

Injecting the BM25
Score as Text
Improves BERT-
Based Re-rankers



The basics

Cross-Encoder

Architecture classique d'un Cross-Encoder

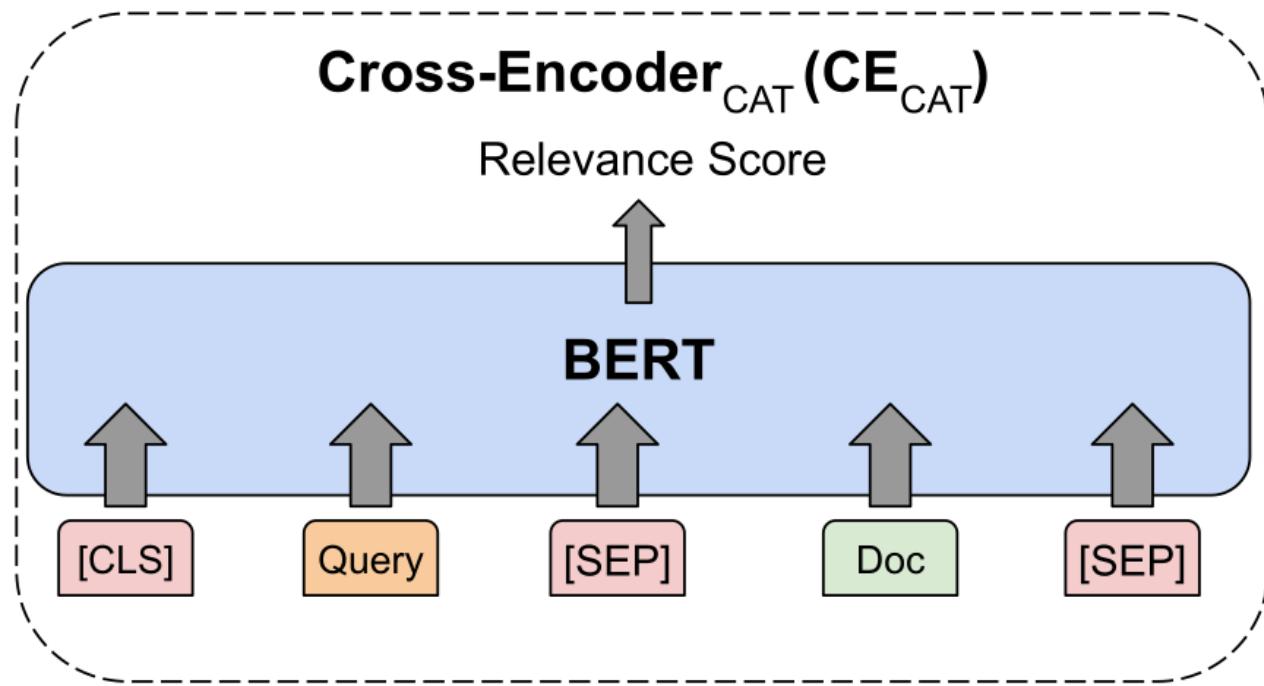


Fig. 1. Regular cross-encoder input

Re-Ranking

- Ré-ordonner les résultats
- Pipeline classique :
 - Premier rank sur tout le corpus avec BM25
→ Top 1000 de documents (pour chaque question)
 - Re-rank des 1000 documents avec un Cross-Encoder
- C'est ce qui marche le mieux actuellement
 - MRR@10 : 39.02 % (MS Marco Dev)
 - nDCG@10 : 74.31 % (TRECL DL 19)

Notre article

Et si on réutilise le score BM25 ?

- La méthode classique : combinaison linéaire du score CE et BM25
→ Peut réduire les performances...
- Plus avancé : on combine des modèles... (Augmented SBERT : Cross-Encoder + Bi-Encoder)
- ✨ Notre méthode ✨

Méthode proposée

Les modèles BERT savent capturer les chiffres

⇒ Et si on lui injecte directement le score ?

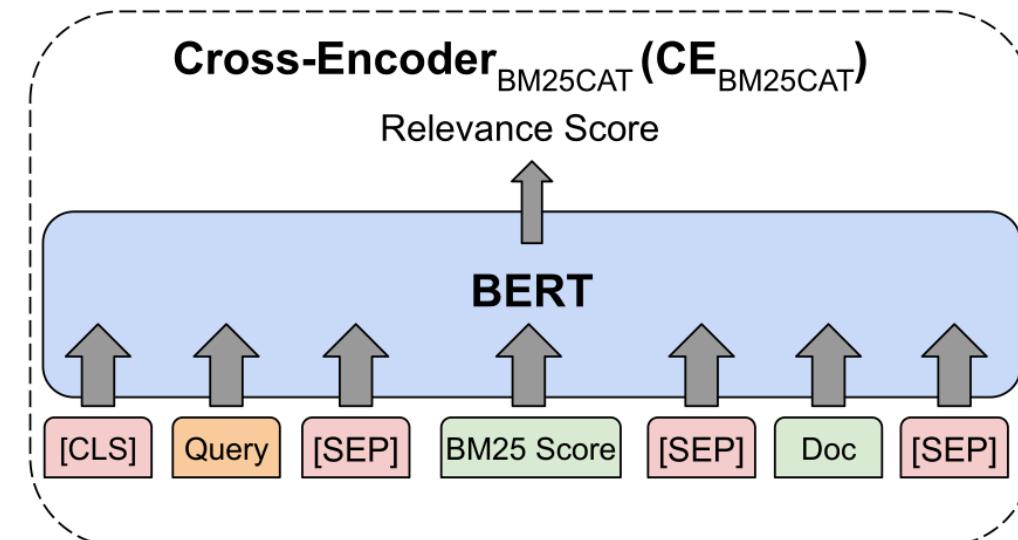


Fig. 2. Injection of BM25 in input

Problèmes :

1. Est-ce que ça marche ? À quel point ?
2. Et par rapport aux autres méthodes de combinaison de score classique ?

Méthode

Méthode : Injection du score

- Le score BM25 est **non borné**, besoin de **normaliser** pour garder l'interprétation :
 - $Min - Max(S_{BM25}) = \frac{S_{BM25} - s_{min}}{s_{max} - s_{min}}$
 - $Standard(S_{BM25}) = \frac{S_{BM25} - \mu(S)}{\sigma(S)}$
 - $Sum(S_{BM25}) = \frac{S_{BM25}}{sum(S)}$
- Deux manières de choisir les paramètres $s_{min}, s_{max}, \sigma(S), \mu(S)$:
 - Local : calcul sur la "ranked list of scores per query"
 - Global : Respectivement $\{0, 50, 42, 6\}$

Méthode : Injection du score

- Les modèles BERT ont des difficultés à interpréter les flottants
 - ⇒ Conversion en nombre entier
 - ⇒ Arrondi des flottants

Méthode : Combinaison du score BM25 et du CE

- Grosse littérature sur comment combiner les scores
- Méthodes linéaires et non linéaires :
 - Somme : $s_{BM25} + s_{CE}$
 - Max : $\max(s_{BM25}, s_{CE})$
 - Moyenne pondérée : $\alpha s_{BM25} + (1 - \alpha) s_{CE}$

Méthode : protocole d'évaluation

- Évaluation sur :
 - MS Marco Dev (7000 queries)
 - TREC DL 19
 - TREC DL 20
- Métriques :
 - MRR@10
 - nDCG@10
 - MAP
 - Ajout de notre part : Rappel@1000

Résultats

Effet de la normalisation sur le score

Normalization	Local/Global	Float/Integer	MSMARCO DEV		
			nDCG@10	MAP	MRR@10
(a) MiniLM _{CAT} (without injecting BM25 score)			.419	.363	.360
(b) Original Score	—	—	.420	.364	.362
(c) Min-Max	Local	Float	.411	.359	.354
(d) Min-Max	Local	Integer	.414	.361	.355
(e) Min-Max	Global	Float	.422	.365	.363
(f) Min-Max	Global	Integer	.424†	.368†	.367†
(g) Standard	Local	Float	.407	.355	.352
(h) Standard	Local	Integer	.410	.358	.354
(i) Standard	Global	Float	.420	.363	.361
(j) Standard	Global	Integer	.421	.365	.363
(k) Sum	-	Float	.402	.349	.338
(l) Sum	-	Integer	.405	.350	.342

Cross-Encoder classique vs ✨ le notre ✨

Model	TREC DL 20		TREC DL 19		MSMARCO DEV		
	nDCG@10	MAP	nDCG@10	MAP	nDCG@10	MAP	MRR@10
BM25	.480	.286	.506	.377	.234	.195	.187
Re-rankers							
BERT-BaseCAT	.689	.447	.713	.441	.399	.346	.342
BERT-BaseBM25CAT	.705†	.475†	.723†	.453†	.422†	.367†	.364†
BERT-LargeCAT	.695	.464	.714	.467	.401	.344	.360
BERT-LargeBM25CAT	.728†	.482†	.731†	.477†	.424†	.367†	.369†
DistilBERT _{CAT}	.670	.442	.679	.440	.383	.310	.325
DistilBERT _{BM25CAT}	.682†	.456†	.699†	.451†	.390†	.323†	.339†
MiniLM _{CAT}	.681	.448	.704	.452	.419	.363	.360
MiniLM _{BM25CAT}	.710†	.473†	.711†	.463†	.424†	.368†	.367†

Comparaison avec les combinaisons linéaires classiques

Model	Ensemble	MSMARCO DEV		
		nDCG@10	MAP	MRR@10
BM25	—	.234	.195	.187
BERT-Base _{CAT}	—	.399	.346	.342
BM25 and BERT-Base _{CAT}	Sum	.270	.225	.218
BM25 and BERT-Base _{CAT}	Max	.237	.197	.190
BM25 and BERT-Base _{CAT}	Weighted-Sum (tuned)	.353	.295	.290
BM25 and BERT-Base _{CAT}	Naive Bayes	.314	.260	.254
BERT-Base _{BM25CAT}	BM25 Score Injection	.422	.367	.364

Explicabilité du modèle

Query [SEP] BM25 [SEP] Passage	Label	Model: Rank
[CLS] what is the shingles jab ? [SEP] 22 [SEP] the shingles vaccine . the vaccine , called zostavax , is given as a single injection under the skin (subcutaneously) . it can be given at any time in the year . unlike with the flu jab	R	BM25: 3 BERT _{BM25CAT} : 1 BERT _{CAT} : 104
[CLS] what is the shingles jab ? [SEP] 11 [SEP] shingle is a corruption of german schindle (schindel) meaning a roofing slate . shingles historically were called tiles and shingle was a term applied to wood shingles , as is still mostly the case outside the us [SEP]	N	BM25: 146 BERT _{BM25CAT} : 69 BERT _{CAT} : 1

Nos résultats...

	MRR@10	MAP	NDCG@10	Recall@1000
BM25	.187	.277	.234	.808
Baseline	.368	.371	.478	.857
Cross-Encoder	.254	.261	.380	.857

Sûrement un problème dans l'apprentissage et de tuning de paramètres... 😔

Forces et faiblesses de la contribution

- Injection du score : Facile et efficace
- Très lent (de base, il faut ici ajouter le calcul du score pour l'entraînement)...
- Possible sur n'importe quel Cross-Encoder

Nos remarques

- Matériel nécessaire et temps de train des Cross Encoder assez conséquent
- Classe Cross-Encoder custom de celle de Sentence Transformer :
 - Boucle de fit qui log les loss (avec WandB 😊)
 - Tokenizer pour input [CLS] query [SEP] BM25 [SEP] passage [SEP]
- Les challenges de ce projet :
 - Comprendre et utiliser les datasets
 - Trouver une manière efficace de calculer BM25 sur le corpus
 - ~~À la main ; Elastic Search Docker ; Elastic Search Cloud ; Pyserini sur ppti-gpu~~
 - Pyserini en local → écriture du tsv → Transfert et lecture du ppti-gpu

Merci

Avez vous des questions ?

Notre ré-implémentation

Code - Pyserini

Utilisation de [Pyserini](#) pour l'indexation, le calcul de BM25 et la récupération des documents.

⇒ Utilisation de plusieurs scripts `bash` :

- `prep.sh` qui convertit les documents de MSMARCO-Passage en `.json` pour l'indexation.
- `index.sh` pour indexer MSMARCO-Passage (8.8 millions de documents) (~ 5 minutes sur une bonne machine avec SSD).
- `baseline.sh` pour vérifier et reproduire les résultats de l'état de l'art (retrieval sur le dev).

Code - Transformation des données

Pour pouvoir entraîner notre Cross-Encoder, on a besoin de calculer le score BM25 pour chaque paire (query, passage). Cela se fait *relativement* rapidement après avoir créé un index.

- `download_dataset.py` : explicite.
- `create_dataset.py` :
 - Création du jeu d'entraînement avec un positive-to-negative ratio fixé à 4 (paramètre). ~45 minutes pour 20 millions de couples (query, passage).
 - Création du jeu d'évaluation à partir d'un fichier BM25 retrieval de Pyserini (format `msmarco`). ~15 minutes pour le top 1000 des 6980 queries du `dev.small` .
- `load_dataset.py` : permet de charger le corpus, les queries, les données de train et d'eval d'une manière compréhensible par Python.

Code - Entraînement du Cross-Encoder

Création d'une classe Cross-Encoder custom, basé sur la classe `CrossEncoder` de la librairie `SentenceTransformers`. La classe est dans `crossencoder_bm25.py`.

Adaptation de la boucle d'apprentissage : ajout d'un early-stopping, ajout d'un callback (pour log), calcul d'un MRR@10 pendant le train.

Adapation du tokenizer pour le train et la prédiction : sinon, on n'a pas `[CLS]` `query [SEP] score [SEP] passage [SEP]`.

Ensuite on charge nos données avec les fonctions précédentes et on attend ~13 heures le temps d'apprentissage du modèle (pour une époque).

Code - Evaluation du Cross-Encoder

Et enfin, on peut évaluer. Il faut donc charger les données retrieve par BM25 puis prédire avec le Cross-Encoder (~2 heures pour 6980 queries TOP 1000), avant de réordonner le tout puis calculer les métriques (MRR@10, MAP, nDCG, PR) avec Pyserini avec le script `eval.sh`. On calcule ces métriques en créant un fichier sous format `microsoft` à partir des prédictions.

L'évaluation est disponible dans un notebook Jupyter `eval_bm25.ipynb`. Il y a le même notebook `eval_bm25_baseline.ipynb` pour la baseline sur un Cross-Encoder classique pré-entraîné par nos soins également.

Il y a également un notebook `eval_bm25_comparison.ipynb` pour comparer le modèle avec un modèle où l'on retrieve d'abord avec un Bi-Encoder puis qu'on re-rank avec notre Cross-Encoder custom.