Enhancing Text Retrieval Pipelines

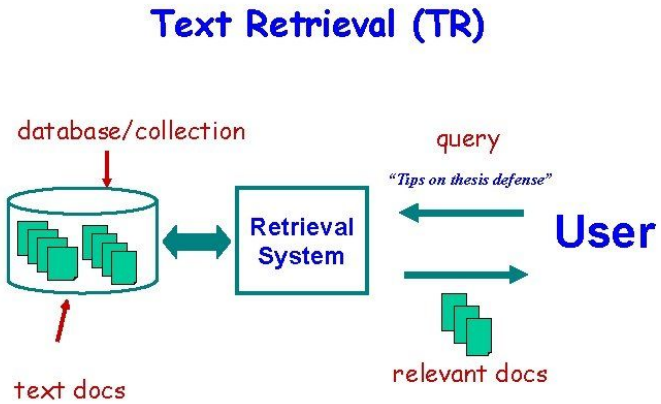
Sameer Singh Rawat

code - https://github.com/SameerR007/contrastive_learning

Introduction

What is retrieval?

Retrieval refers to fetching similar documents compared to the query, from a large collection of documents.



Use cases - web search, retrieval augmented generation (RAG)

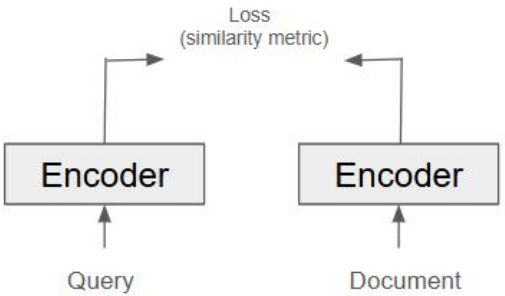
word retrieval will be used for text retrieval unless explicitly mentioned

Types of retriever based on architecture

- Bi-encoder based - In this the query and the document are encoded separately and a similarity score is calculated based on these two embeddings.
- Cross-encoder based - In this the query and the document are combined and a single joint embedding is calculated which is then passed onto a linear layer to predict the similarity score.
- Bi-encoders are more efficient for large scale retrieval hence are used in retrievers while cross-encoders are more precise making them effective in reranking.
- We will only be looking into bi-encoder ones.

Components of retriever architecture

1. Encoder - encodes query and document.
2. Similarity metric - metric with which similarity is calculated between query and document embedding.
3. Loss function - loss function used to optimize the embeddings by updating encoder weights.



Approach - In this paper we experimented with different combination of these components to see their effect on retrieval.

Methods

Overview (1)

Based on different combinations of retriever components we trained three different models.

Model	**Embeddings used**	**Similarity/Distance**	**Loss function**
Baseline	BERT CLS token	cosine	infoNCE
Average	BERT final layer average	cosine	infoNCE
Hyperbolic	Hyperbolic	Lorentz	infoNCE+ α*entailment loss

Baseline model (1)

Embedding used - BERT CLS token

Similarity metric used - cosine similarity

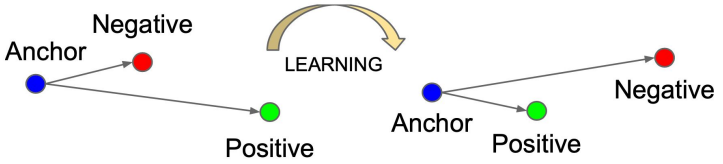
Loss function used - infoNCE loss

Baseline model (2)

infoNCE loss

$$\mathcal{L}(Q, D^+, \{D_N\}) = -\log p(D = D^+ \mid Q) =$$

$$-\log \frac{\exp(\mathrm{Sim}(Q, D^+))}{\exp(\mathrm{Sim}(Q, D^+)) + \sum_{D_i^- \in \{D_N\}} \exp(\mathrm{Sim}(Q, D_i^-))}$$



where, D^+ represents the document that is similar to the query (positive) and $\{D_N\}$ is the set of negative documents that are not similar to the query (negatives)

Average model (1)

Embedding used - Average of BERT final layer tokens

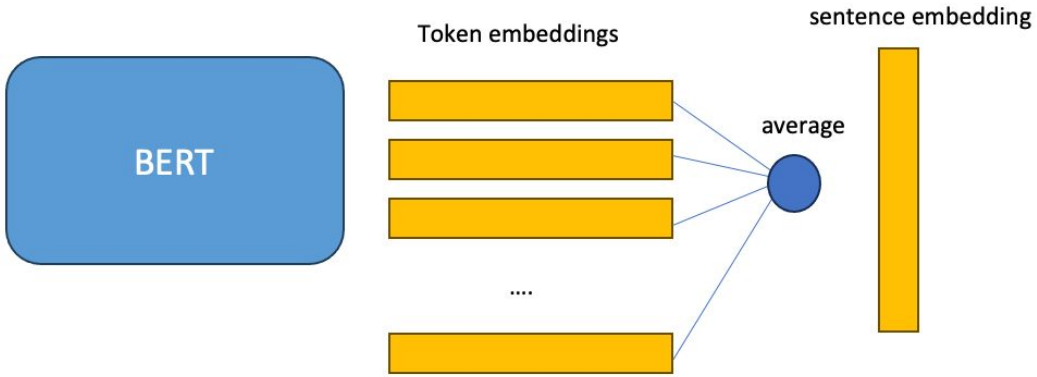
Similarity metric used - cosine similarity

Loss function used - infoNCE loss

Note - same as baseline model except for embeddings used.

Average model (2)

Average of BERT final layer tokens



Hyperbolic model (1)

Embedding used - Hyperbolic embedding

Similarity metric used - negation of lorentz distance

Loss function used - infoNCE loss + α * entailment loss

Hyperbolic model (2)

Hyperbolic embeddings

$$\mathbf{x}_{space} = \frac{sinh(\sqrt{c}\,\|\mathbf{v}_{space}\|)}{\sqrt{c}\,\|\mathbf{v}_{space}\|}\mathbf{v}_{space}$$

where, v_{space} is CLS token embedding from BERT and this is projected to hyperbolic space resulting in x_{space}

Hyperbolic model (3)

Lorentz distance

$$x_{time} = \sqrt{1/c + \|\mathbf{x}_{space}\|^2}$$

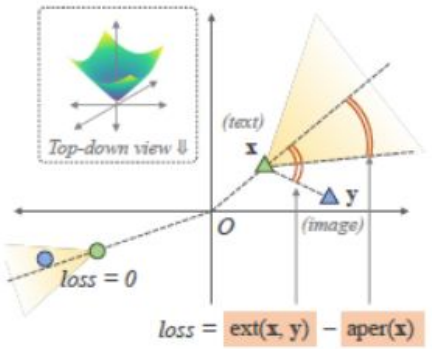
$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} = \langle \mathbf{x}_{space}, \mathbf{y}_{space} \rangle - x_{time}\, y_{time}$$

$$d_{\mathcal{L}}(\mathbf{x}, \mathbf{y}) = \sqrt{1/c} \cdot \cosh^{-1}(-c\, \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}})$$

where,
$\langle x,y \rangle_L$ is inner product,
$d_L(x,y)$ is lorentz distance, and
c is a hyperparameter

Hyperbolic model (4)

Entailment loss - It enforces a hierarchical relationship between the query and the positive document (see figure below, from Desai et al., 2023).



Entailment loss: This loss pushes image embedding y inside an imaginary cone projected by the paired text embedding x, and is implemented as the difference of exterior angle and half aperture of the cone. Loss is zero if the image embedding is already inside the cone (left quadrant).

Evaluation

Implementation details (1)

Data - trained on MS macro dataset

Positives per query - 1

Negatives per query - 5

Total epochs - 3

Batch size - 32

α = 0.1 (?)

LoRA fine-tuning (prevents overfitting)

Evaluation metric used - Mean Reciprocal rank (MRR)

*alpha is the entailment loss weightage in hyperbolic model loss where loss = infoNCE + α * Entailment loss*

Implementation details (2)

Choice of α

We perform training and evaluation on four datasets with α = 0.1, 0.5 and 0.9

	α		
	0.1	0.5	0.9
MS MACRO (v1.1)	**0.5333**	0.5288	0.4648
MS MACRO (v2.1)	**0.6435**	0.623	0.61
arXiv	**0.8582**	0.7921	0.2755
dbpedia14	**0.9977**	0.4101	0.4093

MRR dips as we increase the value of α. This can suggest that in hyperbolic model, our primary minimization objective should be contrastive loss with only a small weightage given to the minimization of entailment loss.

Results (1)

Results on MS macro dev set and BEIR dataset (OOD)

Dataset	Model	MRR	minutes/epoch
MS MACRO	Baseline	0.5448	33
	Average	**0.5581**	33
	Hyperbolic	0.5333	33

BEIR	Baseline	Average	Hyperbolic
arguana	0.5909	**0.6639**	0.4428
climate fever	0.4005	**0.5892**	0.4097
fever	0.563	**0.7188**	0.4776
fiqa	0.566	**0.6015**	0.5039
hotpotqa	0.44	**0.5779**	0.4051
NFcorpus	0.4281	**0.6634**	0.4483
NQ	0.6282	**0.6984**	0.5166
QUORA	0.895	**0.9421**	0.6887
SCIDOCS	0.4049	**0.4054**	0.3976
SciFact	0.5417	**0.5988**	0.4319
trec-covid	0.4547	**0.6639**	0.4367
touche2020	0.417	**0.535**	0.3997
dbpedia	0.5767	**0.582**	0.4113
Mean	0.5313	**0.6339**	0.4592

Average model performs well as compared to the other models, so letting all the tokens in the last hidden layer represent the document leads to substantially better results.

Results (2)

Corrupted Queries - Evaluation in real word scenario

A query might not be precisely written by the user due to several reasons such as typo, vagueness, language barrier.

It is essential to test retrieval under such circumstances.

Method - We corrupt queries by randomly choosing its words and injecting typos in them or replacing them with synonyms. Following this we evaluate the model's performance.

Results (3)

Corrupted Queries - Evaluation on MS macro dev set

n	0	1	2	3
Baseline	0.5448	0.5373	0.529	0.5173
Average	**0.5581**	**0.551**	**0.5422**	0.5267
Hyperbolic	0.5333	0.5332	0.5332	**0.5332**

here, n are the number of words corrupted in the query

Two key take away -

1. Performance of models declines with increase in corrupt words
2. Performance of hyperbolic model is more robust with corrupt data

Results (4)

Corrupted Queries - Evaluation on BEIR dataset

for p=0.5				
BEIR	Baseline	Average	Hyperbolic	median word count
arguana	0.5843	0.605	0.4254	174
climate fever	0.3707	0.5374	0.3923	19
fever	0.493	0.5976	0.4357	8
fiqa	0.5297	0.5387	0.4629	10
hotpotqa	0.4028	0.4846	0.3861	15
NFcorpus	**0.4322**	**0.446**	**0.4732**	**2**
NQ	0.5849	0.6545	0.5039	9
QUORA	0.8151	0.8589	0.6037	9
SCIDOCS	**0.3952**	**0.3895**	**0.4117**	**9**
SciFact	0.5023	0.5581	0.4474	12
trec-covid	0.488	0.5247	0.3907	10
touche2020	0.4272	0.5163	0.3626	6
dbpedia	0.5123	0.5442	0.4306	5

BEIR	Baseline	Average	Hyperbolic
Mean % MRR decrease	4.7529	11.5903	3.6162

1. Hyperbolic model can outperform for certain datasets in noisy settings
2. Mean percentage decrease in MRR with and without corruption across all 13 datasets is most for average and least for hyperbolic model (?)

p=0.5 denotes randomly chosen half of the words (except stopwords) are corrupted

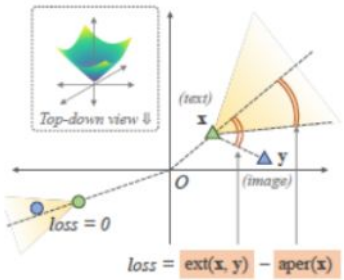
Results (5)

Corrupted Queries - Why average model performs bad and hyperbolic model performs better

- In average model all the tokens (even the corrupted ones) have a say in representation.
- Corruption makes query more general => General queries are positioned closer to origin in hyperbolic space=> more chance of positive to be inside the cone

Eg - Temperature of California? (specific)

Temperature of xxx? (general)



Discussion

We derive three insights from this work -

1. In hyperbolic model loss optimization only small weightage should be given to entailment loss with contrastive learning as primary objective.

2. For clean queries, taking into account all the embeddings to represent the document performs better than relying solely on CLS token for representation, without any tradeoff in training time.

3. For noisy settings, hyperbolic model can outperform the euclidean based models since it can utilize its hierarchical learning to capture vague and general concepts better.

Further works can include - training on corrupt data, analyzing how much of each corruption factor like typos, synonym replacement contribute to MRR decrease.

References

1. Ma, X., Wang, L., Yang, N., Wei, F., & Lin, J. (2023). Fine-tuning LLaMA for multi-stage text retrieval. arXiv. https://arxiv.org/abs/2310.08319
2. Lingling Xu, Haoran Xie, Zongxi Li, Fu Lee Wang, Weiming Wang, and Qing Li. 2023. Contrastive Learning Models for Sentence Representations. ACM Trans. Intell. Syst. Technol. 14, 4, Article 67 (August 2023), 34 pages. https://doi.org/10.1145/3593590
3. Desai, K., Nickel, M., Rajpurohit, T., Johnson, J., & Vedantam, R. (2024). Hyperbolic image-text representations. arXiv. https://arxiv.org/abs/2304.09172
4. Pal, A., van Spengler, M., D'Amely di Melendugno, G. M., Flaborea, A., Galasso, F., & Mettes, P. (2024). Compositional entailment learning for hyperbolic vision-language models. arXiv. https://arxiv.org/abs/2410.06912
5. Desai, U., Tamilselvam, S., Kaur, J., Mani, S., & Khare, S. (2020). Benchmarking popular classification models' robustness to random and targeted corruptions. arXiv. https://arxiv.org/abs/2002.00754

Thank you!