Actes de CORIA-TALN-RJCRI-RECITAL 2025
Marseille, France, 30 juin au 4 juillet 2025

UC-FIRe: Approche efficace pour la recherche d'informations non supervisée

Maxime Hanus Quentin Guignard Christophe Rodrigues
Léonard De Vinci Pôle Universitaire, Research Center, 92916 Paris la Défense, France
maxime.hanus@edu.devinci.fr, quentin.guignard@edu.devinci.fr,
christophe.rodrigues@devinci.fr

RÉSUMÉ

Nous présentons un modèle de recherche d'informations non supervisé conciliant efficacité et faible coût computationnel, fonctionnant uniquement sur CPU. Plutôt que de remplacer BM25, nous l'améliorons en réduisant l'écart lexical. Notre méthode repose sur l'entraînement de vecteurs de mots FastText et la construction de matrices de coexistence et de similarité pour regrouper des mots interchangeables en clusters. Documents et requêtes sont réécrits avec ces clusters, améliorant la pertinence des résultats sans alourdir l'inférence. Expérimenté sur plusieurs corpus de BEIR, notre modèle surpasse des approches plus coûteuses en calcul et obtient de meilleures performances que BM25 sur diverses métriques, tout en conservant une vitesse d'inférence similaire. Cette recherche démontre que notre méthode offre une alternative pratique, scalable et économique aux modèles denses et hybrides, facilitant son adoption dans des systèmes de recherche réels. UC-FIRe est disponible publiquement : https://github.com/Limekaaa/UC-FIRe.

ABSTRACT

UC-FIRe, Unsupervised, Cost-Effective and Fast model for Information Retrieval

This paper introduces UC-FIRe, a novel unsupervised retrieval model designed to balance efficiency and effectiveness while operating exclusively on CPUs. Unlike many existing methods that rely on extensive supervision or incur high computational costs, our approach provides a cost-effective alternative with strong performance on the BEIR benchmark. Instead of replacing BM25, we aim to enhance it, leveraging its robustness by reducing the lexical gap. Our method preprocesses text by training FastText on the target corpus to obtain word vectors, constructing coexistence and similarity matrices to identify interchangeable words, and forming word clusters. By rewriting documents and queries using these clusters, we improve BM25's ability to capture semantic relationships. Extensive experiments on multiple BEIR corpora demonstrate that UC-FIRe outperforms more computationally expensive retrieval models while maintaining an inference speed comparable to BM25. Moreover, our model consistently surpasses BM25 across various evaluation metrics, highlighting its potential as a practical and scalable alternative for real-world information retrieval applications. UC-FIRe is publicly available : https://github.com/Limekaaa/UC-FIRe.

MOTS-CLÉS : Recherche d'informations non supervisée, Recherche d'informations, réduction de l'écart lexical, amélioration de BM25, plongements de mots, regroupement de mots, modèles de recherche efficient.

KEYWORDS: Unsupervised Information Retrieval, Information Retrieval, Lexical Gap Reduction, Lexical Gap, BM25 Enhancement Embeddings, Word Clustering, Cost-effective Retrieval Models.

Actes de la 20e Conférence en Recherche d'Information et Applications (CORIA), pages 249–264.

1 Introduction

La recherche d'informations (RI) efficace et précise est fondamentale pour les applications modernes basées sur les données, permettant aux utilisateurs de localiser des informations pertinentes à partir de vastes dépôts textuels. Les méthodes traditionnelles de recherche lexicale, telles que BM25 (Robertson & Jones, 1976), sont restées populaires en raison de leur efficacité et de leur robustesse. Cependant, elles peinent à gérer l'écart lexical entre les termes utilisés dans les requêtes et ceux présents dans les documents, un problème bien connu en RI (Furnas *et al.*, 1987; Crestani, 2000; Zhao & Callan, 2010), car elles ne reconnaissent pas les termes sémantiquement proches mais lexicalement différents comme équivalents. Cela limite leur efficacité dans des contextes où les reformulations ou paraphrases sont fréquentes.

Plusieurs approches ont été proposées pour atténuer cet écart lexical, notamment l'expansion de requêtes ou de documents (Metzler *et al.*, 2023), l'exploitation de relations sémantiques (Almasri, 2017), ou encore l'utilisation de plongements distribués (Zhuang & Zuccon, 2021). Ces méthodes apportent des améliorations notables, mais impliquent souvent des coûts de calcul additionnels ou requièrent des ressources annotées.

Les modèles de récupération neuronaux plus avancés, tels que Dense Passage Retrieval (DPR) (Karpukhin *et al.*, 2020), ColBERT (Khattab & Zaharia, 2020) et Contriever (Izacard *et al.*, 2022), surmontent les écarts lexicaux en codant les requêtes et les documents dans des espaces vectoriels denses, permettant une correspondance sémantique. Ces méthodes entraînent des coûts de calcul élevés à la fois pour l'entraînement et l'inférence, nécessitant de grandes quantités de données labellisées. Les approches hybrides, telles que SPLADE (Formal *et al.*, 2021), visent à équilibrer efficacité et efficience en intégrant des signaux lexicaux parcimonieux avec des plongements denses, mais elles nécessitent encore de lourds coûts à l'inférence.

Dans ce travail, nous proposons un nouveau modèle de recherche non supervisé qui améliore BM25 en traitant ses limitations lexicales tout en maintenant son efficacité. Notre approche introduit une étape de prétraitement où nous entraînons des vecteurs de mots FastText (Grave *et al.*, 2018; Bojanowski *et al.*, 2016; Mikolov *et al.*, 2013) et construisons des matrices de coexistence et de similarité pour regrouper les mots interchangeables. En réécrivant les documents et les requêtes à l'aide de ces clusters, nous réduisons l'écart lexical, améliorant ainsi l'efficacité de la récupération sans ajouter de surcharge d'inférence significative.

Contrairement aux modèles de récupération denses, notre méthode ne nécessite pas d'inférence neuronale coûteuse, la rendant ainsi efficace sur le plan computationnel. De plus, comparée aux modèles hybrides qui s'appuient sur des distributions de termes apprises, notre approche est entièrement non supervisée et ne nécessite pas de données labellisées. Des expériences approfondies sur plusieurs jeux de données du benchmark BEIR (Thakur *et al.*, 2021) démontrent que notre modèle surpasse des méthodes plus coûteuses en termes de calcul tout en préservant la rapidité d'inférence de BM25. En offrant une alternative évolutive et interprétable aux techniques de RI existantes, notre approche présente une solution convaincante pour les applications du monde réel nécessitant à la fois efficacité et précision.

Cet article est structuré comme suit : la section 2 fournit un aperçu de l'état actuel des travaux en Recherche d'Informations. La section 3 détaille notre méthode proposée, y compris les étapes de prétraitement, la construction des matrices de coexistence et de similarité, ainsi que le processus de clustering. La section 4 décrit la configuration expérimentale et présente les résultats, comparant

les performances de notre modèle aux benchmarks établis. La section 5 mène des études d'ablation, analysant l'impact de différents paramètres. Enfin, la section 6 discute des implications de nos résultats et des directions potentielles pour les travaux futurs, et conclut l'article avec un résumé de nos contributions et de la signification de notre approche.

À travers cette recherche, nous visons à contribuer à l'avancement des techniques de récupération d'informations, en offrant une solution pratique et efficace qui peut être facilement adoptée dans diverses applications.

2 Travaux Connexes

La RI est essentielle pour permettre aux utilisateurs d'extraire des documents pertinents à partir de vastes corpus textuels. Les méthodes traditionnelles de RI reposent sur des techniques statistiques basées sur la correspondance de termes, tandis que les avancées récentes exploitent l'apprentissage profond pour améliorer la compréhension sémantique. Malgré le succès des modèles neuronaux, leur coût computationnel élevé et leur dépendance à de grandes quantités de données d'entraînement posent des défis majeurs. Cette section passe en revue les approches existantes en RI, en les classant en modèles lexicaux classiques, méthodes de recherche dense et techniques hybrides ou basées sur le clustering.

2.1 Modèles Lexicaux et Probabilistes

Les modèles lexicaux et probabilistes ont historiquement dominé la RI en raison de leur simplicité, de leur efficacité et de leur faible coût computationnel. Parmi eux, TF-IDF (Ramos, 2003) et BM25 (Robertson & Jones, 1976) restent des fonctions de classement largement utilisées, reposant sur le cadre probabiliste de pertinence. Ces modèles estiment la pertinence d'un document en prenant en compte la fréquence des termes, leur fréquence inverse dans le corpus, ainsi qu'une normalisation basée sur la longueur du document.

Malgré leurs avantages, les modèles lexicaux souffrent de leur dépendance à la correspondance exacte des mots-clés, ce qui limite leur capacité à récupérer du contenu sémantiquement similaire. Par exemple, une requête contenant "automobile" pourrait ne pas récupérer un document traitant de "voiture", bien que les deux termes soient sémantiquement proches. Ce problème "d'écart lexical" a conduit les chercheurs à explorer des techniques d'expansion de requêtes.

Les techniques d'expansion de requêtes visent à enrichir les requêtes des utilisateurs avec des termes sémantiquement liés afin d'améliorer les performances de récupération. Les approches traditionnelles, telles que le pseudo-relevance feedback, exploitent les documents initialement récupérés pour extraire des termes supplémentaires en vue d'un affinement de la requête (Metzler *et al.*, 2023). Des méthodes plus récentes utilisent les grands modèles de langage (LLMs) pour générer dynamiquement des reformulations de requêtes, améliorant ainsi la récupération dans les cas où les méthodes basées sur des mots-clés échouent (Gao *et al.*, 2022).

Une alternative à l'expansion de requêtes est l'expansion de documents, où les documents eux-mêmes sont enrichis avec des termes supplémentaires, augmentant ainsi leur probabilité de correspondre à des requêtes pertinentes. Les méthodes d'expansion de documents ont démontré leur efficacité

dans des contextes de recherche zero-shot, en améliorant la couverture des termes sans nécessiter de données étiquetées explicites (Izacard *et al.*, 2022).

2.2 Approches Neuronales et Recherche Dense

Les modèles neuronaux de recherche représentent un changement de paradigme en RI, exploitant l'apprentissage profond pour encoder les requêtes et les documents sous forme de représentations vectorielles denses. Contrairement aux modèles lexicaux basés sur la correspondance exacte des termes, les approches de recherche dense capturent la similarité sémantique à l'aide de plongements de haute dimension.

Un exemple notable est Dense Passage Retrieval (DPR) (Karpukhin *et al.*, 2020), qui utilise des encodeurs à base de transformeurs (tels que BERT) pour projeter les requêtes et les documents dans un espace vectoriel commun. La récupération est ensuite effectuée via une recherche des plus proches voisins approximés, permettant ainsi une correspondance sémantique au-delà du chevauchement explicite des termes. Bien que DPR améliore considérablement le recall par rapport à BM25, il introduit des coûts computationnels élevés en entraînement et en inférence.

ColBERT (Khattab & Zaharia, 2020) affine la recherche dense en intégrant des mécanismes d'interaction tardive (late interaction), où les plongements des requêtes et des documents sont calculés à un niveau granulaire de tokens. Cette approche améliore la précision de la récupération, mais nécessite d'importantes ressources de stockage et de mémoire en raison du besoin de conserver les plongements au niveau des tokens (Wang *et al.*, 2024).

Des avancées récentes ont exploré des techniques d'apprentissage auto-supervisé et contrastif pour améliorer l'efficacité des modèles de recherche dense. Les méthodes auto-supervisées comme Contriever (Izacard *et al.*, 2022) génèrent des paires d'entraînement pseudo-positives pour entraîner des récupérateurs denses sans données étiquetées, les rendant ainsi plus applicables dans des contextes réels. De même, les stratégies de pré-entraînement contrastif ont montré des améliorations en performance de récupération, notamment dans des scénarios zero-shot où les annotations de pertinence manuelles sont rares (Izacard *et al.*, 2022).

2.3 Recherche Hybride et Basée sur le Clustering

Les modèles hybrides de récupération cherchent à combiner les atouts des approches de recherche lexicale et neuronale. Les grands modèles de langage (LLMs) ont été appliqués au classement de passages et à la récupération de documents, mais leur passage à l'échelle reste un défi (Zhu *et al.*, 2023).

Une alternative aux modèles hybrides est l'utilisation de techniques de clustering pour structurer les documents et les requêtes en groupes sémantiquement cohérents. Les méthodes de récupération basées sur le clustering partitionnent les collections de documents en clusters en fonction des motifs de coexistence des mots et des mesures de similarité (Jeong *et al.*, 2021). Ces approches offrent une alternative efficace aux modèles de deep learning, car elles ne nécessitent pas de grandes quantités de données étiquetées pour l'entraînement. Les méthodes éparses telles que docT5query (Nogueira *et al.*, 2019) identifient les termes d'expansion de document en utilisant un modèle séquence-séquence qui génère des requêtes possibles pour lesquelles le document donné serait pertinent.

Les modèles hybrides tels que SPLADE (Formal *et al.*, 2021) combinent des caractéristiques lexicales éparses avec des plongements denses pour exploiter à la fois les signaux basés sur les termes et ceux de nature sémantique. Bien que ces approches montrent des résultats prometteurs, elles nécessitent souvent un ajustement coûteux (fine-tuning) et une indexation computationnellement intensive.

Notre méthode de récupération proposée s'appuie sur des techniques de clustering pour regrouper des termes sémantiquement liés, améliorant ainsi la capacité des systèmes de RI traditionnels à gérer la variation lexicale. Contrairement aux modèles neuronaux, notre approche reste interprétable et efficace sur le plan computationnel, tout en surpassant BM25 dans les scénarios impliquant synonymie et substitution de termes.

2.4 Positionnement de Notre Travail

Notre modèle de récupération étend BM25 en intégrant un mécanisme de clustering des mots afin d'affiner l'appariement entre documents et requêtes. Comparé aux modèles de recherche dense, notre méthode conserve son efficacité et son interprétabilité tout en évitant les coûts d'inférence associés aux architectures neuronales. Contrairement aux modèles hybrides tels que SPLADE, qui reposent sur des distributions de termes apprises, notre approche regroupe les mots en fonction de statistiques empiriques de coexistence, garantissant ainsi des performances stables sur divers ensembles de données. Nos résultats expérimentaux indiquent que notre modèle surpasse BM25 sur plusieurs benchmarks du dataset BeIR tout en préservant son efficacité computationnelle.

3 Méthode

Dans cette étude, nous avons d'abord prétraité le texte en supprimant les mots apportant peu d'information (`stop words`) et en effectuant une lemmatisation à l'aide de la bibliothèque spaCy (avec `'en_code_web_sm'`). Le corpus ainsi prétraité a ensuite été utilisé pour entraîner FastText de manière non supervisée avec le modèle `skip-gram`, ce qui a permis d'obtenir des vecteurs de mots de dimension 100.

Pour analyser la coexistence des mots (Turney & Pantel, 2010), nous avons construit une matrice de coexistence. Pour chaque paire de mots w_1 et w_2 dans le corpus, nous avons calculé la probabilité de coexistence selon la formule suivante :

$$\text{coexistence}(w_1, w_2) = \frac{\text{len}(\text{intersection}(E_1, E_2))}{\max(\text{len}(\text{union}(E_1, E_2)), 1)}$$

où E_1 et E_2 représentent les ensembles de documents dans lesquels w_1 et w_2 apparaissent respectivement. Afin d'améliorer le temps de calcul et l'utilisation de la mémoire, nous avons construit cette matrice sous forme creuse (`sparse`).

Ensuite, une matrice de similarité a été créée en sélectionnant les n plus proches voisins de chaque mot w_1 en fonction d'une métrique de similarité. Pour accélérer les calculs sur de grands corpus, nous avons utilisé l'implémentation de Faiss (Douze *et al.*, 2024), notamment lorsque la métrique choisie est la similarité cosinus. Les distances entre voisins ont été calculées, donnant lieu à une matrice de

même dimension que la matrice de coexistence, mais contenant des scores de similarité. Cette matrice est également stockée sous une forme creuse pour limiter la consommation de mémoire.

Les clusters de mots ont ensuite été identifiés en combinant les matrices de similarité et de coexistence. Plus précisément, pour chaque paire de mots w_1 et w_2, nous avons évalué si la condition suivante était satisfaite :

$$\alpha \cdot \text{similarité}(w_1, w_2) + (1 - \alpha) \cdot \text{coexistence}(w_1, w_2) > \tau$$

Si cette condition était remplie, indiquant que la combinaison du score de similarité et de coexistence dépassait un seuil prédéfini τ, les mots w_1 et w_2 étaient considérés comme interchangeables. Pour organiser systématiquement ces relations, nous avons construit un graphe non orienté

$$G = (V, E)$$

où chaque nœud $v \in V$ représentait un mot unique du corpus, et où une arête $(v_i, v_j) \in E$ était tracée entre deux nœuds si leur condition d'interchangeabilité était vérifiée.

Une fois le graphe construit, notre objectif était d'identifier des groupes de mots pouvant former des clusters interchangeables. Ces clusters ont été définis en recherchant toutes les composantes connexes du graphe, c'est-à-dire que chaque cluster correspondait à une composante du graphe. Chaque composante était constituée d'un ensemble de mots liés directement ou indirectement par une séquence de paires interchangeables. Pour nommer les clusters, nous leur avons simplement attribué un numéro.

Après identification des clusters, nous avons procédé à la réécriture des documents en fonction des clusters obtenus. Chaque mot du texte prétraité a été remplacé par le nom de son cluster associé, aboutissant à des documents réécrits exclusivement composés de noms de clusters. Les requêtes ont été réécrites selon la même méthode que les documents : chaque mot de la requête prétraitée a été remplacé par son cluster correspondant, selon les regroupements précédemment déterminés. Dans les cas où un mot de la requête n'avait pas été rencontré lors de l'entraînement, nous avons eu la possibilité soit de l'ignorer, soit de lui attribuer un cluster basé sur son `embedding`. Pour cela, chaque mot inconnu w_i de la requête a été assigné au cluster ayant la plus forte similarité agrégée, en se basant sur un grand nombre de voisins les plus proches extraits du modèle pré-entraîné FastText. Plus précisément, étant donné l'`embedding` de w_i, les voisins les plus proches ont été identifiés, filtrés selon leur score de similarité et leur appartenance à un cluster, puis une agrégation pondérée des scores de similarité a été calculée pour sélectionner le cluster le plus représentatif pour chaque mot.

$$\mathcal{N}(w_i) = \{(s_j, c_j) \mid s_j > \tau, c_j \in C\}$$

où s_j est le score de similarité, τ est le seuil, et $c_j \in C$ est un cluster issu du dictionnaire des clusters. Le cluster final c_i pour chaque mot est choisi comme étant celui avec la plus forte similarité agrégée :

$$S(c_j) = \frac{\sum_{(s_j, c_j) \in \mathcal{N}(w_i)} s_j}{|\mathcal{N}_{c_j}(w_i)|}, \quad c_i = \arg\max_{c_j} S(c_j)$$

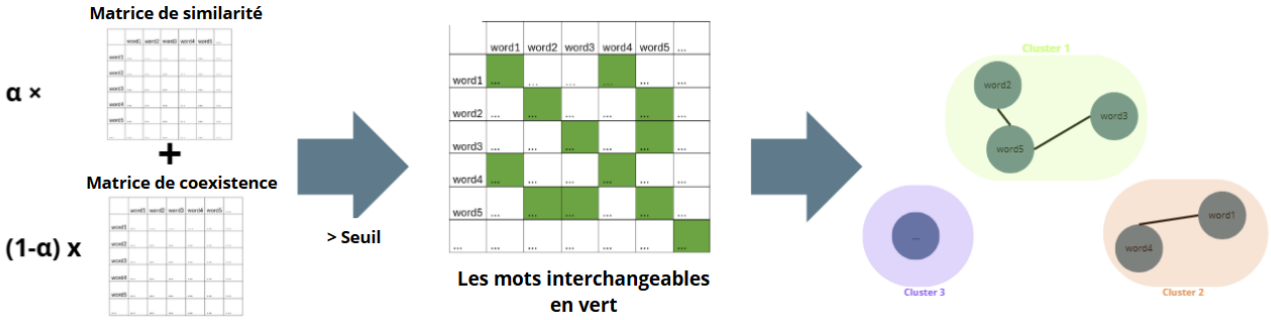

FIGURE 1 – Illustration du processus de clustering des mots

Après cette étape, la recherche de documents a été effectuée en appliquant l'algorithme BM25 (Robertson & Jones, 1976) sur le corpus réécrit et les requêtes réécrites.

En résumé, le prétraitement des corpus est une tâche effectuée une seule fois, après quoi BM25 peut être appliqué pour la recherche d'information. Cette approche garantit que le modèle atteint des performances comparables à celles du BM25 traditionnel, tout en exploitant les représentations de mots enrichies obtenues grâce à FastText et aux clusters de mots construits pour améliorer les résultats.

Pour illustrer visuellement ce processus de clustering, la Figure 1 fournit un aperçu de la manière dont les matrices de similarité et de coexistence sont combinées pour former des clusters de mots. La partie gauche de la figure présente les deux matrices : la matrice de similarité, qui capture les relations entre les mots via les plongements pré-entraînés (Grave *et al.*, 2018), et la matrice de coexistence, qui encode les motifs de coexistence des mots dans les documents (Turney & Pantel, 2010). Ces deux matrices sont fusionnées en utilisant une somme pondérée contrôlée par le paramètre α. Les mots dépassant le seuil dans la matrice résultante sont liés, formant un graphe non orienté où les arêtes indiquent des mots interchangeables. L'étape finale consiste à extraire les composantes connexes de ce graphe pour définir les clusters de mots, comme illustré sur la partie droite de la Figure 1.

4 Expériences

Dans cette section, nous évaluons notre modèle sur plusieurs corpus issus du benchmark BEIR. Toutes nos expériences ont été effectuées sur CPU uniquement, avec les mêmes paramètres pour tous les corpus et sans ajustement spécifique.

4.1 Configuration Expérimentale

Pour chaque test, nous avons défini les paramètres suivants pour notre modèle : $n_{\text{neighbors}} = 0.75$, métrique = similarité cosinus, seuil = 0.75, et $\alpha = 0.76$. Ces paramètres ont été déterminé par recherche en grille sur le coprus NFCorpus. Pour BM25, nous avons utilisé ses paramètres par défaut avec $k_1 = 1.5$ et $b = 0.75$. Afin d'optimiser l'utilisation de la mémoire, les mots ayant une probabilité de coexistence inférieure à un seuil $\theta = 0.05$ ont été mis à zéro.

Pour gérer les mots inconnus, nous avons utilisé un modèle FastText pré-entraîné sur Wikipedia

et Common Crawl avec CBOW et pondération de position. Ce modèle fonctionne dans un espace de 300 dimensions avec des n-grammes de caractères de longueur 5, une fenêtre de taille 5 et 10 échantillons négatifs. Il a été développé par Grave et al. (Grave *et al.*, 2018), qui ont démontré son efficacité dans l'apprentissage de représentations de mots multilingues robustes. Contrairement aux représentations de mots traditionnelles, FastText intègre des informations sous-mots, lui permettant de générer des plongements pour des mots inconnus en exploitant leur structure morphologique. Cette capacité est particulièrement bénéfique pour traiter les mots rares ou les fautes d'orthographe, ce qui en fait un choix adapté pour notre modèle de RI. Afin de garantir la cohérence entre les modèles, nous avons réduit la taille de ses vecteurs pour qu'elle corresponde aux dimensions des autres plongements utilisés dans notre approche.

Pour les mots présents dans les corpus étudiés, nous générons des plongements à l'aide d'un modèle FastText entraîné sur le corpus respectif avec la méthode skip-gram, les paramètres par défaut et une dimension vectorielle de 100. Ces choix de conception ont été guidés par des études d'ablation et notre objectif d'optimisation de l'efficacité mémoire.

4.2 Datasets

Nous évaluons notre modèle en utilisant le benchmark BEIR, introduit par Thakur et al. (Thakur *et al.*, 2021), qui comprend 18 ensembles de données de recherche couvrant neuf tâches distinctes, notamment la vérification des faits et la prédiction de citations. Ces jeux de données couvrent divers domaines tels que Wikipedia et les publications scientifiques. Comme la plupart des ensembles de données BEIR ne disposent pas d'un jeu d'entraînement, ce benchmark se concentre principalement sur la recherche zero-shot.

Conformément aux pratiques standards, nous rapportons deux métriques d'évaluation : nDCG@10 et Recall@100. nDCG@10 évalue la qualité du classement des 10 premiers documents récupérés, ce qui le rend particulièrement pertinent pour les applications orientées utilisateur, comme les moteurs de recherche. En revanche, Recall@100 mesure la proportion de documents pertinents récupérés parmi les 100 premiers résultats, ce qui le rend plus adapté aux tâches d'apprentissage automatique en aval, telles que le question-réponse.

4.3 Résultats

Pour évaluer notre approche, nous comparons les performances des modèles entièrement non supervisés. La Table 1 présente les performances de récupération sur de petits corpus (c'est-à-dire ceux contenant moins de 100 000 documents), notamment NFCorpus (Boteva *et al.*, 2016), SciFact (Wadden *et al.*, 2020), ArguAna (Wachsmuth *et al.*, 2018), SCIDOCS (Cohan *et al.*, 2020), et FiQA-2018 (Maia *et al.*, 2018). Notre modèle montre des résultats compétitifs selon les métriques nDCG@10 et Recall@100, surpassant BM25 de 13% en NDCG@10 et de 2% en Recall@100. De plus, il surpasse les récupérateurs denses et épars proposés précédemment.

Cependant, sur TREC-COVID (Voorhees *et al.*, 2021), Touché-2020 (Bondarenko *et al.*, 2020) et Quora, notre modèle sous-performe par rapport à BM25 en termes de NDCG@10. Néanmoins, il atteint des scores élevés en Recall@100, surpassant BM25 sur deux de ces trois ensembles de données.

Globalement, notre approche améliore BM25 de 2,5% sur la métrique NDCG@10 et de 1,8% sur Recall@100, tout en maintenant un temps d'inférence raisonnable—3,8 fois plus lent lors du remplacement des mots hors vocabulaire et seulement 1,2 fois plus lent sans ce remplacement, avec une perte de performance de seulement 0,12%.

La Figure 2 illustre que docT5query est le seul modèle surpassant BM25 tout en étant plus rapide que UC-FIRe lorsqu'il gère les mots inconnus. Cependant, notre méthode est plus rapide lorsque les mots inconnus sont ignorés, surpassant ainsi docT5query en termes de temps d'inférence. De plus, que UC-FIRe gère ou non les mots hors vocabulaire, notre approche obtient de meilleurs résultats en moyenne que certains autres modèles nécessitant des temps d'inférence bien plus longs. On peut observer que SPLADE obtient les meilleurs résultats en moyenne, mais au prix d'un temps d'inférence plus de vingt fois supérieur à celui de BM25.

Ces résultats mettent en évidence le potentiel du regroupement de mots sémantiquement liés pour combler le fossé lexical. Ils soulignent également que les modèles de RI efficaces ne doivent pas nécessairement être coûteux en calcul et peuvent fonctionner uniquement sur CPU.

TABLE 1 – Performances de récupération en mode in-domain et zero-shot sur les jeux de données BEIR. Les scores correspondent à NDCG@10. La meilleure performance pour un jeu de données est en **gras** et la deuxième meilleure est soulignée.

Dataset	BM25	DeepCT	SPARTA	docT5query	ANCE	TAS-B	GenQ	ColBERT	Splade v2	Contriever	Ours	Ours w/o handling unseen words
SCIDOCS	0.158	0.124	0.126	0.162	0.122	0.149	0.143	0.145	0.158	**0.165**	0.157	0.157
SciFact	0.665	0.630	0.582	0.675	0.507	0.643	0.644	0.671	**0.693**	0.677	**0.693**	0.692
NFCorpus	0.325	0.283	0.301	0.328	0.237	0.319	0.319	0.305	**0.334**	0.328	**0.334**	0.332
ArguAna	0.315	0.309	0.279	0.349	0.415	0.429	0.493	0.233	0.479	0.446	**0.499**	**0.499**
FiQA-2018	0.236	0.191	0.198	0.291	0.295	0.300	0.308	0.317	**0.336**	0.329	0.240	0.240
TREC-COVID	0.656	0.406	0.538	**0.713**	0.654	0.481	0.619	0.677	0.710	0.596	0.614	0.614
Touché-2020	**0.367**	0.156	0.175	0.347	0.240	0.162	0.182	0.202	0.364	0.230	0.319	0.319
Quora	0.789	0.691	0.630	0.802	0.852	0.835	0.830	0.854	0.838	0.865	0.743	0.743
Avg. Performance vs. BM25	+0%	-20.54%	-19.43%	+4.44%	-5.38%	-5.50%	+0.77	-3.05%	+11.42%	+3.56%	+2.51%	+2.39%
Inference time vs. BM25	1	1.25	1	1.5	13.75	6.25	6.25	-	24.5	11.5	3.8	1.2
Best on	1	0	0	1	0	0	0	0	3	2	3	1

TABLE 2 – Performances de récupération en mode in-domain et zero-shot sur les jeux de données BEIR. Les scores correspondent à Recall@100. La meilleure performance pour un jeu de données est en **gras** et la deuxième meilleure est soulignée. Pour le dataset TREC-COVID on a utilisé capped Recall@100.

Dataset	BM25	DeepCT	SPARTA	docT5query	ANCE	TAS-B	GenQ	ColBERT	Splade v2	Contriever	Ours	Ours w/o unseen words
SCIDOCS	0.356	0.314	0.297	0.360	0.269	0.335	0.332	0.344	0.364	**0.378**	0.359	0.359
SciFact	0.908	0.893	0.863	0.914	0.816	0.891	0.893	0.878	0.920	**0.947**	0.925	0.925
NFCorpus	0.250	0.235	0.243	0.253	0.232	0.280	0.280	0.254	0.277	**0.300**	0.262	0.260
ArguAna	0.942	0.932	0.893	0.972	0.937	0.942	**0.978**	0.914	0.972	0.977	0.971	0.972
FiQA-2018	0.539	0.489	0.446	0.598	0.581	0.593	0.618	0.603	0.621	**0.656**	0.526	0.526
TREC-COVID	0.498	0.347	0.409	0.541	0.212	0.387	0.456	0.464	0.123	0.407	**0.580**	**0.580**
Touché-2020	0.538	0.406	0.381	**0.557**	0.458	0.431	0.451	0.439	0.354	0.294	0.552	0.552
Quora	0.973	0.954	0.896	0.982	0.987	0.986	0.988	0.989	0.987	**0.993**	0.955	0.955
Avg. Performance vs. BM25	0%	-8.67%	-11.51%	+3.46%	-10.23%	-3.17%	-0.16%	-2.37%	-7.71%	+8.36%	+1.83%	+1.80%
Best on	0	0	0	1	0	1	-	0	0	5	1	1

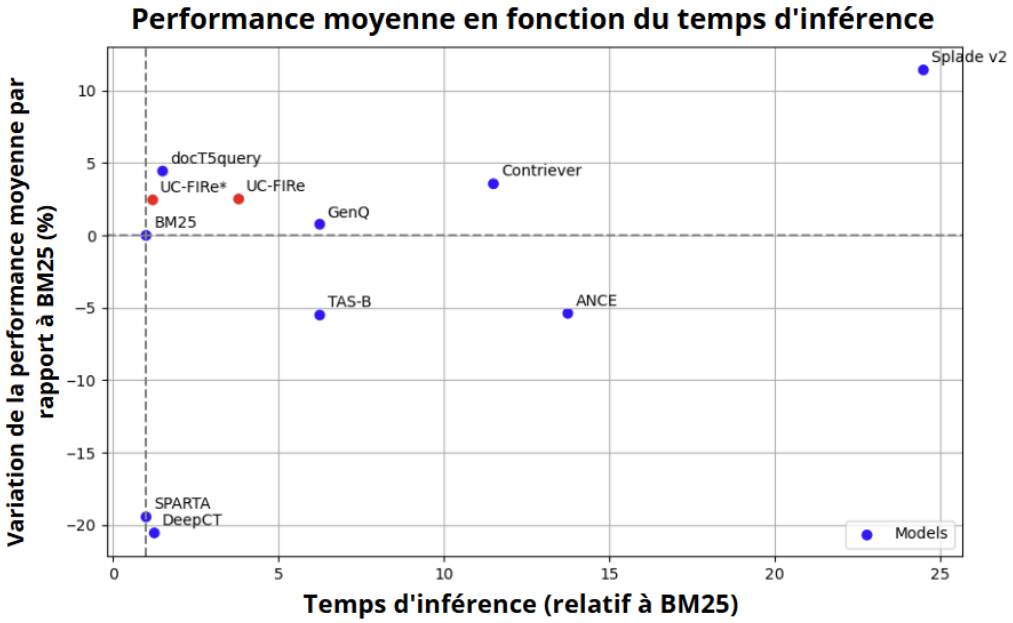

FIGURE 2 – Performance moyenne sur NDCG@10 vs. Temps d'inférence relatif à BM25. UC-FIRe* représente UC-FIRe sans gestion des mots inconnus.

5 Étude d'Ablation

Dans cette section, nous étudions l'influence des différents paramètres sur notre méthode. Dans ces expériences, lorsque nous mentionnons un modèle FastText pré-entraîné, nous faisons référence au modèle cc.en.300 FastText réduit afin d'obtenir des vecteurs de dimension 100 (le même modèle que celui décrit dans (Grave *et al.*, 2018; Bojanowski *et al.*, 2016)). Sinon, les modèles FastText apprennent les représentations de mots uniquement à partir du corpus étudié de manière non supervisée avec le modèle skip-gram, générant ainsi des vecteurs de mots de dimension 100.

Impact du seuil θ sur la matrice de coexistence. Tout d'abord, nous examinons l'influence de ce paramètre sur plusieurs jeux de données. Son rôle principal est de réduire les ressources computationnelles nécessaires pour le prétraitement. Une part significative des valeurs dans la matrice de coexistence est proche de zéro. Par exemple, dans NFCorpus, plus de 93% des valeurs sont inférieures à 0.1, et 50% sont inférieures à 0.01, comme indiqué dans les Tables 3 et 4. Des distributions similaires à celles de la figure 5 sont observées dans d'autres ensembles de données.

Pour NFCorpus et SciFact, nous observons que les performances commencent à se dégrader légèrement lorsque le seuil de la matrice de coexistence dépasse 0.06 (Figures 3, 4). Cependant, en dessous de ce seuil, les résultats restent largement inchangés. Cela indique que mettre plus de la moitié des valeurs à zéro réduit drastiquement l'utilisation mémoire sans affecter négativement les performances.

TABLE 3 – Valeurs des quantiles de coexistence de 0.0 à 1.0 pour NFCorpus

Quantiles	Valeurs
0.0	0.0004
0.1	0.0021
0.2	0.0039
0.3	0.0058
0.4	0.00813
0.5	0.01084
0.6	0.1470
0.7	0.204
0.8	0.0304
0.9	0.0588
1.0	1.0

TABLE 4 – Valeurs des quantiles de coexistence de 0.9 à 1.0 dans NFCorpus

Quantiles	Valeurs
0.9	0.0588
0.91	0.0666
0.92	0.0796
0.93	0.0901
0.94	0.1111
0.95	0.1428
0.96	0.2
0.97	0.25
0.98	0.5
0.99	1.0
1.0	1.0

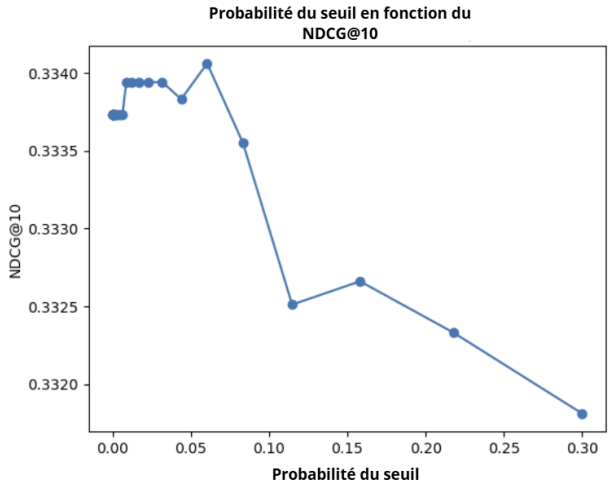

FIGURE 3 – Performance (NDCG@10) sur NFCorpus en fonction du seuil θ sur la matrice de coexistence.

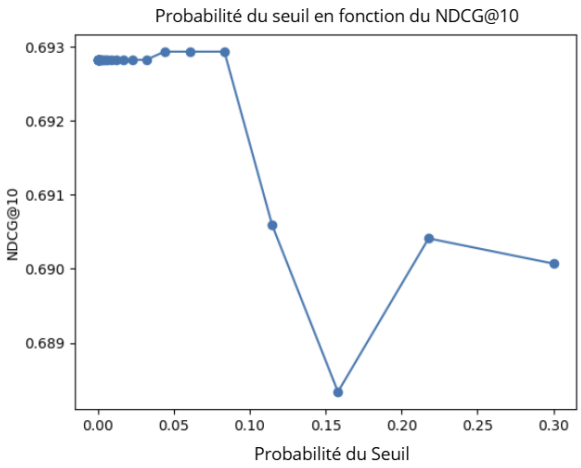

FIGURE 4 – Performance (NDCG@10) sur SciFact en fonction du seuil θ sur la matrice de coexistence.

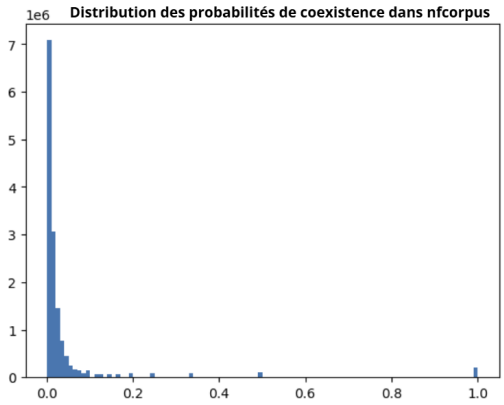

FIGURE 5 – Distribution des probabilités de coexistence dans NFCorpus.

Utilisation de la matrice de coexistence seule vs utilisation de la matrice de similarité seule vs combinaison des deux. Ensuite, nous évaluons l'impact du paramètre α, qui équilibre les contributions des matrices de similarité et de coexistence.

En utilisant uniquement la matrice de coexistence (Figure 6), nous observons qu'avec un seuil supérieur à 0.5, notre modèle dépasse BM25, atteignant des scores NDCG@10 d'environ 0.33 sur NFCorpus lorsqu'il gère les mots inconnus.

En revanche, en s'appuyant uniquement sur la matrice de similarité (Figure 7), les scores restent systématiquement bas et ne dépassent jamais ceux de BM25 combiné avec une gestion des mots manquants. De plus, le nombre de voisins a un impact minimal sur ce résultat.

Lorsque α est fixé à 0.5 (Figure 8), nous obtenons les meilleurs résultats avec un seuil de 0.6 et plus de 83 voisins. En dessous de ce seuil, les scores les plus élevés sont obtenus avec seulement 5 voisins. Cependant, lorsque le nombre de voisins augmente, le score NDCG@10 diminue.

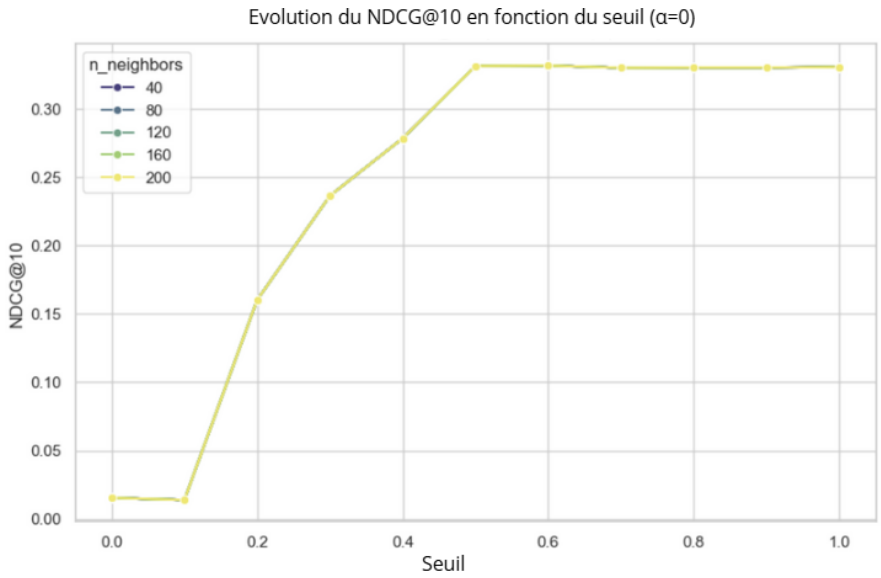

FIGURE 6 – Performance (NDCG@10) sans la matrice de similarité ($\alpha = 0$) en fonction des seuils τ.

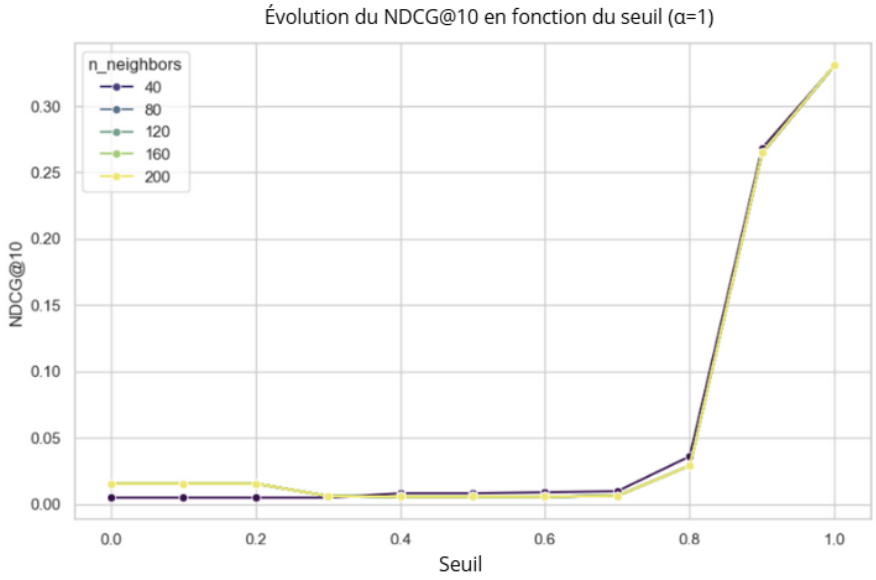

FIGURE 7 – Performance (NDCG@10) sans la matrice de coexistence ($\alpha = 1$) en fonction des seuils τ.

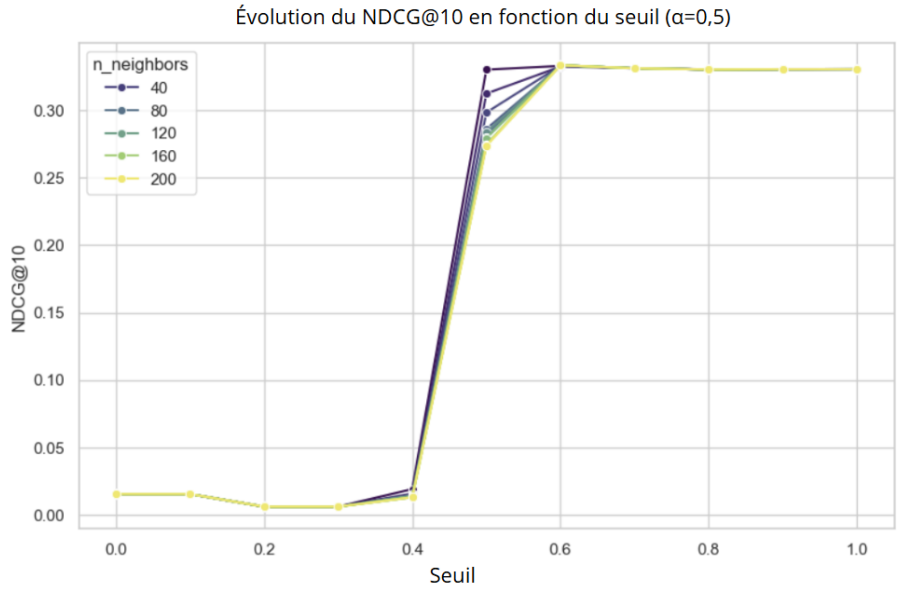

FIGURE 8 – Performance (NDCG@10) avec une pondération égale des matrices ($\alpha = 0.5$) en fonction des seuils τ.

Ignorer les mots non vus vs gérer les mots non vus avec un modèle pré-entraîné vs gérer les mots non vus avec un modèle entraîné uniquement sur le corpus. Bien que la différence soit subtile, les meilleurs résultats sont généralement obtenus en utilisant un modèle FastText entraîné sur le corpus pour générer des embeddings pour les mots du corpus, combiné avec un modèle FastText pré-entraîné pour gérer les mots non vus, comme le montre le Tableau 5. Le modèle pré-entraîné est particulièrement efficace pour cette tâche en raison de son entraînement extensif, ce qui lui permet de fournir des représentations de mots plus précises pour deux raisons principales : il possède un vocabulaire plus large et offre de meilleures représentations pour les mots hors vocabulaire.

TABLE 5 – Scores NDCG@10 pour différents jeux de données utilisant diverses stratégies d'embedding et méthodes de gestion des mots non vus. Le meilleur résultat est en **gras** et le deuxième meilleur est souligné.

Embeddings	Gestion des mots non vus	NFCorpus	Scifact	Arguana	Scidocs	FiQA	Temps d'inférence vs BM25
Pré-entraîné	Pré-entraîné	0.33107	0.67683	0.46843	0.15671	0.22999	3.8
Pré-entraîné	non géré	0.32914	0.67935	0.46843	0.15673	0.22999	1.2
Entraîné sur le corpus	Entraîné sur le corpus	0.33162	0.69029	**0.49902**	0.15651	0.23439	1.4
Entraîné sur le corpus	non géré	0.33163	0.69231	0.49897	0.15674	0.23953	1.2
Entraîné sur le corpus	Pré-entraîné	**0.33389**	**0.69265**	0.49888	**0.15695**	**0.23971**	3.8

6 Conclusion

En conclusion, UC-FIRe démontre des performances prometteuses, surpassant le traditionnel BM25 sur diverses métriques d'évaluation, y compris NDCG@10 et Recall@100. En utilisant le regroupement de mots pour atténuer l'écart lexical, notre méthode permet d'obtenir à la fois une efficacité et une capture effective des relations sémantiques. Par conséquent, elle présente une alternative pratique et économique aux méthodes de RI neuronales ou hybrides existantes, en particulier dans les contextes où les données d'entraînement étiquetées sont rares ou indisponibles.

De plus, le fonctionnement du modèle exclusivement sur des CPU, sans nécessiter d'inférence intensive sur GPU, le distingue des approches neuronales qui nécessitent généralement des ressources computationnelles substantielles. Cette caractéristique améliore son adéquation pour un déploiement dans des environnements à ressources limitées ou des scénarios nécessitant des réponses de récupération rapides. Ainsi, UC-FIRe peut être intégré directement dans un système d'entreprise déjà basé sur BM25, sans modifications majeures de l'infrastructure.

Plusieurs pistes prometteuses pour des recherches futures méritent d'être explorées. Une direction possible consiste à étudier les plongements dynamiques—des représentations de mots contextualisées qui s'ajustent en fonction de contextes spécifiques, comme ce que propose BERT. Bien que les plongements dynamiques puissent potentiellement améliorer la précision sémantique, ils pourraient introduire des défis computationnels significatifs. Par conséquent, explorer des stratégies pour équilibrer les avantages des plongements dynamiques avec l'efficacité computationnelle serait une étape importante.

De plus, des recherches supplémentaires sur des techniques de regroupement avancées, comme le regroupement spectral par exemple, pourraient améliorer les performances de notre modèle. Évaluer le modèle dans des environnements réels, tels que des systèmes de recherche interactifs ou des plateformes de recherche d'entreprise à grande échelle, fournirait également des informations précieuses sur son efficacité pratique et son potentiel d'application plus large.

Références

ALMASRI A. (2017). *Reducing term mismatch probability by exploiting semantic term relations*. Thèse de doctorat, Université Claude Bernard Lyon 1.

BAI J., SONG R., LIN C.-Y. & YU Y. (2011). A kernel-based approach to handling term mismatch. In *Proceedings of the 20th international conference on World wide web*, p. 105–114 : ACM.

BOJANOWSKI P., GRAVE E., JOULIN A. & MIKOLOV T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv :1607.04606*.

BONDARENKO A., FRÖBE M., BELOUCIF M., GIENAPP L., AJJOUR Y., PANCHENKO A., BIEMANN C., STEIN B., WACHSMUTH H., POTTHAST M. & HAGEN M. (2020). Overview of touché 2020 : Argument retrieval. In *Working Notes Papers of the CLEF 2020 Evaluation Labs*, volume 2696 de *CEUR Workshop Proceedings*.

BOTEVA V., GHOLIPOUR D., SOKOLOV A. & RIEZLER S. (2016). A full-text learning to rank dataset for medical information retrieval. In *Proceedings of the 38th European Conference on Information Retrieval (ECIR 2016)*, p. 716–722.

COHAN A., FELDMAN S., BELTAGY I., DOWNEY D. & WELD D. (2020). Specter : Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 2270–2282.

CRESTANI F. (2000). Exploiting the similarity of non-matching terms at retrieval time. *Information Processing & Management*, **36**(4), 495–516.

DOUZE M., GUZHVA A., DENG C., JOHNSON J., SZILVASY G., MAZARÉ P.-E., LOMELI M., HOSSEINI L. & JÉGOU H. (2024). The faiss library. *arXiv preprint arXiv :2401.08281*.

FORMAL T., LASSANCE C., PIWOWARSKI B. & CLINCHANT S. (2021). Splade v2 : Sparse lexical and expansion model for information retrieval. *arXiv preprint arXiv :2109.10086*.

FURNAS G. W., LANDAUER T. K., GOMEZ L. M. & DUMAIS S. T. (1987). The vocabulary problem in human-system communication. *Communications of the ACM*, **30**(11), 964–971.

GAO L. *et al.* (2022). Precise zero-shot dense retrieval without relevance labels. *arXiv preprint arXiv :2209.10063*.

GOLDBERGER J., ROWEIS S., HINTON G. & SALAKHUTDINOV R. (2005). Neighbourhood components analysis. In *Advances in Neural Information Processing Systems*, volume 17, p. 513–520.

GRAVE E., BOJANOWSKI P., GUPTA P., JOULIN A. & MIKOLOV T. (2018). Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

IZACARD G. *et al.* (2022). Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

JEONG S., BAEK J., PARK C. & PARK J. C. (2021). Unsupervised document expansion for information retrieval with stochastic text generation. *arXiv preprint*.

KARPUKHIN V. *et al.* (2020). Dense passage retrieval for open-domain question answering. In *EMNLP*.

KHATTAB O. & ZAHARIA M. (2020). Colbert : Efficient and effective passage search via contextualized late interaction over bert. In *SIGIR*.

LEI Y. *et al.* (2023). Unsupervised dense retrieval with relevance-aware contrastive pre-training. *arXiv preprint arXiv :2306.03166v1*.

MAIA M., HANDSCHUH S., FREITAS A., DAVIS B., MCDERMOTT R., ZARROUK M. & BALAHUR A. (2018). Www'18 open challenge : Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018, WWW '18*, p. 1941–1942.

METZLER D. *et al.* (2023). Query expansion with large language models : Challenges and opportunities. *arXiv preprint arXiv :2301.08801v1*.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*.

NOGUEIRA R., YANG W., LIN J. & CHO K. (2019). Document expansion by query prediction. *arXiv preprint arXiv :1904.08375*.

RAMOS J. (2003). Using tf-idf to determine word relevance in document queries.

ROBERTSON S. & JONES S. (1976). Relevance weighting of search terms. *Journal of the American Society for Information science*, **27**, 129–146. DOI : 10.1002/asi.4630270302.

THAKUR N. *et al.* (2021). Beir : A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *NeurIPS*.

TURNEY P. D. & PANTEL P. (2010). From frequency to meaning : Vector space models of semantics. *arXiv preprint arXiv :1003.1141*.

VOORHEES E., ALAM T., BEDRICK S., DEMNER-FUSHMAN D., HERSH W. R., LO K., ROBERTS K., SOBOROFF I. & WANG L. L. (2021). Trec-covid : Constructing a pandemic information retrieval test collection. *SIGIR Forum*, **54**(1).

WACHSMUTH H., SYED S. & STEIN B. (2018). Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 241–251.

WADDEN D., LIN S., LO K., WANG L. L., VAN ZUYLEN M., COHAN A. & HAJISHIRZI H. (2020). Fact or fiction : Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 7534–7550.

WANG J., HUANG J. X., TU X., WANG J., HUANG A. J., LASKAR M. T. R. & BHUIYAN A. (2024). Utilizing bert for information retrieval : Survey, applications, resources, and challenges. *ACM Computing Surveys*, **56**(7), 1–33.

ZHAO L. & CALLAN J. (2010). Term necessity prediction. *Proceedings of the 19th ACM international conference on Information and knowledge management (CIKM)*, p. 259–268.

ZHU Y. *et al.* (2023). Large language models for information retrieval : A survey. *arXiv preprint arXiv :2308.07107v4*.

ZHUANG Y. & ZUCCON G. (2021). Dealing with typos for bert-based passage retrieval and ranking. *arXiv preprint arXiv :2108.12139*.