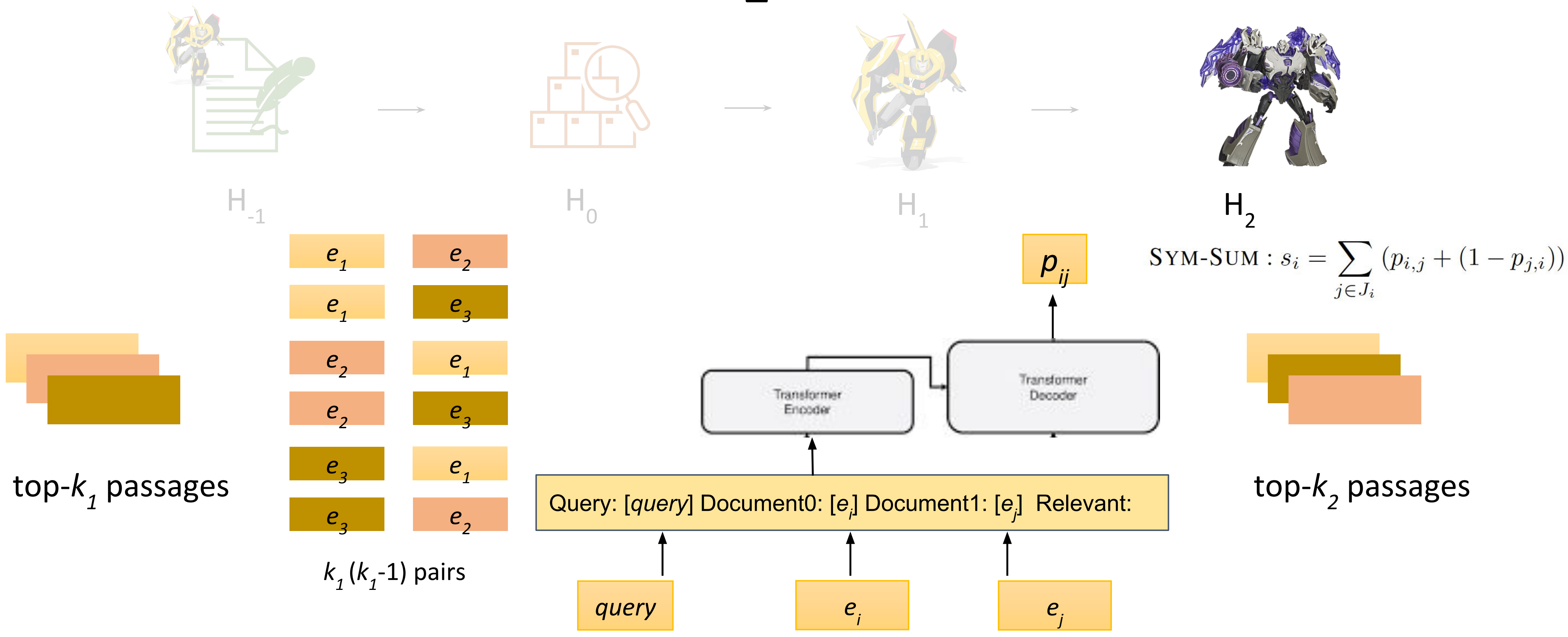
top-k_1 passages

Note: From here on, we use the original passage text, not the expanded one!

For the Passage Ranking Task, k_1 = 50.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. "Document Ranking with a Pretrained Sequence-to-Sequence Model." *Findings of EMNLP 2020*.

How did we rank: (H_2) duoT5



Results

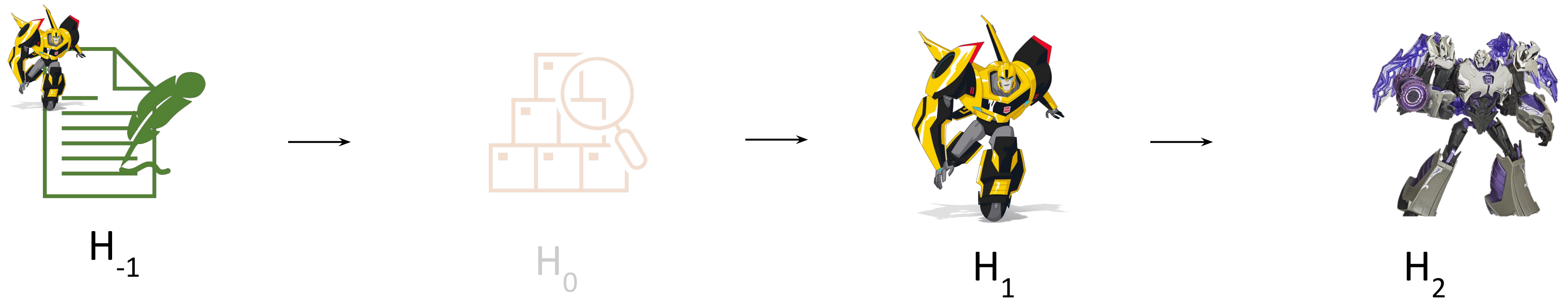
Run	AP	NDCG@10	NDCG@1k	RR	R@1k
(0) median	0.4413	0.6810	0.6631	0.8443	-
(1) p_bm25	0.2856	0.4796	0.5830	0.6585	0.7863
(2) p_bm25rm3	0.3019	0.4821	0.6046	0.6360	0.8217
(3) p_d2q_bm25	0.4074	0.6187	0.6840	0.7326	0.8452
(4) p_d2q_bm25rm3	0.4295	0.6172	0.7041	0.7424	**0.8699**
(5) p_bm25rm3_duo	0.5355	0.7583	0.7387	0.8759	-
(6) p_d2q_bm25_duo	0.5609	**0.7837**	0.7539	**0.8798**	-
(7) p_d2q_rm3_duo	**0.5643**	0.7821	**0.7732**	**0.8798**	-

Table 1: Results on TREC's Deep Learning 2020 Passage Ranking track.

run	group	subtask	neural	RR (MS)	RR	NDCG@10	NCG@1000	AP
pash_r3	PASH	rerank	nnlm	0.3678	0.9147	0.8031	0.7056	0.5445
pash_r2	PASH	rerank	nnlm	0.3677	0.9023	0.8011	0.7056	0.5420
pash_f3	PASH	fullrank	nnlm	0.3506	0.8885	0.8005	0.7255	0.5504
pash_f1	PASH	fullrank	nnlm	0.3598	0.8699	0.7956	0.7209	0.5455
pash_f2	PASH	fullrank	nnlm	0.3603	0.8931	0.7941	0.7132	0.5389
p_d2q_bm25_duo	h2oloo	fullrank	nnlm	0.3838	0.8798	0.7837	0.8035	0.5609
p_d2q_rm3_duo	h2oloo	fullrank	nnlm	0.3795	0.8798	0.7821	0.8446	0.5643
p_bm25rm3_duo	h2oloo	fullrank	nnlm	0.3814	0.8759	0.7583	0.7939	0.5355
CoRT-electra	HSRM-LAVIS	fullrank	nnlm	0.4039	0.8703	0.7566	0.8072	0.5399
RMIT-Bart	RMIT	fullrank	nnlm	0.3990	0.8447	0.7536	0.7682	0.5121
pash_r1	PASH	rerank	nnlm	0.3622	0.8675	0.7463	0.7056	0.4969
NLE_pr3	NLE	fullrank	nnlm	0.3691	0.8440	0.7458	0.8211	0.5245
pinganNLP2	pinganNLP	rerank	nnlm	0.3579	0.8602	0.7368	0.7056	0.4881
pinganNLP3	pinganNLP	rerank	nnlm	0.3653	0.8586	0.7352	0.7056	0.4918
pinganNLP1	pinganNLP	rerank	nnlm	0.3553	0.8593	0.7343	0.7056	0.4896
NLE_pr2	NLE	fullrank	nnlm	0.3658	0.8454	0.7341	0.6938	0.5117
NLE_pr1	NLE	fullrank	nnlm	0.3634	0.8551	0.7325	0.6938	0.5050
1	nvidia_ai_apps	rerank	nnlm	0.3709	0.8691	0.7271	0.7056	0.4899
bigIR-BERT-R	QU	rerank	nnlm	0.4040	0.8562	0.7201	0.7056	0.4845
fr_pass_roberta	BITEM	fullrank	nnlm	0.3580	0.8769	0.7192	0.7982	0.4990
bigIR-DCT-T5-F	QU	fullrank	nnlm	0.3540	0.8638	0.7173	0.8093	0.5004
rr-pass-roberta	BITEM	rerank	nnlm	0.3701	0.8635	0.7169	0.7056	0.4823
bcai_bertl_pass	bcai	fullrank	nnlm	0.3715	0.8453	0.7151	0.7990	0.4641
bigIR-T5-R	QU	rerank	nnlm	0.3574	0.8668	0.7138	0.7056	0.4784
2	nvidia_ai_apps	fullrank	nnlm	0.3560	0.8507	0.7113	0.7447	0.4866
bigIR-T5-BERT-F	QU	fullrank	nnlm	0.3916	0.8478	0.7073	0.8393	0.5101
bigIR-T5xp-T5-F	QU	fullrank	nnlm	0.3420	0.8579	0.7034	0.8393	0.5001
nlm-ens-bst-2	NLM	fullrank	nnlm	0.3542	0.8203	0.6934	0.7190	0.4598
nlm-ens-bst-3	NLM	fullrank	nnlm	0.3195	0.8491	0.6803	0.7594	0.4526
nlm-bert-rr	NLM	rerank	nnlm	0.3699	0.7785	0.6721	0.7056	0.4341
relemb_mlm_0_2	UAmsterdam	rerank	nnlm	0.2856	0.7677	0.6662	0.7056	0.4350
nlm-prfun-bert	NLM	fullrank	nnlm	0.3445	0.8603	0.6648	0.6927	0.4265
TUW-TK-Sparse	TU_Vienna	rerank	nn	0.3188	0.7970	0.6610	0.7056	0.4164
TUW-TK-2Layer	TU_Vienna	rerank	nn	0.3075	0.7654	0.6539	0.7056	0.4179
p_d2q_bm25	anserini	fullrank	nnlm	0.2757	0.7326	0.6187	0.8035	0.4074
p_d2q_bm25rm3	anserini	fullrank	nnlm	0.2848	0.7424	0.6172	0.8391	0.4295
bert_6	UAmsterdam	rerank	nnlm	0.3240	0.7386	0.6149	0.7056	0.3760
CoRT-bm25	HSRM-LAVIS	fullrank	nnlm	0.2201	0.8372	0.5992	0.8072	0.3611
CoRT-standalone	HSRM-LAVIS	fullrank	nnlm	0.2412	0.8112	0.5926	0.6002	0.3308

Documents

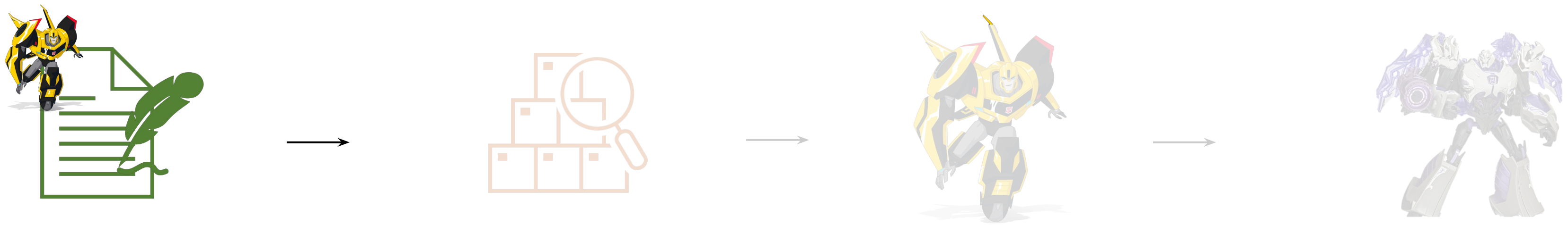
Documents: too long for these poor Transformers!



1. T5 *can* handle > 512 input tokens but pretrain/finetune step still use a maximum of 512 input tokens.
2. Running with arbitrarily long input sequences can be very computationally inefficient.

Our Solution: Sliding Window Segmentation! Use a window of size n_{length} sentences and a stride of n_{stride}
.

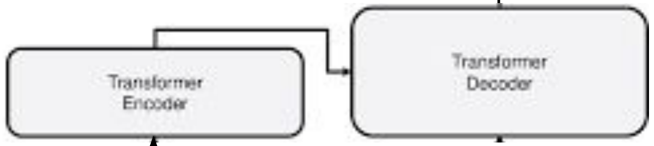
How did we rank: (H_{-1}) docT5query



H_{-1} H_0 H_1 H_2

<u>Generated Queries (N = 2):</u> $q_{i,1}$ $q_{i,2}$

top-k sampling (k=10)



e_i

<u>Segments:</u> $\mathbf{e_1}$: $s_1\ s_2\ s_3\ s_4$; $\mathbf{e_2}$: $s_3\ s_4\ s_5\ s_6$; $\mathbf{e_3}$: $s_5\ s_6\ s_7\ s_8$

Sliding Window Segmentation n_{length} = 4 with n_{stride} = 2

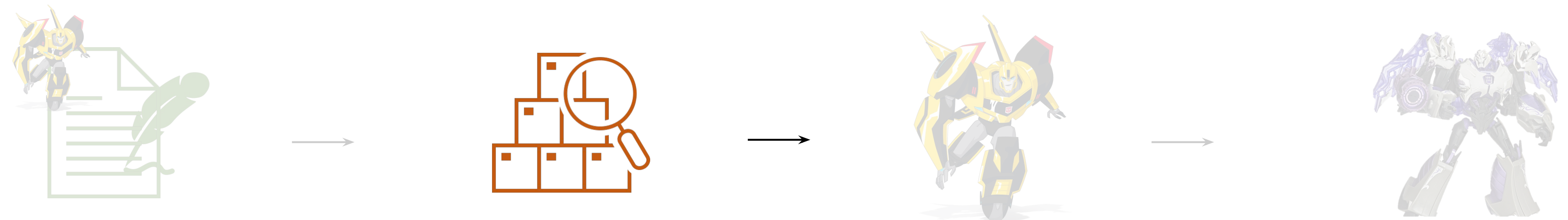
<u>Document:</u> $s_1\ s_2\ s_3\ s_4\ s_5\ s_6\ s_7\ s_8$ where s_j is sentence j.

<u>Expanded Document:</u>

$s_1\ s_2\ s_3\ s_4\ s_5\ s_6\ s_7\ s_8$ $q_{1,1}$ $q_{1,2}$ $q_{2,1}$ $q_{2,2}$ $q_{3,1}$ $q_{3,2}$

For the MS MARCO Document Corpus, we generate N = 10 queries per segment.
Sliding Window Segmentation is performed with maximum length of n_{length} = 10 and strides of n_{stride} = 5.

How did we rank: (H_0) Anserini Retriever



H_{-1} → H_0 → H_1 → H_2

TREC DL 2020 Queries

[MS MARCO Doc] Docs
(run$_1$: d_bm25)

or

[MS MARCO Doc] Expanded Docs
(run$_3$: d_d2q_bm25)

Indexing →

Inverted Index

Anserini Retriever

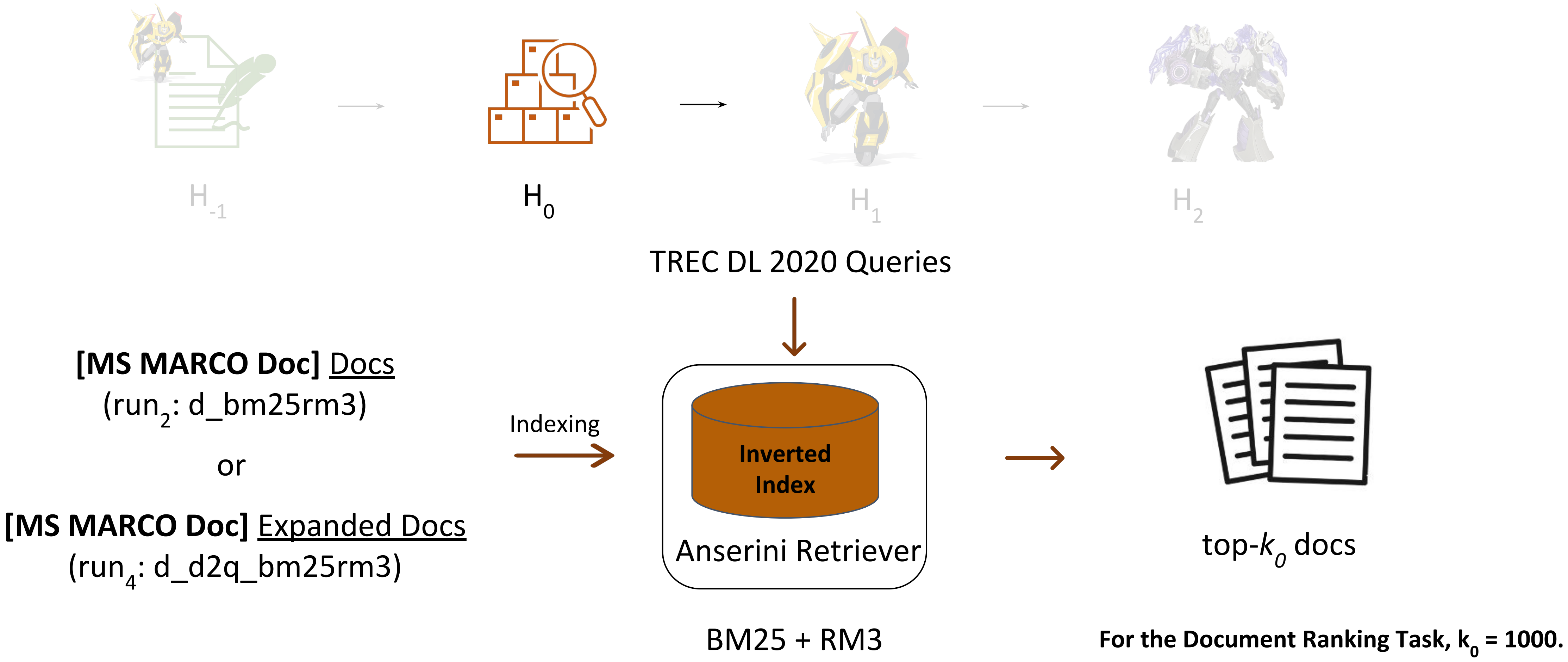
BM25

→

top-k_0 docs

For the Document Ranking Task, k_0 = 1000.

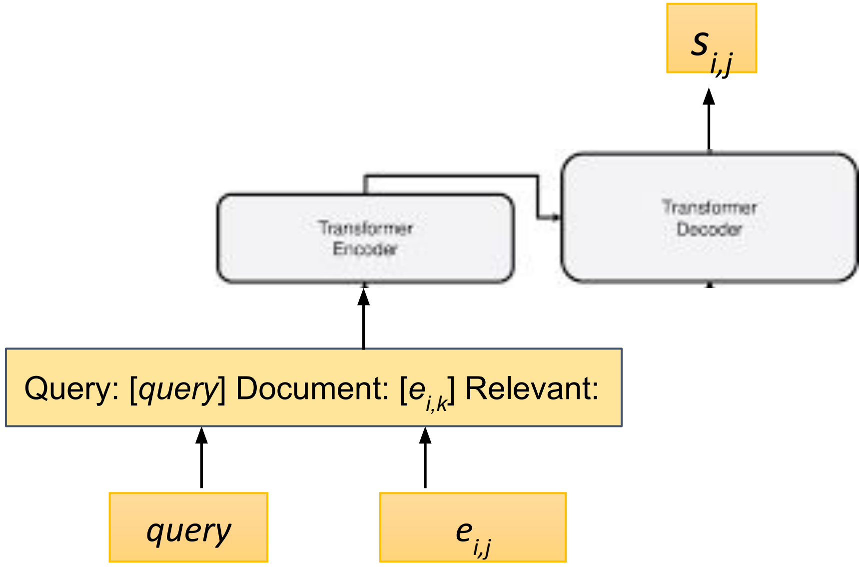
How did we rank: (H_0) Anserini Retriever



How did we rank: (H_1) monoT5

top-k_0 docs

segments of top-k_0 docs



$e_{i,j}$: *j*-th segment of document *i*

$s'_i = max_j s_{i,j}$

e'_i : highest scoring evidence segment

top-k_1 docs

For the Document Ranking Task, k_1 = 50.

How did we rank: (H_2) duoT5

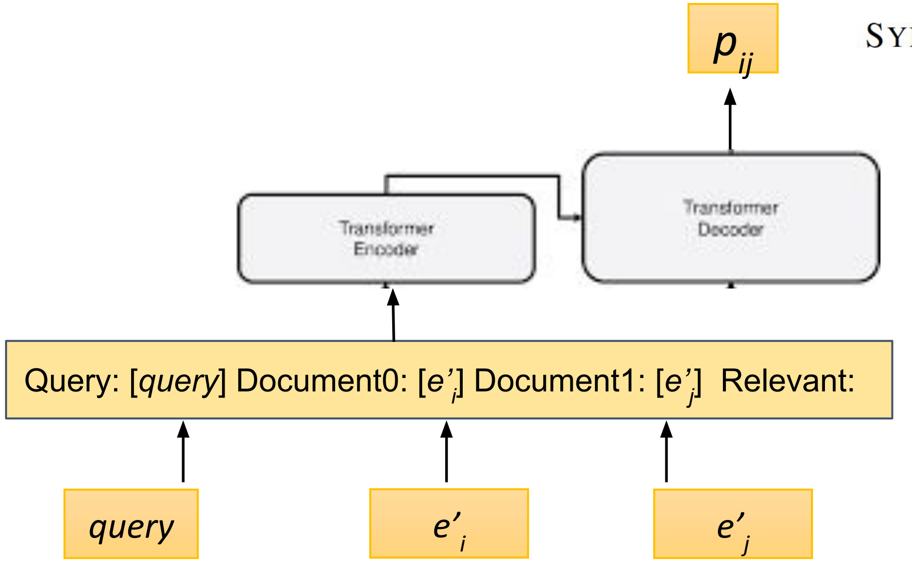
$$\text{SYM-SUM}: s_i = \sum_{j \in J_i} (p_{i,j} + (1 - p_{j,i}))$$

top-k_1 docs

Highest mono scoring segments of top-k_1 docs

e'_1	e'_2
e'_1	e'_3
e'_2	e'_1
e'_2	e'_3
e'_3	e'_1
e'_3	e'_2

$k_1(k_1\text{-}1)$ pairs



top-k_2 docs

Results

Run	AP	NDCG@10	NDCG@1k	RR	R@1k
(0) median	0.3902	0.5733	0.5859	0.9444	-
(1) d_bm25	0.3791	0.5271	0.5647	0.8521	0.6110
(2) d_bm25rm3	0.4006	0.5248	0.5726	0.8541	0.6392
(3) d_d2q_bm25	0.4230	0.5885	0.6115	0.9369	0.6412
(4) d_d2q_bm25rm3	0.4228	0.5407	0.5902	0.8147	**0.6555**
(5) d_bm25rm3_duo	0.5270	0.6794	0.6929	**0.9476**	-
(6) d_d2q_bm25_duo	0.5422	**0.6934**	0.7089	**0.9476**	-
(7) d_d2q_rm3_duo	**0.5427**	0.6900	**0.7122**	**0.9476**	-

Table 2: Results on TREC's Deep Learning 2020 Document Ranking track.

run	group	subtask	neural	RR (MS)	RR	NDCG@10	NCG@100	AP
d_d2q_duo	h2oloo	fullrank	nnlm	0.4451	0.9476	0.6934	0.7718	0.5422
d_d2q_rm3_duo	h2oloo	fullrank	nnlm	0.4541	0.9476	0.6900	0.7769	0.5427
d_rm3_duo	h2oloo	fullrank	nnlm	0.4547	0.9476	0.6794	0.7498	0.5270
ICIP_run1	ICIP	rerank	nnlm	0.3898	0.9630	0.6623	0.6283	0.4333
ICIP_run3	ICIP	rerank	nnlm	0.4479	0.9667	0.6528	0.6283	0.4360
fr_doc_roberta	BITEM	fullrank	nnlm	0.3943	0.9365	0.6404	0.6806	0.4423
ICIP_run2	ICIP	rerank	nnlm	0.4081	0.9407	0.6322	0.6283	0.4206
roberta-large	BITEM	rerank	nnlm	0.3782	0.9185	0.6295	0.6283	0.4199
bcai_bertb_docv	bcai	fullrank	nnlm	0.4102	0.9259	0.6278	0.6604	0.4308
ndrm3-orc-full	MSAI	fullrank	nn	0.4369	0.9444	0.6249	0.6764	0.4280
ndrm3-orc-re	MSAI	rerank	nn	0.4451	0.9241	0.6217	0.6283	0.4194
ndrm3-full	MSAI	fullrank	nn	0.4213	0.9333	0.6162	0.6626	0.4069
ndrm3-re	MSAI	rerank	nn	0.4258	0.9333	0.6162	0.6283	0.4122
ndrm1-re	MSAI	rerank	nn	0.4427	0.9333	0.6161	0.6283	0.4150
mpii_run2	mpii	rerank	nnlm	0.3228	0.8833	0.6135	0.6283	0.4205
bigIR-DTH-T5-R	QU	rerank	nnlm	0.3235	0.9119	0.6031	0.6283	0.3936
mpii_run1	mpii	rerank	nnlm	0.3503	0.9000	0.6017	0.6283	0.4030
ndrm1-full	MSAI	fullrank	nn	0.4350	0.9333	0.5991	0.6280	0.3858
uob_runid3	UoB	rerank	nnlm	0.3294	0.9259	0.5949	0.6283	0.3948
bigIR-DTH-T5-F	QU	fullrank	nnlm	0.3184	0.8916	0.5907	0.6669	0.4259
d_d2q_bm25	anserini	fullrank	nnlm	0.3338	0.9369	0.5885	0.6752	0.4230
TUW-TKL-2k	TU_Vienna	rerank	nn	0.3683	0.9296	0.5852	0.6283	0.3810
bigIR-DH-T5-R	QU	rerank	nnlm	0.2877	0.8889	0.5846	0.6283	0.3842
uob_runid2	UoB	rerank	nnlm	0.3534	0.9100	0.5830	0.6283	0.3976
uogTrQCBMP	UoGTr	fullrank	nnlm	0.3521	0.8722	0.5791	0.6034	0.3752
uob_runid1	UoB	rerank	nnlm	0.3124	0.8852	0.5781	0.6283	0.3786
TUW-TKL-4k	TU_Vienna	rerank	nn	0.4097	0.9185	0.5749	0.6283	0.3749
bigIR-DH-T5-F	QU	fullrank	nnlm	0.2704	0.8902	0.5734	0.6669	0.4177

PyGaggle

- You too can replicate our results!
- Gaggle of Deep Neural Architectures for Text Ranking and Question Answering.
- Support for MS MARCO Passage/Document Retrieval as well as TREC-COVID.
- Find us at pygaggle.ai!

Thank you!