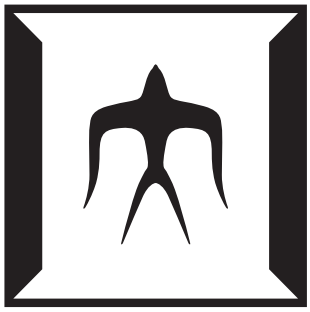
論文 / 著書情報
Article / Book Information

題目(和文)	事前学習済み言語モデルを用いた検索モデルに対する教師なしドメイン適応
Title(English)	
著者(和文)	飯田大貴
Author(English)	Hiroki Iida
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第12780号, 授与年月日:2024年3月26日, 学位の種別:課程博士, 審査員:岡崎 直観,井上 中順,徳永 健伸,宮崎 純,村田 剛志
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第12780号, Conferred date:2024/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

博士論文

事前学習済み言語モデルを用いた検索モデルに対する教師なしドメイン適応

飯田 大貴



東京工業大学 情報理工学院
情報工学系 知能情報コース

本論文は東京工業大学情報理工学院に
博士（工学）授与の要件として提出した博士論文である.

審査委員：
岡崎 直観 教授　（主指導教員）
井上 中順 准教授
徳永 健伸 教授
宮崎 純 教授
村田 剛志 教授

Abstract

Information retrieval (IR) is a widely used technique for retrieving data that matches a query request from a large amount of data. In particular, Web search, a service for retrieving documents on the Internet, has become so pervasive that it is now indispensable in our daily lives. Traditional IR methods rely on token matching between queries and documents. These models fail to capture contextual nuances and semantic similarities when the same concept is presented with different terminologies. Recent advances have seen the integration of pre-trained language models like BERT in retrieval models, enhancing the ability of search systems to interpret the meanings of queries and documents. A well-known example in these models is dense retrieval, which encodes queries and documents into dense vectors, computing relevance through the inner product of these vectors. This approach has shown a marked improvement in capturing semantic relationships over traditional token-based matching, mainly when substantial supervision data (query-document pairs), typically sourced from click logs, is available.

Despite these advancements, challenges persist in environments where ample supervised data is impractical or infeasible, such as in specialized domain searches or searches within private organizational databases. These settings often suffer from a scarcity of query-document pairs due to the requirement of domain expertise for data generation or security restrictions that prevent using log data. In addition, transferring a dense retrieval model to a target domain trained on a source domain for which a large amount of supervision data is available is also difficult because the distribution between the source domain and the target domain changes in terms of vocabulary, word frequencies, types of queries, and relevant

documents to the queries.

Therefore, this study aims to improve the accuracy of zero-shot retrieval where no supervised data is available on the target domain. Specifically, we propose an unsupervised domain adaptation method for retrieval models using pretrained language models.

First, we propose a method that uses the importance of tokens in the target domain. One of the causes of inaccuracy in dense retrieval on the target domain is the inability to rank documents higher with exact keyword matches in the query. On the other hand, IR models that use token matching, like BM25, cannot capture context. Therefore, we propose Contextualized-BM25 (C-BM25), a hybrid model combining the strength of keyword matching with contextual understanding, using context similarity of exact matching tokens between queries and documents. For calculating the context similarity, we use a dense retrieval model. Furthermore, weighting the exactly matched tokens with BM25 allows us to give more weight to keywords in the target domain. We evaluated C-BM25 on a benchmark dataset of zero-shot retrieval. The result showed the effectiveness of C-BM25.

One of the challenges of C-BM25 is that it can only apply to dense retrieval. However, it is challenging to use dense retrieval in highly specialized domains. This is because the accuracy of dense retrieval is greatly reduced since the vocabulary and word frequencies in specialized domains differ from those in the source domain. One retrieval model that solves this problem is SPLADE, which encodes queries and documents into sparse vectors and performs query expansion and document expansion. SPLADE is also a highly accurate retrieval model in domains other than the source domain. Therefore, an unsupervised domain adaptation method applicable to SPLADE is required to improve retrieval accuracy further in specialized domains.

For this reason, we propose using a domain adaptation method of a pretrained language model to improve the accuracy of SPLADE in highly specialized domains. We used AdaLM, a domain adaptation method for pretrained language models, which adds vocabulary to BERT and performs continual pretraining on

a corpus from the target. In addition, we used the importance of tokens in the target data to rank documents higher, including the keywords in the query. Specifically, we weighted each element of the sparse vector encoded by SPLADE with IDF. In addition, we added the relevance scores in BM25 together. Through experiments, we have shown that SPLADE with AdaLM is, on average, more accurate in zero-shot retrieval than existing methods in the bio-medical and scientific domains, where the vocabulary and word frequency differ significantly from the source domain. In addition, we showed that AdaLM is more accurate than pseudo-queries, which is an effective unsupervised domain adaptation method for retrieval. This verified the effectiveness of AdaLM as an unsupervised domain adaptation method. Finally, since AdaLM can be applied to all IR models that use pre-trained language models, we applied it to several IR models, including dense retrieval, and demonstrated its effectiveness. We also showed that applying AdaLM to multiple retrieval models and ensembling them further improved the retrieval accuracy.

Keywords: Information Retrieval, Zero-shot Retrieval, Pretrained Language Model, Unsupervised Domain Adaptation, Dense Retrieval

Doctoral Dissertation, TokyoTech

謝辞

本研究の遂行および博士論文を執筆には，多くの方々にお力添えをいただきました．この場にて厚く御礼申し上げます．

主指導教員の岡崎直観教授には，博士課程を通して多くの指導をいただきました．論文執筆に不慣れな私でしたが，論文の書き方をほとんど一から指導いただきました．また，入学後に研究テーマの変更もあり紆余曲折ありましたが，こうして最終的に博士論文として一貫したテーマとしてまとめられたのは，岡崎先生に研究環境を整えていただいたお陰であると感じています．入学前に抱いていた検索に関する課題は，昨今の進展で大きく解決に向かったと考えており，その期間に研究できたことは，大変幸福でした．

井上中順准教授には，博士論文の審査の過程で大変お世話になりました．最先端の状況との差分について多くご質問をいただきました．本論文執筆を通じてよりそれらを明確化できたと考えております．

徳永健伸教授には，おなじ自然言語処理分野のご専門として，示唆に富むコメントをいただきました．特に，法律分野との違いに関しての議論を通じて，継続事前学習がどのようなドメインシフトに対して有効なのかをより明確にすることができました．また，リカレント教育発展研修の審査でも，機械学習の導入過程についてご質問いただきました．また，そのような職務を行う場合に，より解像度の高く遂行できるようにできるように努めたいと思います．

宮崎純教授には，博士課程の入学試験から博士論文の審査と大変お世話になりました．博士論文の審査の過程で，情報検索の見地から基本となる指摘をいただきました．また，入学試験では実用になるのかという質問をいただいておりました．テーマの変更があったものの，いただいた質問に本論文が実用に貢献できていると幸いです．また，リカレント教育発展研修の審査でも，今後の検証で気

を付けるべきポイントを指摘いただきました．

村田剛志教授にも，博士論文の審査にて大変お世話になりました．予備審査でご指摘いただいた，他手法との位置付けの明確化によって，より本論文の位置付けが明確になったと考えております．

秘書・支援員の小西由希子さん，雲財祐子さん，古谷奈緒子さん，中川恵理子さんにはRAの勤怠申請や出張，英文校正や論文出版など研究遂行をスムーズに進めるための事務処理を支えていただきました．手続き間違いなどあったかと思いますが，いろいろご迷惑をおかけしました．

岡崎研究室OB/OGの方々にも感謝いたします．富士通株式会社の平岡達也さんには，入学前から大変お世話になりました．入学試験や1本目の論文の添削，研究自体のやり方など，いろいろとアドバイスいただきました．1本目の論文は，今見ても大変なものだったと思いますが，丁寧に添削いただきました．ありがとうございました．また，平岡さん博士課程に必要な手続き等整理してくださったため，なんとか論文を書き上げることができました．

岡崎研究室のみなさんには，日々のリサーチセミナーやペーパーリーディングなどお世話になりました．高瀬翔元助教や金子正弘研究員には，継続的にトップカンファレンスに研究を発表している研究者の思考方法など，多くの刺激をいただきました．なかなかそのレベルまでには到達していないですが，今後も明確に研究を構成できるよう精進したいとおもいます．また，金子さんには，投稿前論文にもレビューいただきました．水木栄さんには，同じ大学同じ学科出身ということで，入学前から大変お世話になりました．水木さんの構造化された発表やコメントにはいつも憧れを抱いております．一足早く卒業された丹羽彩奈さんには，度々論文を確認していただき，的確なコメントをいただきました．また，博士課程を過ごす上での必要な情報をいろいろいただきました．Dの会にもお誘いくださりありがとうございました．文 翔煥さんには，サーバーインフラやABCIの使い方の面でお世話になりました．また，1年目の際に割り当てポイントをオーバーしてしまい，申し訳ありませんでした．Youmi Maさんには，研究生活でお世話になりました．Webでの論文掲載や水木栄さんがとっていらしたアナウンスのメモも引き継いでくださりありがとうございます．また，言語処理学会では踏み込んだ議論もでき大変楽しく過ごせました．吉川和さんとは研究の議論をいくつかさせてもらいました．村岡雅康さんには，先輩博士課程パパとして，変わら

ず研究されている姿に励ましをいただきました．劉 傲さんには同時期に博士論文審査を開始しました．先行してくださったので，いろいろと参考にさせていただきました．Erick Mendieta Molina さんや Vijay Daultani さんには，大変フランクに交流させてもらいまいした．拙い英語でしたが，聞いて貰える姿勢に安心を覚えました．古山翔太さん，前田航希さん，服部翔さんには，サーバーインフラの面でお世話になりました．古山翔太さんには本論文の確認もいただき，表現の改善点を多数いただきました．平井翔太さんには，ことある事に幹事を引き受けていただいてありがとうございました．お陰様で楽しく交流ができたと思います．Trang NP Nguyen さんには，因果推論と機械学習の関連研究を紹介いただいきました．本研究とは関連はあまりありませんが，将来取り組んでみたいと思います．王安さん，Mengsay Loem さん，大井聖也さんには大規模言語モデルプロジェクトでお世話になっています．工数が裂けずなかなか進められてなかったですが，引き続き頑張っていければと思います．

株式会社レトリバの皆さんには大変お世話になりました．博士論文が完成まで至ったのは，多くの面で融通を利かせてもらったお陰であると考えています．特に，同チームである西鳥羽二郎さんと勝又智さんには業務面で負担をおかけすることもあったかもしれません．また，制度面で融通を利かせてもらった河原一哉さんにも改めて感謝申し上げます．

D の会の皆様には，発表の場をいただきありがとうございました．このように，発表をまとめる場をもらえたため，研究の方向づけができたかと思います．また，いろいろな発表を聞かせてもらいました．様々なシーズがあることや様々な発表スタイルがあること，刺激になりました．

本論文では，要旨の校正に ChatGPT (https://chat.openai.com/ (2023 年 10 月アクセス)) を用いました．この場を借りて明記したします．

最後に，大学とは全く違う専門でも応援してくれた父，母，生活を支えてくれた妻，娘，義父，義母に感謝します．

目次

第 1 章

序論

1.1　検索と事前学習済み言語モデル

情報検索は，大量のデータから要求に合致したデータを取り出す技術であり，広く用いられている．特にインターネット上の文書検索を行うサービスである Web 検索のシェアトップにある Google[1] は，今や日常生活にも欠かせないほど浸透している．実際，人々は一日平均 3~4 回ほど Google で検索を行うと言われている (Wise 2023)．

Google に代表されるように，ユーザは検索システムに単語や自然文をクエリとして入力し，システムは関連する文書をその関連度順に出力する．そのため，以前より自然言語処理の技術が用いられている．その中でも近年は BERT (Devlin et al. 2019) などの事前学習済み言語モデルを用いた改善が行われている．BERT によって改善された事例を図 1.1に示す．以前であれば，“do estheticians stand a lot as work” と言うクエリに対して，基本的にはクエリと文書のトークンの一致をベースにしたアルゴリズムが使われていたため，stand が stand-alone とマッチし，図 1.1左のような，独立開業に関する検索結果になっていた．対して，BERT を用いる場合は，文脈を捉えることが可能となり，図 1.1右のような，身体的な負荷に関する記事を検索できるようになったとされている．

BERT 等の事前学習済み言語モデルを用いた検索モデルでは，まずクエリと文書を文末トークンで結合して BERT に入力し関連度を推定するクロスエンコーダ (Nogueira and Cho 2019, Yang et al. 2019) という方式が提案された．この方

[1] https://www.google.com/

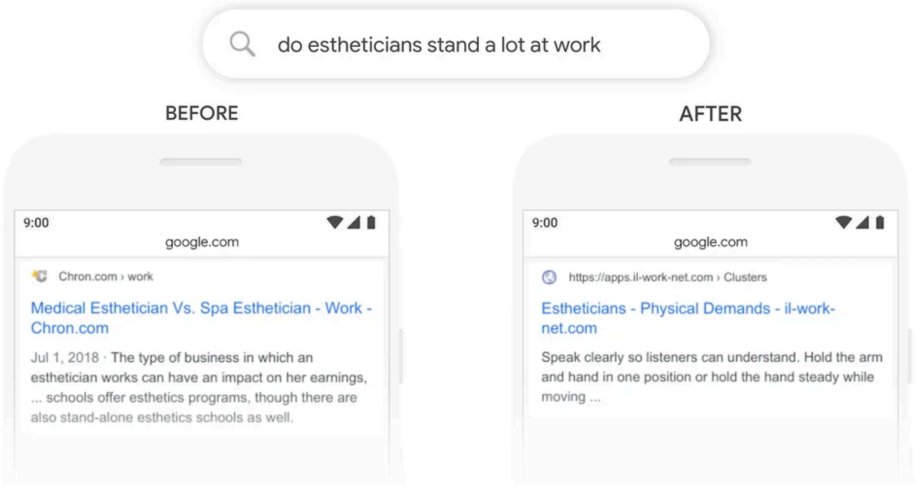


図 1.1: BERT の適用による「エステティシャンは立ち仕事が多いか」というクエリに対するグーグル検索の結果の変化．左が適用前であり，stand と stand-along がマッチしたため，独立開業に関する検索結果が上位に来ている．右が適用後．クエリと文書の文脈を捉えることで，身体的な負荷に関する記事を検索できるようになった．https://blog.google/products/search/search-language-understanding-bert/より引用 (2023 年 9 月にアクセス)．

式は精度が高いもの，検索速度が遅いことが知られれている．従来のトークン一致をベースとしたアルゴリズムでは，高速化のために検索対象文書を事前にインデックスしていたが，クロスエンコーダではこれができないためである．その後，クエリと文書を BERT を用いてそれぞれベクトルにエンコードし，内積やコサイン類似度を用いて関連度を推定する密ベクトル検索モデル (Karpukhin et al. 2020, Xiong et al. 2021, Hofstätter et al. 2021) が提案された．ベクトルにエンコードすることで，トークンの一致を用いたランキングアルゴリズムの課題である，クエリと関連文書におけるトークンの不一致に対しても，同義語・関連度 (Ram et al. 2023) や文脈をとらえて関連文書を検索結果の上位に位置づけることが可能となる．また，密ベクトル検索は近似最近傍法 (Matsui et al. 2018) を用いて事前にベクトルをインデックスすることによって，大量の文書を高速に検索可能である．この性質によって，質問応答やアドホック検索などにおいて，BM25 (Robertson and Walker 1994) のような，トークンの一致をベースとした検索アルゴリズムから，大幅な精度向上を達成しながら，実用的に利用可能な応答速度を実現している．

1.2　事前学習済み言語モデルを用いた検索モデルの課題

ここまで述べたような事前学習済み言語モデルを用いた検索モデルは，大量のクエリと関連文書のペアを教師データとして必要とする．そのため，クリックログ等で教師データが大量に入手可能な Web 検索において用いられている．しかしながら，検索システムは Web 検索でのみ用いられるわけではない．例えば，企業などの組織の内部文書に対する検索，法文書，論文検索など特定のドメインや特定の文書データに対する検索システムが存在する．このようなシステムにおいては，教師データ作成にドメインエキスパートが必要な場合やログデータなどがセキュリティー等の都合上使えない場合があり，必ずしもクエリと文書のペアを十分に集めることができない．そのため，対象データにおいて教師データを用いないことが好まれる．検索問題の設定として対象データにおいて教師データを用いない場合をゼロショット検索と呼ぶ (Thakur et al. 2021)．一方で，BM25 (Robertson and Walker 1994) のような，トークンの一致をベースとした既存の検索アルゴリズムは，ゼロショット検索で一定の精度に達する．そのため，事前学習済み言語モデルを用いた検索モデルも，ゼロショット検索において既存手法以上の検索精度に到達することが求められる．

これを実現する最も単純な方法の一つに，大量の教師データが入手可能なデータで事前学習済み言語モデルを検索モデルとして学習させ，その検索モデルを対象のデータで用いるという方法が考えられる．しかしながら，学習を行なったデータと対象データの間で，語彙や単語頻度，入力されるクエリの種類，関連ありとする文書などが異なる場合がある．このような現象は広くドメインシフト (Kouw and Loog 2018) と呼ばれる．ドメインは，データの出所を指す言葉であり，数学的には確率分布である．また，学習を行ったデータをソースドメイン，対象データを対象ドメインと呼ぶ．このドメインシフトによって，精度が大きく低下することが知られている．Thakur et al. (2021) は，ドメインシフトによる事前学習済み言語モデルを用いた検索モデルの精度低下を計測するために，9 ドメイン 18 データセットから成るベンチマークデータセットである BEIR を提案した．彼らのベンチマーク結果によると，密ベクトル検索を教師データと異なる複数のデータセットに適用した場合，平均的には BM25 を下回る精度になることが明らかになった．そのため，教師データを得られないドメインにおいては，依

然としてBERTを用いた検索モデルを用いることが困難であるため，トークン一致による検索モデルが使用される．故に，このようなドメインでは，事前学習済み言語モデルを用いた検索モデルの特性である文脈を捉えた検索を実現できていない．

さらに，教師データなしで検索の精度向上を行う要求は，特に専門性の高いドメインで強い．専門性の高いドメインでは，専門家が教師データを作成する必要があるなど教師データの作成コストが高い場合が多いためである．このようなドメインでは特に密ベクトル検索の精度低下が大きいことが指摘されている (Thakur et al. 2021)．Thakur et al. (2021) は，その理由を語彙や単語頻度がソースドメインから大きく変化するためとしている．よって，専門的なドメインのように，語彙や単語頻度がソースドメインと異なる場合には，密ベクトル検索によって達成されていた，文脈の考慮や同義語，関連語の考慮はより困難である．ドメインが変化する場合も教師なしで高い検索精度を得る方法の一つは，ドメインの変化に頑健な検索モデルを使用することである．事前学習言語モデルを用いてテキストを高次元の疎ベクトルに変換するSPLADE (Formal et al. 2021) は，BEIRにおいてBM25を平均的に上回る精度となることが示されている．SPLADEによって得られる疎ベクトルの各要素は，事前学習済み言語モデルの語と対応する．そのため，SPLADEはクエリ拡張と文書拡張を行うモデルと考えられる．よって，専門性の高いドメインでも同義語や関連語の考慮が期待できる．さらに，SPLADEは転置インデックスを用いると，高速な検索が可能となる．しかしながら，SPLADEについても密ベクトル検索同様にドメイン変化による精度の低下が発生していると考えられ，ドメイン変化に適切に対応することで，更なる精度向上が見込まれる．よって，密ベクトル検索以外にも適用可能な教師なしドメイン適応手法が求められる．

1.3 解決方法：教師なしドメイン適応

よって，本研究では，ゼロショット検索の精度向上に取り組む．具体的には，事前学習済み言語モデルを用いた検索モデルに対して，教師なしドメイン適応を行う手法を提案する．. これによって，対象データにおいて教師なしで事前学習済み言語モデルを用いた検索モデルを持ちることが可能であるため，トークン一致

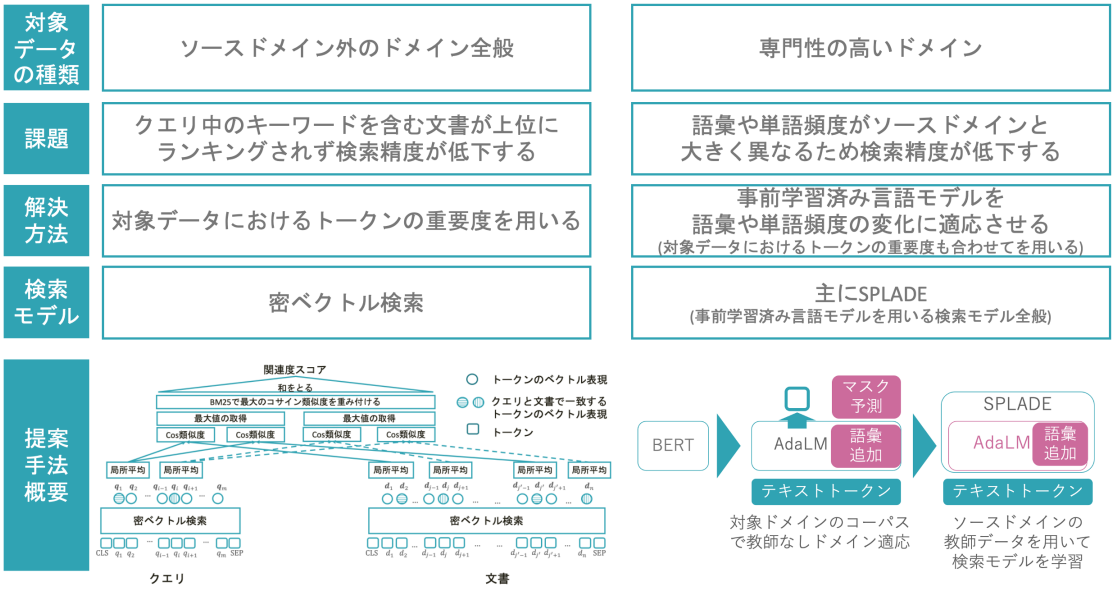


図 1.2: 本研究の概要.

に係る検索モデルの課題を解決することを併せて見る．概要を図 1.2に示す．

一つ目の方法は，対象データにおけるトークンの重要度を用いる方法である．密ベクトル検索において対象データで精度が低下する原因の一つに，クエリ中にあるキーワードが完全一致する文書を上位にできないという点が指摘されている (Formal et al. 2022a, Ram et al. 2023)．この現象は，教師データ中の低頻度語で発生しやすいことが指摘されている (Formal et al. 2022a)．よって，密ベクトル検索がもつ文脈を考慮したスコアリングを行う性質を保持したまま，クエリと文書でトークンが一致することを関連度スコアリングに反映できれば，対象データにおいて精度向上が期待できる．加えて，検索では文書の一部がクエリと適合するため，関連文書とみなされる場合がある (Guo et al. 2016)．そのため，文脈についても文書の一部がクエリと適合していることを関連度として反映させることが更なる精度向上につながると考えられる．

これらを実現する方法として，本研究では Contextualized-BM25 (C-BM25) を提案する．C-BM25 は，クエリと文書で一致したトークン毎に文脈の類似度計算し，同じ語を持つトークン中で類似度の最大値の和を関連度スコアとする．文脈類似度には，トークンの文脈を表現するベクトルのコサイン類似度を用いる．

トークンを表現するベクトルは，クエリ及び文書を密ベクトル検索モデルでそれぞれエンコードすることによって得る．これによって，トークンのみならず文脈についても，文書の一部がクエリと適合することを関連度に反映させる．また，和を取る際に一致したトークンを BM25 で重みづける．BM25 はスコア中でトークンの逆文書頻度 (IDF) をスコアリングとして考慮しており，キーワードは特定の文書のみに出現しやすいため，キーワードが一致する文書の関連度スコアを高くする．そのため，ドメインに合わせたキーワードの重要度を設定できるため，対象データにおいてクエリ中にあるキーワードが完全一致する文書を上位にできないという課題に対して有効な方法と考えられる．

ゼロショット検索における C-BM25 の精度を検証するため，BEIR を用いて評価を行った．その結果，C-BM25 は密ベクトル検索や BM25 を上回る精度を示した．また，対象データにおける精度向上方法として知られる，BM25 の関連度スコアと密ベクトル検索の関連度スコアを足し合わせるという方法 (Ma et al. 2021) や SPLADE を上回る精度となった．特に，密ベクトル検索で精度の低下が大きい，検索対象文書で語彙や単語頻度が大きく異なるドメインに対しては，C-BM25 の方が高い精度を示した．

C-BM25 は，文脈を考慮することによってソースドメイン外において BM25 からの精度向上を達成した．しかし，密ベクトル検索のみに適用可能な手法であった．専門性の高いドメインでは密ベクトル検索の精度低下が大きいため，SPLADE などのドメインの変化に対して頑健なモデルに対してドメイン適応を行うことで，精度向上が見込まれる．また，C-BM25 はクエリと文書でトークンが一致する必要があるが，SPLADE であればクエリと文書でトークンが一致しない場合にも有効である．

そのため，二つ目の方法として，SPLADE にも適用可能かつ専門性の高いドメインで精度を向上させる方法である，事前学習済み言語モデルのドメイン適応を用いることを提案する．本研究では事前学習済み言語モデルに対して，語彙追加と継続事前学習を行う AdaLM (Yao et al. 2021) を行い，その言語モデルに対して SPLADE の学習を行う．さらに，クエリ中のキーワードと完全一致する文書を上位にするため，対象データにおけるトークンの重要度を併せて用いる．具体的には，語彙サイズ次元のベクトルの各要素に対して IDF を掛け合わせる方法，Ma et al. (2021) と同様に BM25 と SPLADE の関連度スコアの和を足し合わせる

方法を共に用いる．本手法の有効性を，教師データである MS MARCO と BEIR のデータセット中で最も語彙と単語頻度が異なる医療ドメインと科学ドメインの 5 データセットを対象データとして検証した．その結果，BM25 と密ベクトル検索のスコアを掛け合わせる，既存の最高精度モデルであった LaPraDor (Xu et al. 2022) を平均的には上回る精度となった．また，ドメイン適応手法の効果を検証するため，疑似クエリを用いたドメイン適応手法である GPL (Wang et al. 2022) と AdaLM の比較を SPLADE や密ベクトル検索など複数の検索モデルに対して行った．その結果，語彙と単語頻度が異なるドメインにおいて，検証したすべての検索モデルにおいて AdaLM が GPL を上回ることを示した．さらに，AdaLM と複数の検索モデルのアンサンブルを行うことで，より精度が向上し，アンサンブルに対しても AdaLM が有効な手法であることを明らかにした．

最後に，提案した二つの手法を汎化誤差の観点から整理した．C-BM25 に関しては，BM25 による重み付けが密度比の変化に対する補正と考えられること定式化した．AdaLM に関しては，汎化誤差の観点からは検索モデルの学習においてエンコーダの初期値以外変化していないことを指摘した．また，AdaLM が入力に対するドメイン適応であることからクエリまたは文書のドメインシフトに適応する手法であることを述べた．そして，実際には共に検索対象文書のドメインシフトに対して有効な方法であることを実験的に示した．

1.4 貢献

本研究の貢献は以下のとおりである．

- 事前学習済み言語モデルを用いた検索モデルに対する教師なしドメイン適応手法として，対象データにおけるトークンの重要度を用いる方法と事前学習済み言語モデルに対する語彙追加を用いる手法の二つ提案した．本論文で提案した手法を汎化誤差の観点から整理を行い，検索対象文書のドメインシフトとの関連を述べた．また，提案手法が実際に検索対象文書のドメインシフトに対して，有効な方法であることを実験的に示した．

- 対象データにおけるトークンの重要度を用いる密ベクトル検索に対するドメイン適応手法として，対象データにおいて教師なしで，キーワードを重視

し文脈情報を利用できるスコアリング方法としてC-BM25を提案した．また，C-BM25の要素分析を通じて，トークンの重要度はドメイン毎に異なるため，BM25による重み付けによるドメイン適応が有効であることを明らかにした．一方，クエリと文書で一致したトークン周辺という局所的な文脈類似度はトークンの重要度よりドメイン変化の影響が小さいため，検索においてソースドメイン外でも有効であることを明らかにした．

- 語彙や単語頻度が異なるドメインにおいて，事前学習済み言語モデルに語彙を追加し継続事前学習するAdaLMを用いることが，疑似クエリを用いる方法よりも効果的であることを示した．また，事前学習済み言語モデルに対するドメイン適応と対象データにおける重みを用いるドメイン適応手法がSPLADEにおいて，共に用いることで精度がさらに向上することを示した．

- AdaLMが複数の検索モデルで有効であり，複数の検索モデルにおいてAdaLMを行った事前学習済み検索モデルを用いた後，それらをアンサンブルすることで，さらに精度が向上することを示した．

1.5 本論文の構成

第2章では，本論文で用いる基本的な概念の導入や関連研究について述べる．具体的には，検索モデルについて述べる．検索モデルを確率モデルで定式化し，古典的な検索モデルであるBM25と事前学習済み言語モデルを用いた検索モデルについて述べる．そして，その学習手法や評価指標について述べる．さらに，事前学習済み言語モデルを用いた検索モデルに対する教師なしドメイン適応について述べる．2.5.4項にてGPLが検索対象文書のドメインシフトへの適法手法であることを述べる．併せて，その他のゼロショット検索における精度向上手法についても述べる．

第3章では，対象データにおけるトークンの重要度を用いる手法として，提案手法であるC-BM25についてその方法と実験について述べる．併せて，C-BM25が共変量シフトに適応する手法と考えられることを述べる．また，クエリと文書で一致したトークン周辺という局所的な文脈の類似度が検索における関連度のス

コアとして広く転移可能であることについて議論する．

第 4 章では，事前学習済み言語モデルに対して語彙追加を行い継続事前学習する教師なしドメイン適応手法である AdaLM の有効性について述べる．また，対象データにおけるトークンの重要度を用いる手法と併用し，その相補性の有無について述べる．さらに，複数の検索モデルに対して AdaLM が有効であることを示し，これらのアンサンブルによって更に精度が向上することを示す．

第 5 章では，本研究のまとめを述べる．また，今後の展望について，近年注目を集めている，大規模言語モデルを用いる方法も含めて述べる．

第 2 章

準備と関連研究

本章では，本論文で用いる基本的な概念や関連研究について述べる．まず，古典的な検索モデルと BERT を用いた検索モデルについて説明する．次に，BERT を用いた検索モデルの学習方法について述べる．その後，本論文のテーマである検索における教師なしドメイン適応について関連研究を述べる．最後に，クエリ中のキーワードを含む文書が上位にランキングされているかどうかを分析する手法と評価指標について述べる．

古典的な検索モデルは，確率モデルで記述できる (Büttcher 他 2020)．また，BERT を用いた検索モデルも同様に確率モデルとして記載できる (Lin et al. 2020)．これらを概観することで，古典的な検索モデルは各トークンを独立に扱っていること，BERT を用いた検索モデルはトークンが独立である仮定なしに利用できることを明らかにする．これらの内容は主に，Büttcher 他 (2020) とLin et al. (2020) を参考にしている．

まず，記号を定義する．語彙を $\mathcal{V}$ し，クエリ集合を Q とし，クエリ $q \in Q$ とする．同様に文書集合を D とし，文書 $d \in D$ とする．また，関連の有無を表す確率変数を $R = \{1, 0\}$ とする．

また，クエリ q, 文書 d が関連する度合いを式 (2.1) のように確率表現によって記述する．

$$p(R = 1|Q = q, D = d) \tag{2.1}$$

簡単のため，以降は $P(R = 1|Q = q, D = d)$ を $P(R = 1|Q, D)$ と表記する．ま

た，r を $R = 1$ の場合とし，$\bar{r}$ を $R = 0$ の場合とする．つまり，

$$p(r|Q, D) = 1 - p(\bar{r}|Q, D) \tag{2.2}$$

と表記する

2.1 古典的な検索モデル

古典的な検索モデルは基本的にクエリと文書で一致するトークンをベースにスコアリングを行う．その定式化においては式 (2.1) から導出される．以下では確率的検索モデルとして代表的な BM25 の導出を行う．その過程で，理想的な単語の重みとして用いるロバートソン/スパルク・ジョーンズ重み付け関数の導出する．ロバートソン/スパルク・ジョーンズ重み付け関数はクエリ中のキーワードを含む文書が上位にランキングされているかどうかを分析に用いることができるが，分析手法自体については 2.6 節で述べる．なお，この分析手法は 3.4.3 項および 4.5 節にて用いている．

2.1.1 確率的検索モデル

ベイズの定理より，$p(r|Q, D)$ と $p(\bar{r}|Q, D)$ はそれぞれ式 (2.3)，式 (2.4) のように表せる．

$$p(r|Q, D) = \frac{p(Q, D|r)p(r)}{p(Q, D)} \tag{2.3}$$

$$p(\bar{r}|Q, D) = \frac{p(Q, D|\bar{r})p(\bar{r})}{p(Q, D)} \tag{2.4}$$

ここで，$p(r|Q, D)$ の対数オッズを関連度のスコアとして用いることを考える．対数オッズは $\log(\frac{p}{1-p})$ で計算されるため，

$$\log\left(\frac{p(r|Q, D)}{1 - p(r|Q, D)}\right) = \log\left(\frac{p(r|Q, D)}{p(\bar{r}|Q, D)}\right) \tag{2.5}$$

$$= \log\left(\frac{p(Q,D|r)p(r)}{p(Q,D|\bar{r})p(\bar{r})}\right) \tag{2.6}$$

となる．さらに，$p(Q,D|R) = p(D|Q,R)p(Q|R)$ とベイズの定理を用いると，

$$\log\left(\frac{p(Q,D|r)p(r)}{p(Q,D|\bar{r})p(\bar{r})}\right) = \log\left(\frac{p(D|Q,r)p(Q|r)p(r)}{p(D|Q,\bar{r})p(Q|\bar{r})p(\bar{r})}\right) \tag{2.7}$$

$$= \log\left(\frac{p(D|Q,r)p(r|Q)}{p(D|Q,\bar{r})p(\bar{r}|Q)}\right) \tag{2.8}$$

$$= \log\left(\frac{p(D|Q,r)}{p(D|Q,\bar{r})}\right) + \log\left(\frac{p(r|Q)}{p(\bar{r}|Q)}\right) \tag{2.9}$$

となる．ここで，$\log(p(r|Q)/p(\bar{r}|Q))$ は，D と独立であるので無視すると，関連度スコアを S とすると，式 (2.10) のように表せる．

$$S = \log\left(\frac{p(D|Q,r)}{p(D|Q,\bar{r})}\right) \tag{2.10}$$

この式は，ある文書がクエリと関連ありと想定されている場合に生成される確率と関連なしと想定されている場合に生成される確率の対数オッズとなっている．

2.1.2 ロバートソン/スパルク・ジョーンズ重み付け関数

さて，文書を語彙 $\mathcal{V}$ 中のトークンが出現するかしないかで表現することを考える．この時，$d = (b_1, b_2, ..., b_{|\mathcal{V}|}) \in \{0,1\}^{|\mathcal{V}|}$ となる．なお，確率変数として表す場合，$D = (B_1, B_2, ..., B_{|\mathcal{V}|}) \in \{0,1\}^{|\mathcal{V}|}$ とする．この時，B_t は 1 か 0 の値を取る確率変数である．また，出現するトークンが独立であるとすると，$p(D|Q,r) = \prod_{t=1}^{|\mathcal{V}|} p(B_t|Q,r)$，と書ける．さらに，クエリも同様な表現をし，クエリ中に現れるトークンのみが関連性に依存するとすると，クエリ中に現れない単語について，$p(B_i|Q,r) = p(B_i|Q,\bar{r})$ となる．この時，クエリに出現するトークンのみを考えれば良いことになる．よって，式 (2.10) は式 (2.11) のように表すことで Q に関する条件づけをなくすことができる．

$$S = \sum_{t \in q} \log\left(\frac{p(B_t|r)}{p(B_t|\bar{r})}\right) \tag{2.11}$$

なお，t はクエリ中のトークンを表すインデックスである．さらに，クエリ中の単語が文書に現れない場合，関連度に影響を与えないと仮定しているため，式 (2.12) のように書ける．

$$S = \sum_{t \in q} \log \left(\frac{p(B_t = b_t|r)}{p(B_t = b_t|\bar{r})} \right) - \sum_{t \in q} \log \left(\frac{p(B_t = 0|r)}{p(B_t = 0|\bar{r})} \right) \tag{2.12}$$

クエリ中に出現するトークンと出現しないトークンで項を分けることで，式 (2.13) のように書ける．

$$S = \sum_{t \in (q \cap d)} \log \left(\frac{p(B_t = 1|r)p(B_t = 0|\bar{r})}{p(B_t = 1|\bar{r})p(B_t = 0|r)} \right) - \sum_{t \in (q \setminus d)} \log \left(\frac{p(B_t = 0|r)p(B_t = 0|\bar{r})}{p(B_t = 0|\bar{r})p(B_t = 0|r)} \right) \tag{2.13}$$

第 2 項は 0 となるため，関連度スコアは式 (2.14) のように書ける．

$$S = \sum_{t \in (q \cap d)} \log \left(\frac{p(B_t = 1|r)p(B_t = 1|\bar{r})}{p(B_t = 0|\bar{r})p(B_t = 0|r)} \right) \tag{2.14}$$

ここで，$p_t = p(B_t = 1|r)$, $\bar{p}_t = p(B_t = 1|\bar{r})$ と表記することで，各単語 t に対しては式 (2.15) のように書ける．

$$w_t = \log \left(\frac{p_t(1 - \bar{p}_t)}{\bar{p}_t(1 - p_t)} \right) \tag{2.15}$$

この，w_t がロバートソン/スパルク・ジョーンズ重み付け関数である．さらに，p_t, $\bar{p}_t$ の推定を $p_t = \frac{n_{t,r}}{n_r}$，$\bar{p}_t = \frac{n_t - n_{t,r}}{|D| - n_r}$ とする．n_t はトークン t を含む文書数，n_r はクエリ q に対する関連ありの文書数，$n_{t,r}$ はトークン t を含むクエリ q に対する関連ありの文書数である．このようにすると，式 (2.15) は式 (2.16) のように書ける．

$$\begin{aligned} w_t &= \log \left(\frac{p_t}{1 - p_t} \right) + \log \left(\frac{1 - \bar{p}_t}{\bar{p}_t} \right) \\ &= \log \left(\frac{n_{t,r}}{n_r - n_{t,r}} \right) + \log \left(\frac{|D| - n_r - n_t + n_{t,r}}{n_t - n_{t,r}} \right) \end{aligned} \tag{2.16}$$

$n_{t,r} \ll n_t$，$n_r \ll |D|$ とすると，式 (2.17) のように近似できる．

$$w_t = \log\left(\frac{p_t}{1-p_t}\right) + \log\left(\frac{|D|-n_t}{n_t}\right) \tag{2.17}$$

さらに，$p_t = \frac{1}{1+\frac{|D|-n_t}{|D|}}$ とおけば，

$$w_t = \log\left(\frac{|D|}{n_t}\right) = \mathrm{IDF}_D(t) \tag{2.18}$$

となる．これは，逆文書頻度（IDF: Inverse Document Frequency）の定義と同じである．ロバートソン/スパルク・ジョーンズ重み付け関数は関連度情報から推定される重みという点で，トークン一致のみで考えた関連度スコアによるランキングに対する理想的な重みと考えられる．また，IDF は関連度情報がない場合の良い近似であると考えられる．

2.1.3 BM25

2.1.2 項では，文書を単語が出現するかどうかで表現していた．しかし，頻度の方が文書の表現としてより多くの情報を含んでいると直感的には考えられる．そのため，文書を単語の頻度で表現することを考える．この時，$d = (f_1, f_2, ..., f_{|\mathcal{V}|}) \in \mathbb{Z}^{\mathcal{V}}$ と書ける．f_i は文書 d 中の単語頻度である．なお，確率変数として表す場合，$D = (F_1, F_2, ..., F_{|\mathcal{V}|}) \in \mathbb{Z}^{\mathcal{V}}$ となる．

このように単語の出現を頻度に変更場合でも，式 (2.14) と同じ形式で書ける．そのため，関連度スコアは式 (2.19) のようになる．

$$S = \sum_{t \in (q \cap d)} \log\left(\frac{p(F_t = f_t|r)p(F_t = f_t|\bar{r})}{p(F_t = 0|\bar{r})p(F_t = 0|r)}\right) \tag{2.19}$$

次に，$p(F_t = f_t|r)$ と $p(F_t = f_t|\hat{r})$ を推定する．クエリと文書と関連の有無と単語の頻度をつなぐ概念として，eliteness (Robertson and Walker 1994) というものを考える．eliteness は，文書中の単語が文書のトピックを表す単語かどうかを示す仮想的な概念である．ある文書の著者はその文書のトピックを伝えるために，

トピックに関連した単語を用いることが一般的と思われる．よって，そのトピックに関係した単語が，他のトピックに関する文書より多くなると期待できる．単語が elite であるかどうかの確率変数を E_t とする．elite であるとき，$E_t = 1$，そうでないとき $E_t = 0$ とする．簡単のため，$E_t = 1$ を e，$E_t = 0$ を $\bar{e}$ とする．関連度と単語頻度の関係は，

$$p(F_t = f_t|r) = p(F_t = f_t|e)p(e|r) + p(F_t = f_t|\bar{e})p(\bar{e}|r) \tag{2.20}$$

$$p(F_t = f_t|\bar{r}) = p(F_t = f_t|e)p(e|\bar{r}) + p(F_t = f_t|\bar{e})p(\bar{e}|\bar{r}) \tag{2.21}$$

と書ける．これを式 (2.19) に代入すると，

$$S = \sum_{t \in q} \log \left(\frac{(p(F_t = f_t|e)p(e|r) + p(F_t = f_t|\bar{e})p(\bar{e}|r))(p(F_t = 0|e)p(e|\bar{r}) + p(F_t = 0|\bar{e})p(\bar{e}|\bar{r}))}{(p(F_t = f_t|e)p(e|\bar{r}) + p(F_t = f_t|\bar{e}))p(\bar{e}|\bar{r})(p(F_t = 0|e)p(e|r) + p(F_t = 0|\bar{e})p(\bar{e}|r))} \right) \tag{2.22}$$

となる．今，$p(F_t = f_t|e)$ と $p(F_t = f_t|\bar{e})$ がポアソン分布 $po(x, \mu)$ に従うと仮定する．つまり，

$$p(F_t = f_t|e) = po(x, \mu_e) = \frac{\exp(-\mu_e)\mu_e^x}{x!} \tag{2.23}$$

$$p(F_t = f_t|\bar{e}) = po(x, \mu_{\bar{e}}) = \frac{\exp(-\mu_{\bar{e}})\mu_{\bar{e}}^x}{x!} \tag{2.24}$$

であるとする．さらに，$p(e|r)$ を q_e，$p(\bar{e}|r)$ を $\bar{q_e}$ として，式 (2.22) に代入すると

$$S = \sum_{t \in q} \log \left(\frac{(po(f_t, \mu_e)q_e + po(f_t, \mu_{\bar{e}})(1 - q_e))(po(0, \mu_e)\bar{q_e} + po(0, \mu_{\bar{e}})(1 - \bar{q_e}))}{(po(f_t, \mu_e)\bar{q_e} + po(f_t, \mu_{\bar{e}})(1 - \bar{q_e}))(po(0, \mu_e)q_e + po(0, \mu_{\bar{e}})(1 - q_e))} \right) \tag{2.25}$$

となる．

この式の性質を見てみると，まず $f_t = 0$ で $s = 0$ となり，クエリと文書の双方で出現するトークンのみが関連度に影響を与えるという仮定を満たしていると言える．また，f_t が増加すると s は増加するという性質も満たしている．

最後に，$f_t \to \infty$ の場合について考える．式 (2.25) を変形し，

$$S = \sum_{t \in q} \log \left(\frac{(q_e + \frac{po(f_t,\mu_{\bar{e}})}{po(f_t,\mu_e)}(1-q_e))(\frac{po(0,\mu_e)}{po(0,\mu_{\bar{e}})}\bar{q}_e + (1-\bar{q}_e))}{(\bar{q}_e + \frac{po(f_t,\mu_{\bar{e}})}{po(f_t,\mu_e)}(1-\bar{q}_e))(\frac{po(0,\mu_e)}{po(0,\mu_{\bar{e}})}q_e + (1-q_e))} \right) \tag{2.26}$$

とする．ここで，$\frac{po(f_t,\mu_{\bar{e}})}{po(f_t,\mu_e)} = \exp(\mu_e - \mu_{\bar{e}})(\frac{\mu_{\bar{e}}}{\mu_e})^{f_t}$ であり，$\mu_{\bar{e}} < \mu_e$ なので，$f_t \to \infty$ で $\frac{po(f_t,\mu_{\bar{e}})}{po(f_t,\mu_e)} \to 0$ となる．また，$\frac{po(0,\mu_e)}{po(0,\mu_{\bar{e}})} = \exp(\mu_{\bar{e}} - \mu_e)$ となる．よって，$f_t \to \infty$ となる t の項を w'_t と表したとき，式 (2.26) より，

$$w'_t = \log \left(\frac{q_e(\bar{q}_e \exp(\mu_{\bar{e}} - \mu_e) + (1-\bar{q}_e))}{\bar{q}_e(q_e \exp(\mu_{\bar{e}} - \mu_e) + (1-q_e))} \right) \tag{2.27}$$

となる．f_t に対して単調増加であることから，w'_t が最大値であることがわかる．

さらに，$\exp(\mu_{\bar{e}} - \mu_e)$ が十分小さく 0 であるとすると，

$$w'_t = \log \left(\frac{q_e(1-\bar{q}_e)}{\bar{q}_e(1-q_e)} \right) \tag{2.28}$$

となる．これは，ロバートソン/スパルク・ジョーンズ重み付け関数をトークンの出現から eliteness に置き換えたものと考えられる．

さて，式 (2.28) を式 (2.15) で近似することを考える．元々の式 (2.25) は，以下の性質を満たしていた．

1. $f_t = 0$ で 0
2. f_t が増加すると単調に増加する
3. 漸近的にはある最大値に収束しする

これらと，$f_t = 1$ の時にロバートソン/スパルク・ジョーンズ重み付け関数と同じ値を取るという性質を加えると，式 (2.28) の近似は

$$w'_t \approx \frac{f_{t,d}(k_1+1)}{k_1 + f_{t,d}} w_t \tag{2.29}$$

と考えられる．ここで，w_t は式 (2.15) で表されるロバートソン/スパルク・ジョーンズ重み付け関数である．また，$f_{t,d}$ は文書 d における単語 t の出現回数であり，

ここまで述べてきた f_t と同一である．

$\frac{f_{t,d}(k_1+1)}{k_1+f_{t,d}}$ は，$f_{t,d}=0$ で 0 となり，$f_{t,d}=1$ で 1 となる．$f_{t,d}$ の増加で増加し，$f_{t,d}\to\infty$ で k_1+1 に収束する．よって，期待された性質を満たしている．2.1.2 項で示したとおり，関連度の情報がない場合，IDF がロバートソン/スパルク・ジョーンズ重み付け関数の近似であったため，通常は w_t に IDF を用いる．

最後，文書の長さについて考慮する．文書が長い方が $f_{t,d}$ の値は大きくなる．そのため，

$$f'_{t,d}=f_{t,d}\left(\frac{l_{\mathrm{avg}}}{l_d}\right) \tag{2.30}$$

のように正規化することを考える．なお，l_d は文書 d の長さであり，l_{avg} は文書集合 D に対する l_d の平均値である．式 (2.29) の $f_{t,d}$ を $f'_{t,d}$ に置き換えると，

$$\hat{w}_t\approx\frac{f'_{t,d}(k_1+1)}{k_1+f'_{t,d}}w_t \tag{2.31}$$

$$=\frac{f_{t,d}(\frac{l_{\mathrm{avg}}}{l_d})(k_1+1)}{k_1+f_{t,d}(\frac{l_{\mathrm{avg}}}{l_d})}w_t \tag{2.32}$$

$$\approx\frac{f_{t,d}(k_1+1)}{k_1(\frac{l_d}{l_{\mathrm{avg}}})+f_{t,d}}w_t \tag{2.33}$$

となる．一方で，長い文書はより多くの情報を保持しているため，このような正規化をしない方が良い場合がある．そこで，単語頻度と文書の長さのバランスを調整するパラメータ $b(0\le b\le 1)$ を導入して，式 (2.34) のようにする．

$$\tilde{w}_t\sim\frac{f_{t,d}(k_1+1)}{(k_1((1-b)+b\frac{l_d}{l_{\mathrm{avg}}})+f_{t,d}}w_t \tag{2.34}$$

以上を統合し，クエリ側の単語頻度を考慮すると，BM25 の関連度スコア $S_{\mathrm{BM25}}(q,d)$ は式 (2.35) のようになる

$$S_{\mathrm{BM25}}(q,d)=\sum_{t\in q_i}\frac{f_{t,d_j}(k_1+1)}{k_1((1-b)+b\frac{l_d}{l_{\mathrm{avg}}})+f_{t,d_j})}\mathrm{IDF}_D(t) \tag{2.35}$$

これがBM25の式である．

以上より，BM25は単語頻度と文書の長さを考慮した，関連度情報が得られない場合のロバートソン/スパルク・ジョーンズ重み付け関数の近似であることがわかる．

2.2 BERTを用いたリランキングモデル

前節の検索モデルでは，以下二つの仮定があった．

- 文書中の単語の出現はそれぞれ独立である
- クエリ中の単語が文書に現れない場合，関連度に影響を与えない

しかし，これらの仮定が現実的ではない場面がある．文書は概ねあるトピックについて書かれると考えると，そのトピックに関する単語は同時に生起しやすいと考えられる．そのため文書中の単語の出現が独立とは言えない．また，例えば夕飯時に「レストラン」というクエリを入力した場合,「居酒屋」なども範疇に入ると考えられる場合があるが，居酒屋と書かれている文書は関連なしとなってしまう．そのため，クエリ中の単語が文書に現れない場合，関連度に影響を与えないという仮定も現実的ではない場面がある．

このような仮定を取り除く試みは意味検索 (Li and Xu 2014) として研究が行われてきた．近年はBERT (Devlin et al. 2019) などの事前学習済み言語モデルが出現し，言語現象に関わる仮定を全て言語モデルにより考慮する方法が用いられるようになってきている．

本節では，まずBERTについて簡単に述べたのち，BERTの検索への応用として最初に登場した，クロスエンコーダによるリランキングを紹介する．まず，記号について前節に追加して定義する．本節では，クエリ q はトークン列で表現され，その長さは l_q とする．よって，$\boldsymbol{q} = (t_1, t_2, ..., t_{l_q}) \in \mathcal{V}^{l_q}$ となる．文書 d も同様にトークン列で表現され，その長さは l_d とする．$\boldsymbol{d} = (t'_1, t'_2, ..., t'_{l_d}) \in \mathcal{V}^{l_d}$ となる．

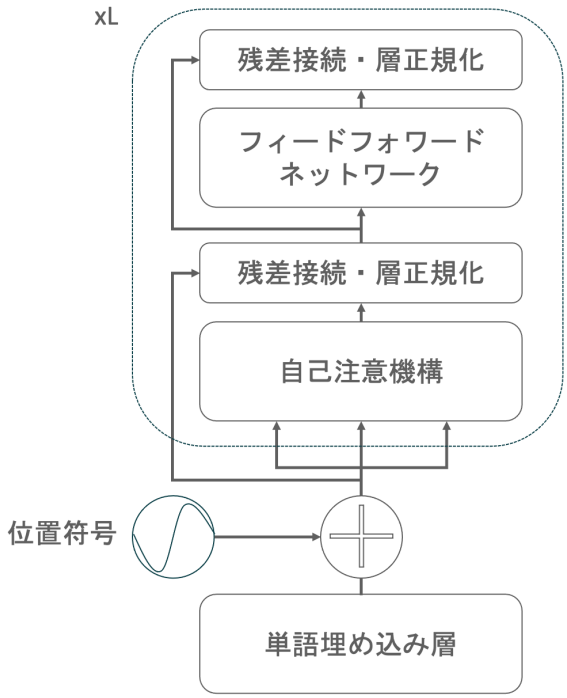


図 2.1: BERT で用いられるニューラルネットワークのアーキテクチャの模式図. Vaswani et al. (2017) を参考に作成

2.2.1 BERT

BERT は大規模なコーパスを用いて自己教師あり学習を行なったニューラルネットワークのモデルである．自己教師あり学習は，教師となるアノテーションのないデータを用いて教師信号を持つタスクを構成し，ニューラルネットワークのモデルを学習する方法であり，大量のデータを用いることが多い．BERT での自己教師あり学習のタスクとしては，文の一部を隠した上で推定するという単語穴埋め問題を解くマスク言語モデル（Masked LM）と入力したテキストが連続したものかどうかを推定する次文予測（Next Sentence Prediction）を用いる．ニューラルネットワークは Transformer (Vaswani et al. 2017) のエンコーダからなり，主な構成要素は，1. 自己注意機構（Self-attention），2. フィードフォワードネットワーク（FFN），3. 位置符号，4. 残差接続・層正規化がある．これらを一つのユニットとして，層を積み重ねたネットワークを用いる．概要を図 2.1に示す．以下，それぞれ説明を行う．なお，この説明の多くは 岡崎他 (2022) によるもので

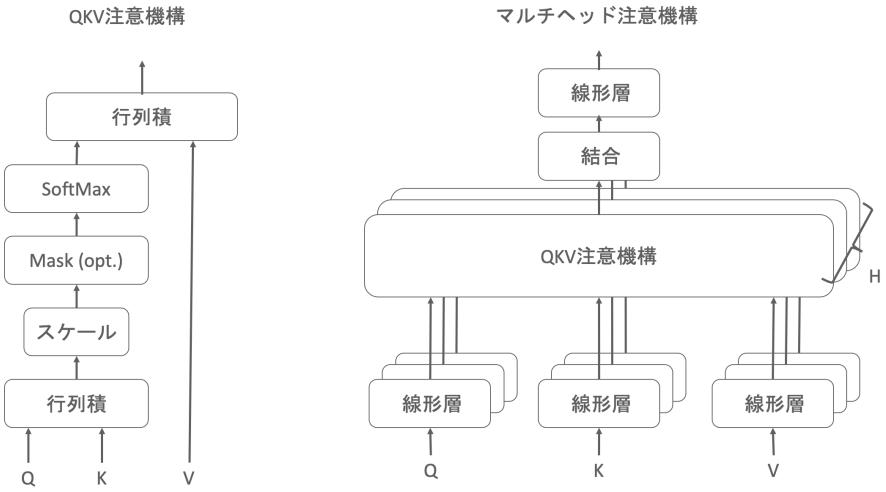


図 2.2: QKV 注意機構及びマルチヘッド注意機構の模式図．Vaswani et al. (2017) を参考に作成

あるため，詳細はそちらを参照されたい．

アーキテクチャ

自己注意機構　は，入力系列における他のベクトルを参照しながら，自身のベクトルを再構成する役割を担っている．そのため，文脈情報をエンコードしていることが期待される．BERT では，自己注意機構の計算にクエリ・キー・バリュー（query-key-value）型の注意機構を用いている．以降，頭字をとって QKV 注意機構と呼ぶ．今，入力系列を $\boldsymbol{x} = (x_1, ..., x_{l_x})$ し，ニューラルネットワークの潜在空間の次元を d とし，m 層のユニットへの入力されるベクトル系列を $\boldsymbol{H}^m = (\boldsymbol{h}_1^{(m)}, ..., \boldsymbol{h}_{|l_x|}^{(m)}) \in \mathbb{R}^{d \times l_x}$ とする．QKV 注意機構はクエリ行列 $\boldsymbol{Q} = \boldsymbol{W}^{(Q)}\boldsymbol{H}$，$\boldsymbol{K} = \boldsymbol{W}^{(K)}\boldsymbol{H}$，$\boldsymbol{V} = \boldsymbol{W}^{(V)}\boldsymbol{H}$ を用いて式 (2.36) のように計算される．

$$\hat{\boldsymbol{H}} = \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{V}\text{softmax}(c\boldsymbol{K}^T\boldsymbol{Q}) \tag{2.36}$$

なお，$\boldsymbol{W}^{(Q)} \in \mathbb{R}^{d \times d}, \boldsymbol{W}^{(K)} \in \mathbb{R}^{d \times d}, \boldsymbol{W}^{(V)} \in \mathbb{R}^{d \times d}$ は入力ベクトルをクエリ，キー，バリューに変換する行列である．また，c はスケーリング係数であり，$c = \frac{1}{\sqrt{d}}$ とすることで，次元数に応じたスケーリングを行う場合が多い．これにより，内積の値が大きくなりすぎることでソフトマックスの勾配が小さくなることを防ぐ．

さて，ソフトマックスは指数を取るため，元々の最大値を 1 に近づけ，それ以外を 0 に近づける効果が働く．そのため，自己注意機構において複数の観点で情報を取り出すことを難しくしている．これを解消する機構として，マルチヘッドアテンションが用いられる．マルチヘッドアテンションは式 (2.37) で表される．

$$\begin{aligned} \hat{\boldsymbol{H}} = \mathrm{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Concat}(\hat{\boldsymbol{H}}_1, \hat{\boldsymbol{H}}_2, ..., \hat{\boldsymbol{H}}_H)\boldsymbol{W}^{(O)} \\ \hat{\boldsymbol{H}}_i = \mathrm{Attention}(\boldsymbol{W}_i^{(Q)}\boldsymbol{Q}, \boldsymbol{W}_i^{(K)}\boldsymbol{K}, \boldsymbol{W}_i^{(V)\boldsymbol{V}}) \end{aligned} \tag{2.37}$$

ここで，$i \in 1, ..., H$，$\boldsymbol{W}_i^Q \in \mathbb{R}^{d_k \times d}, \boldsymbol{W}_i^K \in \mathbb{R}^{d_k \times d}, \boldsymbol{W}_i^V \in \mathbb{R}^{d_v \times d}, \boldsymbol{W}^{(O)} \in \mathbb{R}^{Hd_v \times d}$ は重み行列であり，H はヘッド数である．通常，$d_k = d_v = \frac{d}{H}$ とする．

フィードフォワード層　は，最も古典的なニューラルネットワークであり，二つの線形層と活性化関数で構成で構成される．自己注意機構の出力をを入力としている．今，その入力を $\hat{\boldsymbol{h}}_i$ として出力を $\bar{\boldsymbol{h}}_i$ とすると式 (2.38) で表現できる．

$$\begin{aligned} \bar{\boldsymbol{h}}_i &= \boldsymbol{W}_2\tilde{\boldsymbol{h}}_i + \boldsymbol{b}_2 \\ \tilde{\boldsymbol{h}}_i &= f(\boldsymbol{W}_1\hat{\boldsymbol{h}}_i + \boldsymbol{b}_1) \end{aligned} \tag{2.38}$$

ここで，$\boldsymbol{W}_1 \in \mathbb{R}^{d_f \times d}, \boldsymbol{b}_1 \in \mathbb{R}^{d_f}, \boldsymbol{W}_2 \in \mathbb{R}^{d \times d_f}, \boldsymbol{b}_2 \in \mathbb{R}^d$ はフィードフォワード層のパラメータである．中間層の次元 d_f はモデルの次元 d よりも大きい値を用いることが多く，4 倍の値を設定することが多い．bert-base[1] の場合，$d_f = 3072, d = 768$ である．関数 $f(\cdot)$ は活性化関数であり，通常は ReLU 関数が用いられる．

位置符号　は，単語の出現位置や順序情報をモデルに伝える役割を担う．自然言語処理では，これらの情報が重要な役割を果たす場合がある．入力系列の先頭から何番目の入力かをベクトルで表現する．モデルの次元 d に対して，位置 t の位置符号を $\boldsymbol{e}_t^{(pos)} = (e_{t,1}^{(pos)}, ..., e_{t,d}^{(pos)})^\top$ とする．Transformer の原論文 (Vaswani et al.

[1] https://huggingface.co/bert-base-uncased

2017) では，正弦関数を用いて，式 (2.39) のように表される．

$$e_{t,k}^{(pos)} = \begin{cases} \sin(\frac{t}{T^{(k-1)/d}}) & (t \text{ is odd}) \\ \cos(\frac{t}{T^{(k-2)/d}}) & (t \text{ is even}) \end{cases} \tag{2.39}$$

ただし，T は仮想的な最大系列長であり，Vaswani et al. (2017) では $T = 10000$ を用いている．

今，i 番目の単語の埋め込み表現を $\boldsymbol{e}_i^{(emb)}$ とすると，位置符号は式 (2.40) のように使用される．

$$\boldsymbol{h}_i^{(0)} = \sqrt{d}\boldsymbol{e}_i^{(emb)} + \boldsymbol{e}_i^{(pos)} \tag{2.40}$$

ここで，$\boldsymbol{h}_i^{(0)}$ は第一層目の入力となるベクトルである．なお，$\boldsymbol{e}_i^{(emb)}$ を $\sqrt{d}$ 倍するのはノルムを同程度に揃えるためである．

残差結合・層正規化　は，共に学習時の勾配に関する問題を解消するために用いられる．残差結合は勾配消失問題を緩和させる．T 個のベクトルの系列からなる入力系列 $\boldsymbol{P}^{(l-1)} = (\boldsymbol{p}_1^{(l-1)}, ..., \boldsymbol{p}_T^{(l-1)})$ に対して，l 番目の層が処理を行い，ベクトルの系列 $\boldsymbol{O}^{(l)} = (\boldsymbol{o}_1^{(l)}, ..., \boldsymbol{o}_T^{(l)})$ が出力されるとする．この時，系列の t 番目において，残差結合は式 (2.41) のように表せる．

$$\boldsymbol{p}_t^{(l)} = \boldsymbol{p}_t^{(l-1)} + \boldsymbol{o}_t^{(l-1)} \tag{2.41}$$

残差結合を用いることで，最終層に多くの層が直接接続されることと同等になり，勾配消失が起こりにくくなると考えられる．なぜならば，$\boldsymbol{p}_t^{(0)} = \boldsymbol{o}_t^{(0)}$ として，式 (2.41) を再帰的に適用すると，

$$\boldsymbol{p}_t^{(l)} = \sum_{i=0}^{l} \boldsymbol{o}_t^{(i)} \tag{2.42}$$

となるためである．

層正規化はベクトルの各要素を正規化することで，勾配爆発を防ぐ．正規化

前の d 次元ベクトルを $\boldsymbol{x}$ とすると，層正規化は式 (2.43) によってなされる．

$$x_k^{'} = a_k \left(\frac{x_k - \mu_x}{\sqrt{\sigma_x^2 + \epsilon}} \right) + b_k \tag{2.43}$$

なお，$\boldsymbol{a} = (a_1, ..., a_k, ..., a_d)^\top$ と $\boldsymbol{b} = (b_1, ..., b_k, ..., b_d)^\top$ はそれぞれはスケールパラメータ，シフトパラメータと呼ばれる．共に学習可能なパラメータである．ϵ はハイパーパラメータであり，$\epsilon = 1.0 \times 10^{-6}$ などが用いられる．さらに，μ_x と σ_x はそれぞれ，$\boldsymbol{x}$ の平均と標準偏差である．これらは式 (2.44) のように計算される．

$$\mu_x = \frac{1}{d} \sum_{k=1}^{d} x_k, \qquad \sigma_x = \sqrt{\frac{1}{d} \sum_{k=1}^{d} (x_k - \mu_x)^2} \tag{2.44}$$

オリジナルの Transformer で注意機構の出力やフィードフォワード層の出力に対して残差結合したのちに層正規化を行っているが，BERT では，それぞれの入力前に層正規化を行っている．これは使用する層数の違いによるものである．Transformer の層正規化は事後層正規化と呼ばれており，事後層正規化は層数が多いと不安定になるものの性能は良いとされている．一方，BERT での層正規化は事前層正規化と呼ばれており，事前層正規化は性能は劣るものの層数が多くても安定して学習できるとされている．

事前学習

BERT は，前述のニューラルネットワークのアーキテクチャに対して，大規模なコーパスを用いて自己教師あり学習を行なったモデルである．自己教師あり学習のタスクとしてマスク言語モデルと次文予測を用いる．以下，それぞれ説明する．

マスク言語モデル　は，トークンの一部をランダムに [MASK] という特殊トークンに置換し，その特殊トークンに対して最終層のベクトルと埋め込み層を用いて元々のトークンを予測するタスクである．

今，トークン系列を $s = (x_1, x_2, ..., x_N)$ とし，最終層の出力を $\boldsymbol{h}_i^{(L)} \in \mathbb{R}^d$ とす

ると，

$$P(x_i|x_1, ..., x_{i-1}, x_{i+1}, ..., x_N) = \text{softmax}_{x_i}(\boldsymbol{W}_{xh} h_i^L) \tag{2.45}$$

として表される．$\boldsymbol{W} \in R^{|V| \times d}$ は埋め込み層の重み行列である．また，$|V|$ はモデルの語彙数である．

特殊トークンへの置換は，テキスト中の 15%のトークンをランダムに選んだ上で，その 80%を [MASK] トークンに置換し，10%はランダムな単語に置換し，残りの 10%は置換しないでそのままとする．

次文予測　は，BERT に入力された二つのパッセージが連続したものかランダムに選ばれたものかを判別するタスクである．BERT では，入力トークンの先頭に特殊トークン [CLS] を入力する．この [CLS] の最終層の埋め込み表現を用いて，2 値分類を行う．これによって，テキストの一貫性をモデルに学習させることを狙っている．しかし，二つのパッセージが同じトピックかどうかで判定できるため，マスク言語モデルで同じような学習が可能であるとも言われている (Liu et al. 2019)．

自然言語処理における BERT の主な利用方法

事前学習を通じて得られた BERT のモデルは，特定の自然言語処理タスクに対して学習を行うことで，精度を向上させる．このように特定のタスクに対して学習することを，ファインチューニングと呼ぶ．例えば，感情分類や文書分類など文書分類タスクや，文類似度推定や含意関係認識など文のペアを入力とするタスクに対しても用いられる．これらは [CLS] トークンの埋め込みを入力全体の表現とみなして，その埋め込み表現を分類器となるニューラルネットワークに入力し，学習を行う．また，質問応答や固有表現抽出では，最終層の各トークンの埋め込みに対してラベルを予測することでタスクを実施する．質問応答では回答の開始と終了を予測するタスクとし，固有表現抽出では固有表現のクラスと開始・継続．終了のラベルとなる．BERT でエンコードされたベクトルには品詞や依存構造などの文法に関する情報 (Tenney et al. 2019, Hewitt and Manning 2019) が埋め込まれているため，BERT を用いることが有効となる．さらに，BERT をエンコー

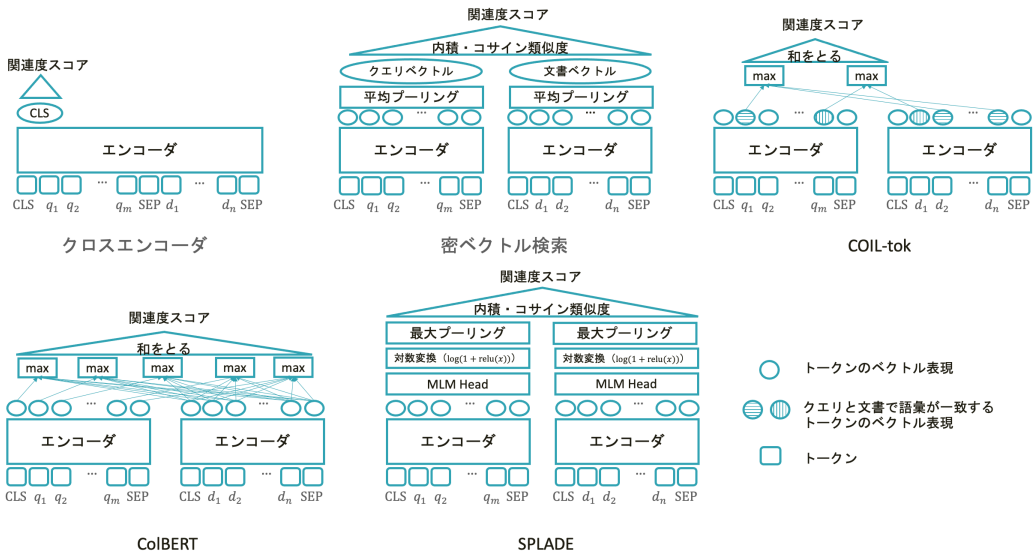


図 2.3: BERT を用いた検索モデルのアーキテクチャの模式図

ダとして，単語や文の埋め込み表現を用いることで文類似度タスクなどを実行することも可能である．

次項では分類と同様の方法でリランキングモデルとして学習する方法を述べる．その後，次節以降で，検索モデルとして学習する方法を述べる．

2.2.2 クロスエンコーダ

クロスエンコーダは，クエリと文書を入力として受け取り，その関連度を出力するモデル (Nogueira and Cho 2019) であり，最初に提案された検索における利用方法である．利用方法としては，他の自然言語処理の 2 文タスクと同様にクエリと文書を文末トークンで結合し，文頭のトークンで全体を表す特徴としたのち，関連の有無を判別する．模式図を図 2.3に示す．

BERT の文頭トークンは CLS というトークンであるため，式 (2.46) ように定式化できる．

$$p(R=1|Q=q, D=d_i) = S_{\mathrm{CE}}(q, d_i) := \mathrm{softmax}(\boldsymbol{W}\mathrm{BERT}(\boldsymbol{q}, \boldsymbol{d}_i)_{\mathrm{CLS}} + \boldsymbol{b}) \tag{2.46}$$

ただし，$\boldsymbol{W} \in \mathbb{R}^{2\times d}$ は d 次元の埋め込みをスコアに変換する層であり，$\boldsymbol{b} \in \mathbb{R}^2$ はバイアス項である．関連の有無を判別するため学習は，クエリ q に対する正例文書のインデックス集合を J^q_{pos} として負例となる文書のインデックス集合を J^q_{neg} とすると式 (2.47) の損失関数で行われる．

$$\mathcal{L}_{CE} = -\sum_{q\in Q}\left\{\sum_{i\in J_{pos}} \log S_{\mathrm{CE}}(q, d_i) + \sum_{i\in J_{neg}} \log(1 - S_{\mathrm{CE}}(q, d_i))\right\} \tag{2.47}$$

なお，検索問題であることから，PairwiseLoss なども用いられる (Han et al. 2020)．多数の先行研究で，クロスエンコーダは教師データが存在する場合に，BM25 を精度で上回ることが知られている (Nogueira and Cho 2019, Qiao et al. 2019, MacAvaney et al. 2019, Yang et al. 2019, Han et al. 2020)．また，大量の教師データが存在するデータセットで学習したモデルであれば，対象のデータに教師がない場合でも良い精度を示すことも知られている (Thakur et al. 2021)．一方で，クエリと文書のペアから関連度スコアを都度推定するため，計算コストが非常に高いという問題が指定されている (Hofstätter and Hanbury 2019, Khattab and Zaharia 2020)．そのため，クロスエンコーダは，あらかじめ高速な検索方法で上位 K 件を検索文書をリランキングするために用いられることが多い．

式 (2.46) の定式化や BERT の性質を考慮すると，BM25 導出の際の仮定された文書中のトークンの独立性やクエリと文書で共通して現れるトークンのみが関連度に影響するという仮定がなくなっていると考えられる．つまり，言語現象に関わる仮定を BERT によって一体で考えているとみなせる．実際，Dai and Callan (2019) では where に対して，場所を表すトークンとのマッチングがとられていると述べられており，クエリと文書で共通して現れるトークン以外が関連度に影響を与えていることがわかる．なお，検索タスクの種類によってどのような言語現象が重要になるかについては，同義語は多くの検索問題で共通するものの，その他は異なることが知られている (Fan et al. 2021)．

2.3 BERT を用いた検索モデル

前節で述べたクロスエンコーダはリランキングに用いられており，上位 K 件の候補の検索は従来通り BM25 を使用することが多い．そのため，キーワードが一致

しない場合には，上位 K 件から漏れる場合がある．一方，その漏れを防止するために，K の値を大きくすると，遅延が大きくなり，最悪の場合ユーザの不満に至る．そこで，文書全件の検索にも BERT を用いるようなモデルが提案されている．本節では中でも広く用いられている，密ベクトル検索 (Karpukhin et al. 2020, Xiong et al. 2021, Hofstätter et al. 2021)，SPLADE (Formal et al. 2021, 2022b)，ColBERT (Khattab and Zaharia 2020, Santhanam et al. 2022)，COIL (Gao et al. 2021b) について説明する．どのモデルにおいても，文書を予めエンコードし，インデックスを作成しておくことで，高速な検索を実現している．

これを実現するために，エンコーダはクエリと文書を別々にエンコードする．このエンコーダを $\eta : \mathcal{V}^n \to \mathbb{R}^{n\times h}$ と表記する．トークンをベクトルに変換するエンコーダは BERT などが用いられる．クエリのトークン列 $\boldsymbol{q} = (t_1, t_2, ..., t_{l_q}) \in \mathcal{V}^{l_q}$ をエンコーダを通じてエンコードしたベクトル列は $\boldsymbol{Q} = \eta(\boldsymbol{q})$ であり，$\boldsymbol{Q} = (\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_{l_q}) \in \mathbb{R}^{d\times l_q}$ である．同様に，文書のトークン列 $\boldsymbol{d} = (t'_1, t'_2, ..., t'_{l_d}) \in \mathcal{V}^{l_d}$ をエンコーダを通じてエンコードしたベクトル列は $\boldsymbol{D} = \eta(\boldsymbol{d})$ であり，$\boldsymbol{D} = (\boldsymbol{h}'_1, \boldsymbol{h}'_2, ..., \boldsymbol{h}'_{l_d}) \in \mathbb{R}^{d\times l_d}$ である．この時，本節で述べる検索モデルは式 (2.48) のように確率モデルで表せる．

$$p(R = 1|Q = q, D = d) \propto S := \phi(\hat{\eta}(q), \hat{\eta}(d)) \tag{2.48}$$

$\phi : \mathbb{R}^{n\times h} \times \mathbb{R}^{n\times h} \to \mathbb{R}$ は，エンコードされた各ベクトルを関連度スコアに変換する関数である．また，$\hat{\eta}$ は η の入力をテキストとした場合である．つまり，テキストのトークナイズとエンコーダを合わせたものである．

学習方法については，2.4 節で説明し，本節では関連度スコア S の算出方法について説明する．以下にて述べる検索モデルの模式図は図 2.3に示す．

2.3.1 密ベクトル検索

密ベクトル検索は，クエリと文書を潜在空間上の密なベクトルで表現し，その類似度を関連度スコアとする検索方法である．クエリ・文書を一つのベクトルで表現するため，エンコードされた各トークンのベクトルにプーリングを施す．プーリングには，BERT の CLS トークンを用いる CLS プーリング，各トークンのベクトルをトークン方向に平均をとった平均プーリング，トークン方向に最大値を

とった最大値プーリングなどが使用される．また，関連度スコアにはクエリと文書のベクトルの内積やコサイン類似度が用いられる．密ベクトル検索モデルの関連度スコアを $S_{\text{Dense}}(q, d)$ とすると，平均プーリングと内積を用いた場合の密ベクトル検索の式は，式 (2.49) の様になる．

$$\begin{aligned} S_{\text{Dense}}(q, d) &= \phi_{\text{Dense}}(\hat{\eta}(q), \hat{\eta}(d)) = \bar{\boldsymbol{q}}^T \bar{\boldsymbol{d}} \\ \bar{\boldsymbol{q}} &= \text{pooling}(\hat{\eta}(q)) \quad \bar{\boldsymbol{d}} = \text{pooling}(\hat{\eta}(d)) \end{aligned} \tag{2.49}$$

2.3.2 SPLADE

SPLADE は，クエリと文書を事前学習済み言語モデルの語彙サイズ次元のベクトルで表現し，その類似度を関連度スコアとする検索方法である．クエリ・文書の各トークンをベクトルにエンコードした後，単語予測層を通じて，事前学習済み言語モデルの語彙サイズ次元のベクトルに変換する．入力した各トークン以外の要素も 0 以外の値を持つことから，クエリ拡張・文書拡張を行うモデルとみなすことができる．式で表すと式 (2.50) の様になる．

$$\boldsymbol{v}_i = \boldsymbol{E} f(\boldsymbol{h}_i) + \boldsymbol{b} \qquad \boldsymbol{v}'_i = \boldsymbol{E} f(\boldsymbol{h}'_i) + \boldsymbol{b} \tag{2.50}$$

なお，$\boldsymbol{E} \in \mathbb{R}^{|\mathcal{V}'| \times h}$ は，語彙が $\mathcal{V}'$ である事前学習済み言語モデルの単語埋め込み行列である．$f : \mathbb{R}^h \to \mathbb{R}^h$ は，GeLU と層正規化層を持つ線型層であり，$\boldsymbol{b} \in \mathbb{R}^{|\mathcal{V}'|}$ はバイアス項である．この語彙次元のベクトルに対して対数変換を行い，トークン方向に最大値プーリングを取ることで，ベクトル表現を得る．

$$\boldsymbol{v} = \max_{1 \le i \le |\boldsymbol{q}|} \log(\boldsymbol{1} + \text{ReLU}(\boldsymbol{v}_i)) \quad \boldsymbol{v}' = \max_{1 \le i \le |\boldsymbol{d}|} \log(\boldsymbol{1} + \text{ReLU}(\boldsymbol{v}'_i)) \tag{2.51}$$

最後に内積を用いて SPLADE の関連度スコア $S_{\text{SPLADE}}(q, d)$ を得る．

$$S_{\text{SPLADE}}(q, d) = \phi_{\text{SPLADE}}(\hat{\eta}(q), \hat{\eta}(d)) = \boldsymbol{v}^T \boldsymbol{v}' \tag{2.52}$$

なお，Formal et al. (2021) は，クエリのエンコードにかかる計算コストをよ

り下げる方法として，クエリにはBag-of-Words表現を用いるSPLADE-Docを提案している．関連度スコア $S_{\text{SPLADE-Doc}}(q, d)$ は，式 (2.53) のような式となる．

$$S_{\text{SPLADE-Doc}}(q, d) = \sum_{t \in \mathcal{V}'} \mathbf{v}'_t \tag{2.53}$$

SPLADEは学習時にベクトルを疎にするために，FLOPS (Paria et al. 2020) と呼ばれる正則化項用いて学習する．FLOPSではクエリと文書双方のエンコーダに対して正則化を行う．クエリ側の正則化項を $\mathcal{L}^q_{\text{FLOPS}}$，文書側の正則化項を $\mathcal{L}^d_{\text{FLOPS}}$ とする．SPLADE学習時の損失を $\mathcal{L}_{\text{SPLADE}}$，損失関数による損失を $\mathcal{L}$ とすると，

$$\mathcal{L}_{\text{SPLADE}} = \mathcal{L} + \lambda_q \mathcal{L}^q_{\text{FLOPS}} + \lambda_d \mathcal{L}^d_{\text{FLOPS}} \tag{2.54}$$

$$\mathcal{L}^q_{\text{FLOPS}} = \sum_{j=1}^{\mathcal{V}'} \sum_{i=1}^{\mathcal{B}} v_{ij} \quad \mathcal{L}^d_{\text{FLOPS}} = \sum_{j=1}^{\mathcal{V}'} \sum_{i=1}^{\mathcal{B}} v'_{ij} \tag{2.55}$$

となる．ここで，$\mathcal{B}$ はバッチサイズであり，v_{ij} はバッチ中のクエリ i のベクトル $\boldsymbol{v}$ の要素 j であり，v'_{ij} はバッチ中の文書 i のベクトル $\boldsymbol{v}'$ の要素 j である．また，λ_q と λ_d は正則化の係数である．

2.3.3 ColBERT

密ベクトル検索やSPLADEはクエリと文書をそれぞれ一つのベクトルとして表現する方法であった．しかし，これらの方法は内積によって関連度スコアを計算するため，クロスエンコーダでなされていたような，トークンレベルでの相互作用が考慮できていない．これを，クロスエンコーダよりも高速にかつ同等の精度で実現する方法がColBERTである．

ColBERTにおける関連度スコアリングはクエリの各トークン毎に類似度が最大になる文書のトークンとの類似度を計算し，その総和をとることで行う．これは，クロスエンコーダにおいて各層において計算されていたトークン毎の相互作用を最終層のベクトルのみで行い，相互作用の計算も自己注意機構から内積の最

大値に置き換えている．式で表すと，ColBERT の関連度スコア $S_{\text{ColBERT}}(q, d)$ は，

$$S_{\text{ColBERT}}(q, d) = \phi_{\text{ColBERT}}(\hat{\eta}(q), \hat{\eta}(d)) = \sum_{i=1}^{l_q} \max_{1 \leq j \leq l_d} (h_i^T h'_j) \tag{2.56}$$

と計算できる．なお，実際は内積ではなくコサイン類似度を用いている．また，エンコードされたベクトルに線形層を入れて次元を小さくしている．

ColBERT もクエリと文書を分離してエンコードしているため，文書を予めエンコードしておくことで高速化可能である．また，全トークンのベクトルをインデクシングすることで，全件検索を可能としている．その際，近似最近傍探索を用いることで高速化を実現すると共に，離散化によってベクトルを圧縮 (Santhanam et al. 2022) することによりインデックスを実用的なメモリサイズに収めている．

2.3.4 COIL

ColBERT ではクエリと文書の各トークンの出力層のすべてのベクトルを用いて内積を計算していた．しかし，文書が長くなるとそれでも非常に計算量が大きくなる．これをさらに軽減するために，クエリと文書の双方で出現する単語のみで関連度を計算するのが COIL である．

COIL では，ColBERT のようにクエリのトークンのベクトルと文書のトークンのベクトルの内積を計算するが，その計算は，クエリと文書の両方に出現するトークンのみで行う．Gao et al. (2021b) では，密ベクトル検索も用いており，両方に出現するトークンのベクトルの内積部分を COIL-tok と呼んでいる．今，COIL のスコアを $S_{\text{COIL}}(q, d)$，COIL-tok のスコアを $S_{\text{COIL-tok}}(q, d)$ とすると，

$$S_{\text{COIL-tok}}(q, d) = \phi_{\text{COIL-tok}}(\hat{\eta}(q), \hat{\eta}(d)) = \sum_{i} \max_{1 \leq j \leq l_d} (h_i^T h'_j) \tag{2.57}$$

$$S_{\text{COIL}}(q, d) = S_{\text{Dense}}(q, d) + S_{\text{COIL-tok}}(q, d) \tag{2.58}$$

ただし，COIL の場合は密ベクトル検索のプーリング方式は CLS プーリングである．また，エンコードされたトークンのベクトルに線形層を入れて次元を小さしている．

COIL はクエリと文書で一致するトークンのみに計算を限定することで計算量を減らすと共に，転置インデックスを用いることで高速な検索を実現している．また，COIL は検索対象文書が長い場合に，高い検索精度を示すことが明らかになっている (Gao et al. 2021b)．

2.4 BERT を用いた検索モデルの学習方法

本節では，BERT を用いた検索モデルの学習方法について述べる．

2.4.1 交差エントロピー損失を用いた対照学習

式 (2.48) より，検索モデルの学習は，関連の有無を判別することによって実施可能であることがわかる．しかし，2.3 節で述べた検索モデルの場合は，関連度の有無を直接推定することはできない．そのため，検索モデルのスコアを用いてクエリと関連文書を近づけ非関連文書を遠ざける学習を行う．このような学習は対照学習と呼ばれる[2]．

具体的な学習方法であるが，まずクエリ q と文書 d に対する検索モデルのスコアを $S(q, d)$ とする．また，クエリ q に対する正例である関連文書のインデックス集合を J^q_{pos} として負例となる非関連文書のインデックス集合を J^q_{neg} とすると，式 (2.59) で表現される．

$$\mathcal{L}_{\mathrm{CEnt}} = \sum_{q \in Q} \sum_{i \in J^q_{pos}} -\log \frac{\exp(S(q, d_i))}{\exp(S(q, d_i)) + \sum_{j \in J^q_{neg}} \exp(S(q, d_j))} \tag{2.59}$$

ただし，N は文書集合における文書数である．また，この式は $p(R = 1|Q = q, D = d)$ を $p(D = d|Q = q, R = 1)$ に近似していると考えれる．つまり，クエリに対して関連文書 d_i を正例として負例と判別する分類問題と捉えられる．

[2]対照学習は通常一つのデータ (検索ではクエリや文書) が一つのベクトルとして表現される場合に用いられる．ColBERT や Coil は一つのベクトルとして表されるわけではないが，本論文では同様に対照学習と呼ぶ．

2.4.2 負例サンプリング

検索では，交差エントロピーを使用する場合に文書集合中のすべての負例文書を考慮すると，負例が多くなりすぎるため計算量が大きくなる．そのため，負例を一部に限定して学習を行う．既存研究では，主に以下が用いられる．

バッチ内負例 同一バッチ内で，特定の正例ペア以外を負例として扱う方法である (Karpukhin et al. 2020)．ある正例ペアについて，他のペアにおける正例文書は負例と見なされる．また，他のペアにおける負例文書も負例として扱う．

BM25 負例 BM25 の検索結果の上位 K 件のうち，関連文書ではない文書を負例とみなす方法である (Karpukhin et al. 2020, Gao et al. 2021a)．

自己負例 自らの検索結果の上位 K 件のうち，関連文書ではない文書を負例とみなす方法である (Xiong et al. 2021, Zhan et al. 2021)．BM25 負例と併せて，判別の難しい負例と考えられている．なお，近年は BM25 負例を使用しない場合も見られる (Shen et al. 2023)．

2.4.3 クロスエンコーダを用いた知識蒸留

検索の教師データ作成は非常にコストが高く，すべての関連文書が網羅的にアノテーションされているとは言えない．そのため，本来関連文書である文書がアノテーションされておらず，学習時に関連なしとみなされる場合がある．

このような場合を緩和するために，知識蒸留が用いられる．知識蒸留は，高精度なモデルの推定結果を教師データとして，知識蒸留元とは異なるモデルを学習することで，高精度なモデルから知識を転移する手法を指す．検索では最も高い精度を誇るクロスエンコーダの関連度スコアを学習に用いることで，より正確に表現された関連度で検索モデルを学習する方法が提案されている (Hofstätter et al. 2020, Ren et al. 2021)．

マージン二乗誤差 Hofstätter et al. (2020) は検索モデルにおける蒸留方法として，マージン二乗誤差を提案している．マージン二乗誤差は，蒸留元となる検索モデルにおける正例と負例のスコア差と，蒸留先となる検索モデルにおける正例・

負例のスコア差の二乗誤差のスコアの二乗誤差を損失とする．具体的には，クエリ q に対して正例を d_i^+ とし負例を d_j^- とする．また，クエリ q に対する正例文書のインデックス集合を J_{pos}^q として負例となる文書のインデックス集合を J_{neg}^q とする．クロスエンコーダにおける正例と負例のスコア差 $\delta(q, d_i^+, d_j^-)$ は

$$\delta(q, d^+, d_i^-) = S_{\mathrm{CE}}(q, d_i^+) - S_{\mathrm{CE}}(q, d_j^-). \tag{2.60}$$

となる．また，蒸留先の検索モデル M におけるスコア $S_M(q, d)$ とする．正例と負例のスコア差 $\hat{\delta}(q, d_i^+, d_j^-)$ は

$$\hat{\delta}(q, d_i^+, d_j^-) = S_M(q, d_i^+) - S_M(q, d_j^-). \tag{2.61}$$

となる．この時，マージン二乗誤差は

$$\mathcal{L}_{\text{Margin-MSE}} = \sum_{q \in Q} \sum_{i \in J_{pos}^q} \sum_{j \in J_{neg}^q} (\delta(q, d_i^+, d_j^-) - \hat{\delta}(q, d_i^+, d_j^-))^2. \tag{2.62}$$

となる．

KL ダイバージェンスによる蒸留　Ren et al. (2021) は，KL ダイバージェンスを用いた蒸留方法を提案している．クエリ q に対する蒸留先検索モデル M による検索結果上位 K 件の文書集合 $\tilde{D}_q$ に対して，クロスエンコーダのスコアを用いて確率分布 $\tilde{s}_{CE}$ を式 (2.63) のように計算する．

$$\tilde{s}_{CE}(q, d) = \frac{\exp(S_{CE}(q, d))}{\sum_{d' \in \tilde{D}_q} \exp(S_{CE}(q, d'))} \tag{2.63}$$

また，同様に蒸留先の検索モデル M のスコアを用いて確率分布 $\tilde{s}_M$ を式 (2.64) のように計算する．

$$\tilde{s}_M(q, d) = \frac{\exp(S_M(q, d))}{\sum_{d' \in \tilde{D}_q} \exp(S_M(q, d'))} \tag{2.64}$$

この時，KL ダイバージェンスによる蒸留の損失は

$$\mathcal{L}_{\mathrm{KL}} = \sum_{q \in Q} \sum_{d \in \tilde{D}_q} \tilde{s}_{CE}(q, d) \log \frac{\tilde{s}_{CE}(q, d)}{\tilde{s}_M(q, d)} \tag{2.65}$$

となる．

2.5 検索における教師なしドメイン適応

本節では，検索における教師なしドメイン適応について述べる．ドメインとは，データの出所のことを指し，数学的には確率分布として考えられている．検索で考えると，例えばある検索システム $\mathcal{A}$ において，入力されるクエリ全体を Q，検索対象文書全体を D，関連度を R とすると $p_{\mathcal{A}}(R, Q, D)$ である．ドメイン適応は大量に教師データが存在するデータセット（ソースドメイン）において学習を行い，同種のタスクで異なるデータセット（対象ドメイン）にモデルを適用する場合に，その精度の低下を小さくする枠組みである．ソースドメインと対象ドメインの間のドメインの変化のことをドメインシフトと呼ぶ．対象ドメインでは少量の教師データの存在するか教師データが存在しない場合を想定することが多い．後者の場合は特に教師なしドメイン適応と呼ばれる．

事前学習済み言語モデルを用いた検索モデルのドメイン適応として，まず，本論文の提案手法と同種である対象データにおけるトークンの重要度を用いる手法と事前学習済み言語モデルに対するドメイン適応について述べる．次に，ドメイン敵対的学習を用いる手法，疑似クエリを用いる手法について述べる．最後に，ドメイン適応以外でゼロショット検索を実現する手法について述べる．

2.5.1 対象データにおけるトークンの重要度を用いる手法

対象データにおけるトークンの重要度を用いる手法として，事前学習済み言語モデルを用いた検索モデルの関連度スコアと BM25 の関連度スコアの和または積を用いる手法がある (Ma et al. 2021, Formal et al. 2022b, Xu et al. 2022)．本研究では，この手法を BM25 併用と呼び，和をとる場合を用いる．この方法は次項以降で述べるような他手法と異なり，学習を行わない．BM25 併用の精度向上要因は，対象データにおいて，クエリ中のキーワードと完全一致する文書が上位にラ

ンキングされないという問題 (Formal et al. 2022a) を解決するためであると考えられる．この方法は BM25 という検索特有スコアリング方法を用いるため，検索タスク特有のドメイン適応手法と考えられる．

2.5.2 事前学習済み言語モデルに対するドメイン適応

BERT などのニューラルネットワークを用いる言語モデルの事前学習は Wikipedia や Web 上の文書などの一般のコーパスを用いて行われる．BERT の事前学習の一つであるマスク言語モデルは，単語ベクトル (Mikolov et al. 2013) の学習と同様に単語穴埋め問題と見做すことができる．そのため，単語の共起を学習している (Levy and Goldberg 2014) と考えられる．

このように考えると，対象としているタスクがある特定のドメインの場合に，そのドメインで出現する単語は大きく異なると予想されるため，共起する単語も異なることが予想される．また，BERT はサブワード (Mike and Kaisuke 2012, Sennrich et al. 2016, Kudo 2018) という単語より細かい単位の語彙を用いることで，語彙をある程度のサイズ（約 32000 語）に制限しながらも未知語を減らしている．そのため，ドメイン特有の単語は通常の BERT では学習コーパス中で頻度が低いため，サブワードに分解される傾向がある．

このような課題を解決するための，事前学習済み言語モデルに対するドメイン適応方法として，対象ドメインと類似したコーパスを用いて BERT の事前学習を行う方法と，すでに事前学習された BERT に対して対象ドメインと類似したコーパスを用いて継続事前学習を行う方法がある．どちらの方法も，より対象ドメインに近いコーパスで事前学習を行うことが，効果的であるという発想に基づく．以下では，それぞれの方法を紹介し，言語処理及び検索における効果を述べる．

対象ドメインのコーパスを用いた事前学習

この手法は文字通り，対象とするタスクと類似したドメインのコーパスを用いて事前学習を行うことで，ドメインに特化した事前学習済み言語モデルを作成する方法である．例えば，科学技術ドメインのコーパスで学習した SciBERT (Beltagy et al. 2019) やバイオ・医療ドメインで学習した PubMedBERT (Gu et al. 2021) などがある．これらはそれぞれ，事前学習を行ったドメインにおける自然言語処理のタ

スクで通常の BERT より高い性能を示すことが知られている．

検索においては，ドメイン特化した事前学習済み言語モデルを用いることで，リランキングモデルの検索精度が向上することが知られている (MacAvaney et al. 2020)．しかしながら，検索モデルにおいてはその性質は明らかになっていない．本論文では，4 章にて検索モデルである SPLADE に対して，SciBERT と PubMedBERT を用いて実験を行っている．

継続事前学習

あるタスクで学習済みの１つのニューラルネットワークに対して，新しいタスクを追加的に学習する試みは継続学習と呼ばれている (岡谷 2022)．継続事前学習は，すでに事前学習を行ったモデルに対して，新たなデータを用いて改めて事前学習を行うことを指す．継続事前学習はドメイン適応として用いられており，対象のデータセットまたはそのデータセット近いコーパスを用いて事前学習を行い，その後，対象となるタスクである文書分類や固有表現抽出などで学習させることで，対象ドメインで精度を向上させることができる (Gururangan et al. 2020, Ke et al. 2023)．

継続事前学習は， 事前学習と比較して比較的少量のコーパスの場合でも精度を向上させることができる点が特徴である (Gururangan et al. 2020, Zhang et al. 2020)．一方，対象ドメインのコーパスを用いた事前学習と比較し，対象とするドメインにおけるタスクの精度向上幅は小さい傾向にある (Gu et al. 2021)．精度向上幅が小さくなる理由の一つは，事前学習済み言語モデルの語彙にあると考えてられており，実際，対象ドメインの語彙を追加することで，継続事前学習はより精度が向上することが知られている (Yao et al. 2021, Zhang et al. 2020)

上記の性質は，対象データに教師データが存在する場合に，自然言語処理のタスクにおいて示されてきた．本論文では，4 章にて検索モデルである SPLADE に対して，語彙を追加する効果を検証している．また，本論文では先行研究と異なり，対象データに教師データが存在しない場合を取り扱う．

自然言語処理のタスクで有効な手法が検索において有効ではないことがある．自然言語処理において，教師なしドメイン適応手法として UDALM (Karouzos et al. 2021) という手法がある．UDALM は，対象ドメインでマスク言語モデルによる継続事前学習を行ったのち，ソースドメインでタスクの学習をしながら，対象ド

メインのデータで同時に継続事前学習を行う．UDALM の損失関数を L_{UDALM} として，あるデータに対するタスクの損失関数を $l_{\mathrm{TASK}}(x_i)$，マスク言語モデルの損失関数を $l_{\mathrm{MLM}}(x_i)$ とすると

$$L(x) = \frac{n}{n+n'} \sum_{i=1}^{n} l_{\mathrm{TASK}}(x_i) + \frac{n'}{n+n'} \lambda \sum_{i=n+1}^{n+n'} l_{\mathrm{MLM}}(x_i) \tag{2.66}$$

となる．つまり，対象ドメインのデータで継続事前学習で正則化を行なっていると考えられる．Karouzos et al. (2021) は評判分類のデータセットである Amazon Rewiews (Blitzer et al. 2007) を用いて実験を行なった．このデータセットは評判分類というタスクは変化しないが，テキストのドメインは本，家電，キッチン用品など多岐に渡るため，あるドメインをソースドメインとし，他のドメインを各対象ドメインとしている．結果，継続事前学習のみの場合や 2.5.3 項に述べるドメイン敵対的学習を用いる場合よりも平均的に良い精度であることを示した．一方，検索タスクにおいては，UDALM が疑似クエリによる手法に劣ることが示されている (Wang et al. 2022).

2.5.3 ドメイン敵対的学習

ドメイン敵対的学習（DAT: Domain Adversarial Training） (Ganin et al. 2017) は，通常の学習に加えて，ドメイン分類器を導入することで，ドメイン不変な表現を学習する手法である．ニューラルネットワークを前提にした場合，ネットワークの層の途中までを特徴量抽出器として用いる．その後タスクを実行する部分とドメインを判別する部分に分ける．そして，ドメイン分類器の結果を敵対的に用いることで，ドメイン不変な表現を学習する．概要を図 2.4に示す.

この学習を数式を用いて記載する．今，ソースドメイン $\mathcal{S}$ から n 個のデータがサンプルされ，対象ドメイン $\mathcal{T}$ から n' 個のデータがサンプルされるとする．この時，対象ドメインにおいては，教師データ $\mathcal{Y}$ が存在しないとする．次に，特徴量抽出部分のパラメータを θ_f として，タスク部分を θ_y，ドメイン分類部分を θ_d とする．各データにおけるタスクに関する損失を $l_y^i(\theta_f, \theta_y)$，ドメインに関する損

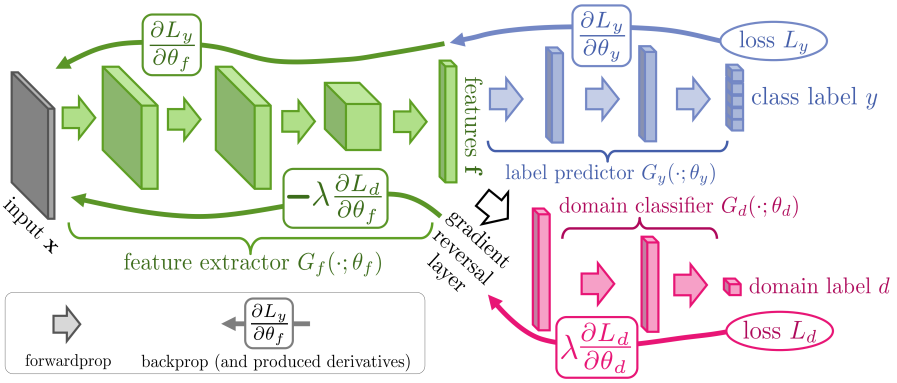


図 2.4: ドメイン敵対的ニューラルネットワークの概要図 (Ganin et al. 2017)

失を $l_d^i(\theta_f, \theta_d)$ とする．損失関数 $L_{\mathrm{DAT}}(\theta_f, \theta_y, \theta_d)$ は

$$L_{\mathrm{DAT}}(\theta_f, \theta_y, \theta_d) = \frac{1}{n}\sum_{i=1}^{n} l_y^i(\theta_f, \theta_y) - \lambda\left(\frac{1}{n}\sum_{i=1}^{n} l_d^i(\theta_f, \theta_d) + \frac{1}{n'}\sum_{i=n+1}^{n+n'} l_d^i(\theta_f, \theta_d)\right) \tag{2.67}$$

となり，式 (2.68) と式 (2.69) で表される問題を求めることとなる．

$$(\hat{\theta}_f, \hat{\theta}_y) = \arg\min_{\theta_f, \theta_y} L_{\mathrm{DAT}}(\theta_f, \theta_y, \hat{\theta}_d), \tag{2.68}$$

$$(\hat{\theta}_d) = \arg\max_{\theta_d} L_{\mathrm{DAT}}(\hat{\theta}_f, \hat{\theta}_y, \theta_d). \tag{2.69}$$

式 (2.67) と式 (2.68) を見ると，タスクと特徴量抽出部分の学習時はドメイン判別に関する損失を引くことで，ドメインの判別ができないような特徴量を学習している．また，式 (2.69) より，ドメイン判別器を学習する際は，特徴量抽出部分のパラメータを固定して，ドメインを判別できるように学習することがわかる．さらに，式 (2.68) と式 (2.69) より，損失関数は min-max 損失となっており，敵対的な学習となっている．

検索においても，ドメイン敵対的学習を用いたドメイン適応が行われており，代表的な方法として MoDIR (Momentum Adversarial Domain Invariant Representations learning) (Xin et al. 2022) がある．Xin et al. (2022) によると，検索

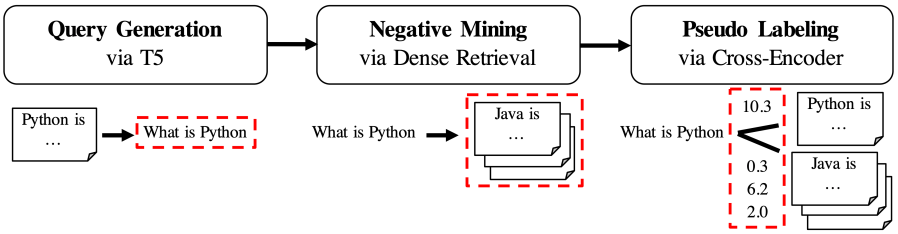


図 2.5: GPL のプロセス (Wang et al. 2022)

においてはマッチングが極めて局所的であるため，分類問題と異なり関連度に対する多様体を形成するような学習がなされていないことが指摘されている．さらに，検索では文書量が多く全文書を負例として用いることができないため，クエリと一部の文書のみをバッチで与えることで学習を行う．そのため，決定境界の学習にソースドメインと対象ドメインの一部のデータのみしか用いることが出来ず，決定境界が不安定になるという問題が発生する．よって，MoDIR ではソースドメイン，対象ドメイン双方のクエリベクトルと文書ベクトルのペアをキューにいれ，1 回の学習ごとにそのキューに入っているペアを用いて，ドメイン判別を行う分類器を学習する．すでに，ベクトルにエンコードされたクエリと文書のペアを用いるため，モーメントキューとXin et al. (2022) は呼んでいる．結果として，BEIR 中の多くのデータで改善が見られた．ただし，その精度向上幅は限定的であった．

2.5.4 疑似クエリを用いたドメイン適応

疑似クエリを用いることで，対象ドメインにおいて教師データが存在しない場合に検索モデルを学習する手法が提案されている (Ma et al. 2021, Wang et al. 2022)．疑似クエリを用いる代表的な手法として，GPL (Generative Pseudo Labeling) (Wang et al. 2022) がある．概要を図 2.5に示す．GPL ではまず，ソースドメインで文書からクエリを生成するモデルとクロスエンコーダによるリランキングモデルを学習する．その後，クエリ生成モデルを用いて対象ドメインで文書からクエリを生成し，そのペアを正例と考える．そして，2.4.3 項で述べたマージン二乗損失 (Hofstätter et al. 2020) を用いて BERT によるリランキングモデルから蒸留を行うことで学習を行う．知識蒸留元としてクロスエンコーダを用いるのは，先行研究 (Thakur et al. 2021) にて，最も対象データでの精度が高いとい

う結果によるものである．

疑似クエリを用いる方法は，分類タスクにおいて，疑似ラベルを用いることに相当すると考えられる．そのため，共変量シフト (Shimodaira 2000) に適応する手法と考えられる．共変量シフトは入力分布がソースドメインから変化することを指す．Kouw and Loog (2018) に沿って共変量シフトを定式化する．分類タスクにおいて，ソースドメイン $\mathcal{S}$ を $(X, Y, p_{\mathcal{S}})$ で表す．一方，対象ドメイン $\mathcal{T}$ を $(X, Y, p_{\mathcal{T}})$ とする．$\mathcal{X}$ は入力の確率変数，$\mathcal{Y}$ は出力の確率変数，$p_{\mathcal{S}}$ はソースドメインの確率分布，$p_{\mathcal{T}}$ は対象ドメイン確率分布である．分類問題であるため，$Y = \{-1, +1\}$ の 2 値分類または $Y = \{1, ..., K\}$ の多値分類である．機械学習などを通じて得られる，分類器を $h \in \mathcal{H}$ と表す．$\mathcal{H}$ は学習によって得られるモデル全体を表す集合である．最後に損失関数を $l(h(x), y)$ とする．すると，ソースドメインにおける汎化誤差 $R^{\mathrm{cls}}_{\mathcal{S}}(h)$ は，

$$R^{\mathrm{cls}}_{\mathcal{S}}(h) = \sum_{y \in Y} \int_X l(h(x), y) p_{\mathcal{S}}(x, y) dx \tag{2.70}$$

$$= \sum_{y \in Y} \int_X l(h(x), y) p_{\mathcal{S}}(y|x) p(x) dx \tag{2.71}$$

と書ける．今，ソースドメインと対象ドメインにおける変化が入力分布のみであるとする．つまり，$p(y, x) = p(y|x)p(x)$ において，$p_{\mathcal{S}}(y|x) = p_{\mathcal{T}}(y|x)$，$p_{\mathcal{S}}(x) \neq p_{\mathcal{T}}(x)$ であるとする．この時，対象ドメインにおける汎化誤差 $R^{\mathrm{cls}}_{\mathcal{T}}(h)$ は，

$$R^{\mathrm{cls}}_{\mathcal{T}}(h) = \sum_{y \in Y} \int_X l(h(x), y) p_{\mathcal{T}}(x, y) dx \tag{2.72}$$

$$= \sum_{y \in Y} \int_X l(h(x), y) p_{\mathcal{T}}(y|x) p_{\mathcal{T}}(x) dx \tag{2.73}$$

$$= \sum_{y \in Y} \int_X l(h(x), y) p_{\mathcal{S}}(y|x) p_{\mathcal{T}}(x) dx \tag{2.74}$$

と書ける．疑似ラベルと用いる手法は，ソースドメインにおいて学習した分類器 $h(x) \approx p_{\mathcal{S}}(y|x)$ を用いて $x \sim p_{\mathcal{T}}(x)$ に対してラベルづけし，それを用いて分類器を学習する．よって，式 (2.74) の状況で学習していると考えられるため，共変量

シフトに対する，ドメイン適応を行う手法と考えられる．

さて，上記と同様の定式化を検索タスクに対して用いる．今，検索モデルを $h(q,d)$ として損失関数を $l(h(q,d),r)$ とする．対象ドメインの予測損失 $R^{\mathrm{ret}}_{\mathcal{T}}(h)$ は式 (2.75) あるいは式 (2.76) のようになる．

$$R^{\mathrm{ret}}_{\mathcal{T}}(h) = \int l(h(q,d),r)p_{\mathcal{T}}(r,q,d)drdqdd \tag{2.75}$$

$$= \int l(h(q,d),r)p_{\mathcal{T}}(r|q,d)p_{\mathcal{T}}(q|d)p_{\mathcal{T}}(d)drdqdd \tag{2.76}$$

分類タスクと異なり，検索タスクでは，学習したモデルがクエリを生成しないため，別途ソースドメインでクエリ生成モデルを学習する．その学習では，ソースドメインにおいて正例となるクエリを生成するように学習する．よって，クエリ生成モデルはソースドメイン同様に $p_{\mathcal{S}}(q|d)$ となるクエリを生成すると期待される．また，GPL では関連度の判定はソースドメインで学習を行ったクロスエンコーダによって判定されているため，関連度を表すモデルもソースドメインと変わらないと考えられる．つまり，$p_{\mathcal{T}}(r|q,d) = p_{\mathcal{S}}(r|q,d)$，$p_{\mathcal{T}}(q|d) = p_{\mathcal{S}}(q|d)$ と仮定して学習していると考えられる．よって，式 (2.76) は，

$$R_{\mathcal{T}}(h) = \int l(h(q,d),r)p_{\mathcal{S}}(r|q,d)p_{\mathcal{S}}(q|d)p_{\mathcal{T}}(d)drdqdd \tag{2.77}$$

となる．以上から，ソースドメインから文書に関するドメインのみシフトした場合と考えられ，この変化は共変量シフトと考えられる．

Wang et al. (2022) は，GPL を MoDIR，継続事前学習，UDALM と比較している．Wang et al. (2022) が実験を行なった BEIR 中の 6 つの対象データで，GPL が最も精度が高いという結果になっている．また，検索タスクにおいては分類タスクと異なり，前述の UDALM よりも継続事前学習が有効であることを示している．Wang et al. (2022) は検索タスクと分類タスクの違いによるものであるとしている．Wang et al. (2022) はさらに，密ベクトル検索モデルの自己教師あり手法である TSDAE (Wang et al. 2021) を行なった後 GPL を適用することで，さらに精度向上が見られることを示している．

なお，対象データの教師データを数件用いる少数事例の設定においても，疑

似クエリを用いる手法が提案されており (Dai et al. 2023)，更なる精度向上をもたらしている．

2.5.5　ゼロショット検索における他の精度向上手法

ゼロショット検索を実現する手法は，教師なしドメイン適応に限られない．本節では，それらについて概観する．

まず，ドメイン変化に頑健な検索モデルを用いる方法がある．2.3 節で述べた SPLADE や ColBERT はドメイン変化に頑健な検索モデルである．さらに，SPLADE のクエリ拡張，文書拡張の性質と ColBERT の文脈化ベクトルの性質を統合するモデルも提案されている (Kong et al. 2023)．本研究では，4 章にて SPLADE や ColBERT に対して AdaLM を適用した結果を述べている．

質問応答検索の他に，要約やテキスト平易化など多様な文対のタスクを用いてマルチタスク学習を行うことで，ゼロショット検索の精度を向上させる試みがある．しかし，単純にマルチタスク学習を行うと個別のタスクに対する精度が低下する場合がある (Fifty et al. 2021)．そこで，タスクやドメイン毎にインストラクションと呼ばれる指示を表現するテキストをモデルに与えながら，マルチタスク学習を行う方法がある (Su et al. 2023, Asai et al. 2023)．これらの手法では，インストラクションの設計において，知見や人手を要する．Cai et al. (2023) はこれを自動化させるため，PrefixTuning (Li and Liang 2021) を用いて学習することで，ゼロショット検索の精度を向上させる手法として HYPER-R を提案している．Tam et al. (2022) では，PrefixTuning 自体は汎化に寄与しないとされていたが，Cai et al. (2023) では，クエリに応じたプロンプトになるようにクエリの埋め込み表現とのアテンションで重み付けを行うと共に，同じタスクはアテンションが類似するように，異なるタスクはアテンションが類似しないように学習することで，ゼロショット検索の精度を向上させている．

これらのマルチタスク学習を用いる手法はドメイン適応としてみた時，コンセプトシフトに対応した手法と考えられる．コンセプトシフトは，対象ドメインにおいて，ソースドメインからラベルの概念が変化していることに相当する．分類問題においては，$p(x, y) = p(y|x)p(x)$ を考えた際，$p_{\mathcal{S}}(y|x) \neq p_{\mathcal{T}}(y|x)$ かつ $p_{\mathcal{S}}(x) = p_{\mathcal{T}}(x)$ となるような状況を指す．検索においては，$p_{\mathcal{S}}(R|Q, D) \neq p_{\mathcal{T}}(R|Q, D)$ あるいは $p_{\mathcal{S}}(R, Q|D) \neq p_{\mathcal{T}}(R, Q|D)$ という状況であると考えられる．つまり，関連

度の基準が変わっているとみなせる．TART (Asai et al. 2023) では，これを意図としてプロンプトで明示している．HYPER-R は教師データとして MS MARCO に加えて Wikipeda を用いた質問応答や関係抽出など複数のタスクで構成された KILT (Petroni et al. 2021) を用いて学習している．KILT での学習は，$p_{\mathcal{S}}(D) = p_{\mathcal{T}}(D)$ であるが，$p_{\mathcal{S}}(R, Q|D) \neq p_{\mathcal{T}}(R, Q|D)$ となっていると考えられる．よって，関連度の基準やクエリが変わっている．これらのことから，プロンプトを用いたマルチタスク学習は主に関連度の基準の変化をプロンプトによって制御する方法と考えられる．本論文の手法は検索対象文書のドメインシフトに関連するものである．よって，コンセプトシフトが共に起きる場合には双方を用いることが有効と考えられる．

言語モデル同様に，検索としての事前学習を行うことでゼロショット検索の精度を向上させる手法も提案されている．一つは疑似正例を作り，対照学習を行う方法である (Xu et al. 2022, Gao and Callan 2022, Izacard et al. 2022, Yu et al. 2022)．各手法は疑似正例を構成する方法や学習方法が異なる．LaPraDoR (Xu et al. 2022) では，疑似正例は文書の一部のスパンをクエリとして使用し残りの部分を正例文書とする Inverse Cloze Training (ICT) (Lee et al. 2019) と，同じテキストに対してエンコードする際にドロップアウト (Srivastava et al. 2014) を用いることで得られる異なるベクトルを正例とする手法 (Gao et al. 2021) を用いている．coCondenser (Gao and Callan 2022)，Contriever (Izacard et al. 2022)，COCO-DR (Yu et al. 2022) では，同じ文書内の二つのテキストスパンを正例としている．coCondenser は併せて，入力の文頭トークン（BERT では CLS トークン）のベクトルをノイズに対して頑健にするために，トランスフォーマー層をスタックし，最終層の CLS トークンのベクトルと低層のトークンのベクトルを入力して，マスク言語モデルで学習する Condenser (Gao and Callan 2021) を併せて学習している．なお，Condenser は後述のオートエンコーダを用いた手法と考えられる．coCondenser は SPLADE でゼロショット検索の精度を向上させることができることが示されている (Formal et al. 2022b)．Contriever は，テキストスパンを用いるのに加えて，スパン中の一部の単語をランダムに削除することで，データ拡張をしている．また，2 つのスパンの重複を許している．なお，Izacard et al. (2022) は ICT よりもテキストスパンを用いたほうが良いことを示している．COCO-DR は，テキストスパンを用いる事前学習を行うと共に，より頑健なモデルを学習する最適

化手法を用いることで，ゼロショット検索の精度を向上させている．近年では，疑似正例としてQAサイトの質問と応答のペアや科学論文で引用関係にあるタイトルペアなどを用いることが提案されている (Wang et al. 2022, Li et al. 2023)．ただし，これらの手法はBEIRを構成するデータセットと重複があるため，ゼロショット検索とは言えない．しかし，検索のみならず分類やクラスタリングなどの多数のタスクで埋め込み表現を評価するベンチマークである，MTEB (Muennighoff et al. 2023) で高い精度を示しており実用上は有用であると考えられる．

他方，オートエンコーダを用いて，検索における事前学習を行う手法も提案されている (Wang et al. 2021, Lu et al. 2021, Xiao et al. 2022, Shen et al. 2023, Liu et al. 2023)．TSDAE (Wang et al. 2021) は，入力の文頭トークンをトランスフォーマーのデコーダにキーとバリューとして入力することでテキストの表現を学習する．Lu et al. (2021) は，より表現力の弱いデコーダを用いる方がテキストの埋め込み表現を改善できることを明らかにした．具体的には，層数の少ないトランスフォーマのデコーダを使用しており，文頭トークン以外はコンテキスト長を制限している．また，エンコーダから表現力の高いベクトルを得るために，エンコーダから文頭トークンのベクトルのみがデコーダの文頭に入力させるように制限している．これによって，デコーダの生成がエンコーダの文頭トークンのベクトルの表現力に向上によって改善されるように制限している．Xiao et al. (2022) は，デコーダの学習もマスク言語モデルで行うことで，ゼロショット検索で更なる精度向上をもたらすことを示している．その際，デコーダにおけるマスク率を通常の15%から50%~70%にすることで，より表現力の高い埋め込み表現を得ることができることを示している．Liu et al. (2023) では，さらにSPLADEのような事前学習済み言語モデルの語彙サイズ次元の疎ベクトルを用いる検索モデルに対する事前学習方法として，SPLADEと同様のエンコード方法を用いて語彙サイズ次元のベクトルが，元々の入力のBag-of-Words表現を復元するような学習をXin et al. (2022) の提案手法と併せて行う方法を提案している．Shen et al. (2023) は疎ベクトル検索モデルに対する，オートエンコーダベースの事前学習手法を提案している．このように密ベクトル検索モデルと疎ベクトル検索モデルの両方に対して事前学習を行うことで，ゼロショット検索の精度を向上させている．検索としての事前学習を行う手法は，エンコーダをより検索に合わせた事前学習を行う手法であるため，4章のように事前学習済み言語モデル自体のドメイン適応と

併せて用いることで，対象ドメインにおいて更なる精度向上をもたらすと考えられる．

2.6 ロバートソン/スパルクジョーンズ重み付け関数による分析手法

事前学習済み言語モデルを用いた検索モデルは，トークン一致のみに依拠したアルゴリズムではない．そのため，トークン一致の度合いを分析するために，ロバートソン・スパルクジョーンズ重み付け（RSJ）関数を用いた分析をFormal et al. (2022a) が提案している．RSJ 関数は 2.1.2 項にて述べたとおり，関連文書から推定される重みであり，トークンの一致のみで考えた場合のランキングに対する理想的な重みと考えられれる．Formal et al. (2022a) は各検索モデルのトークン一致に関する分析を行うため，関連文書の代わりに検索モデルの上位 K 件の文書を関連文書とみなして，RSJ 重みを推定している．そして，RSJ 重みと推定した RSJ 重みの差を用いて分析を行った．

これを定式化すると以下のようになる．まず，式 (2.15) の w_t を $\mathrm{RSJ}_{t,Q}$ と記載する．推定した RSJ 重みを $\widehat{\mathrm{RSJ}}_{t,Q}$ として，上位 K 件の文書を R_Q^K，それ以外の文書を $\bar{R}_Q^K$ とおくと

$$\Delta\widehat{\mathrm{RSJ}}_{t,Q} = \log\left(\frac{p(B_t|R_Q^K)(1-p(B_t|\bar{R}_Q^K))}{(1-p(B_t|R_Q^K))p(B_t|\bar{R}_Q^K)}\right) \tag{2.78}$$

となる．この時，RSJ 重みと推定した RSJ 重みの差 Δw_t は，

$$\Delta\mathrm{RSJ}_{t,Q} = \widehat{\mathrm{RSJ}}_{t,Q} - \mathrm{RSJ}_{t,Q} \tag{2.79}$$

となる．この差を用いて，理想的なトークン一致の度合いと比較を行う．

2.7 評価指標

本論文では，評価データセットに BEIR (Thakur et al. 2021) を用いる．BEIR では評価指標に nDCG@10 (Järvelin and Kekäläinen 2002) と Recall@100 が用いら

れているため，本論文でもその二つを用いる．

Recall@100 は，検索結果上位 100 件の再現率である．クエリ集合 Q としてクエリ q に対する正例文書集合を $|D_q^+|$，検索上位 100 件中の正例文書集合を D_{q100}^+ とすると，

$$\text{Recall@100} = \frac{1}{|Q|} \sum_{q \in Q} \frac{|D_{q100}^+|}{|D_q^+|} \tag{2.80}$$

である．なお，正例文書数自体が，上位 100 件より多い場合がある．そのときは Capped RecallRcap@100 を用いる．これは式 (2.81) で表される．

$$\text{Rcap@100} = \frac{1}{|Q|} \sum_{q \in Q} \frac{|D_{q100}^+|}{min(|D_q^+|, 100)} \tag{2.81}$$

nDCG (normalized Discounted Cumulative Gain) は適合度が 1/0 のような 2 値ではなく多値の場合にも使用できる検索精度の指標である．名前の通り，DCG (Discounted Cumulative Gain) を正規化したものである．今，多値の適合度を利得と考え，検索順位 r の文書の利得を $g(r)$ とする．利得は順位によって減損すると考え，減損利得を $\mathrm{dg}(r) = g(r)/\log_2(r+1)$ とする．この時，DCG は式 (2.82) のようになる．

$$\text{DCG} = \sum_{r=1}^{\mathrm{md}} \mathrm{dg}(r) \tag{2.82}$$

ここで，md は最大順位である．

nDCG は DCG を理想的な DCG (IDCG) で割ったものである．IDCG は理想的なリストを用いて計算される DCG であり，理想的なリストは利得が大きい文書から順位並べた場合の利得である．この理想的なリストにおける順位 r の減損利得を $\mathrm{dg}^*(r)$ とする．この時，nDCG は式 (2.83) のようになる．

$$\text{nDCG} = \frac{\text{DCG}}{\text{IDCG}} = \frac{\sum_{r=1}^{\mathrm{md}} \mathrm{dg}(r)}{\sum_{r=1}^{\mathrm{md}} \mathrm{dg}^*(r)} \tag{2.83}$$

nDCG@10 では，$\mathrm{md} = 10$ となる．つまり，検索結果と理想リストの上位 10 件を

用いて計算される．nDCG において，IDCG の計算に用いる理想リストはすべての適合文書の全てを網羅していない場合がある．そのため，nDCG は適合率の拡張と考えられる．

第 3 章

対象データにおけるトークンの重要度を用いるドメイン適応

密ベクトル検索において対象データで精度が低下する原因の一つに，クエリ中にあるキーワードが完全一致する文書を上位にできないという点が指摘されている (Formal et al. 2022a, Ram et al. 2023)．本章では，対象データにおけるトークンの重要度を用いることでこの課題の解決を行う．その手法として，本研究の提案手法である Contextualized BM25 (C-BM25) について述べる．

対象データにおけるトークンの重要度を用いる手法としては，2.5.1 項で述べた，BM25 併用がある．BM25 はスコア中でトークンの逆文書頻度 (IDF) をスコアリングとして考慮している．また，クエリ中のキーワードは特定の文書のみに出現しやすい傾向がある．そのため，対象データにおけるトークンの重要度を用いることは，クエリ中のキーワードと完全一致する文書が上位にランキングされないという問題 (Formal et al. 2022a) に対して有効な方法である．

しかしながら，BM25 併用にはいくつかの課題がある．BM25 併用は検索における関連度の要素である，文書の一部がクエリと適合しているという要素 (Guo et al. 2016) が十分に関連度スコアに反映されていないと考えられる．なぜならば，BM25 の関連度はトークンの一致のみを考慮するため，コンテキストを考慮していない．よって，BM25 では文脈についてクエリが文書の一部のみと適合しているかどうかを反映できていない．また，密ベクトル検索は文書全体を一つのベクトルで表すため，関連度スコアは全体の類似性に相当すると考えられる．そのため，キーワードを重視しながら，文書中のキーワード周辺の文脈を関連度スコアに反映で

きれば，更なる精度改善が見込める．

そこで，このような課題を解決するスコアリング方法として，C-BM25 を提案する．C-BM25 は，クエリと文書で一致したトークンの文脈類似度の計算に密ベクトル検索モデルでエンコードされるトークンのベクトルを用いる．そして，同じ語を持つトークン中で文脈類似度の最大値を取得し，BM25 で重みづけ，その和で関連度スコアを計算することで，キーワードを重視しながら文脈の関連度をスコアに反映することを可能にする．

本研究では，C-BM25 の有効性をゼロショット検索のベンチマークデータセットである BEIR を用いて検証した．結果，C-BM25 は密ベクトル検索や BM25 より高い精度となった．また，C-BM25 の関連度スコアと密ベクトル検索の関連度スコアの和を関連度スコアとする C-BM25 併用は BM25 併用より nDCG@10 で精度の向上が見られた．さらに，ドメイン変化に頑健なモデルである SPLADE を上回る精度となった．特に，教師データから語彙が単語頻度が異なるドメインで C-BM25 の方が精度が高いことを確認した．加えて，C-BM25 が検索として実用レベルの遅延速度となることを示した．

応用上，大規模な教師データを集めることが困難な場合がある．実際，研究でよく用いられる MS MARCO (Nguyen et al. 2016) は，その用途が研究に限定されている．そこで，C-BM25 に対して教師なし文表現である SimCSE (Gao et al. 2021) を用いて実験をおこなった．その結果，SimCSE を用いる場合でも，BM25 に対して精度が向上することを確認した．

本章では，3.1 節にて，C-BM25 の定式化を行う．3.2 節にて，有効性を検証する実験設定を述べ，その結果を 3.3 節にてを述べる．各要素の分析を 3.4 節にて述べ，事例を用いた考察を 3.5 節にて述べる．

3.1 Contextualized BM25 (C-BM25)

C-BM25 は，キーワードを重視しながら文脈の類似度を関連度スコアに反映することを可能にするスコアリング方法である．これを実現するために，クエリと文書で一致したトークンの文脈類似度を計算し，BM25 で重み付け，トークン毎にその和をとる．文脈類似度の計算には，密ベクトル検索モデルとして訓練されたエンコーダを用いる．概要を図 3.1に示す．

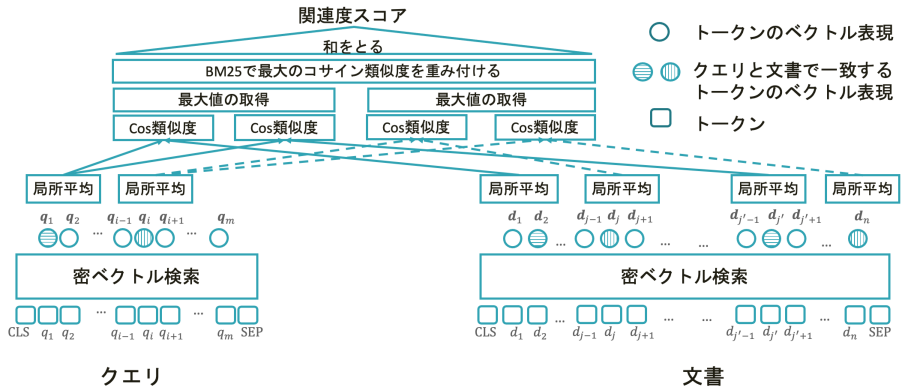


図 3.1: C-BM25 の模式図

このスコアリング方法は，2.3.4 項にて述べた COIL を BM25 で重みづけたものとみなすことができる．異なる点は，エンコーダに密ベクトル検索モデル用いており COIL のスコアリングで学習していないこと及び文脈を表現するベクトルにクエリと文書で一致したトークンを基準に左右の窓幅内にあるトークンを表現するベクトルの平均を用いる点である．

今，2 章と同様に，クエリがトークン列 $\boldsymbol{q} = (t_1, t_2, ..., t_{l_q}) \in \mathcal{V}^{l_q}$ で表され，文書が $\boldsymbol{d} = (t'_1, t'_2, ..., t'_{l_d}) \in \mathcal{V}^{l_d}$ で表されるとする．BM25 の関連度スコア $S_{\mathrm{BM25}}(\boldsymbol{q}, \boldsymbol{d})$ は，式 (2.35) とは表記を変え，式 (3.1) で表されるとする．

$$
\begin{aligned}
S_{\mathrm{BM25}}(\boldsymbol{q}, \boldsymbol{d}) &= \sum_{i=1}^{l_q} \mathrm{BM25}(t_i, \boldsymbol{d}) \\
\mathrm{BM25}(t, \boldsymbol{d}) &= \frac{f(t, \boldsymbol{d})(k_1 + 1)}{f(t, \boldsymbol{d}) + (k_1((1 - b) + b\frac{l_d}{l_{\mathrm{avg}}}))} \mathrm{IDF}(t)
\end{aligned}
\tag{3.1}
$$

$\mathrm{BM25}(t, \boldsymbol{d})$ は，トークン t の文書 $\boldsymbol{d}$ に対する BM25 のスコアである．また，$f(t, \boldsymbol{d})$ は文書 d におけるトークンの頻度である．さて，C-BM25 はクエリと文書

で一致するトークン毎に文脈類似度を計算する．今，クエリのトークンを表現するベクトルを $\boldsymbol{u}_i$ とし，文書のトークンを表現するベクトルを $\boldsymbol{u}'_j$ とすると，C-BM25 は式 (3.2) で表される．

$$S_{\text{C-BM25}}(\boldsymbol{q}, \boldsymbol{d}) = \sum_{i=1}^{l_q} \text{BM25}(t_i, \boldsymbol{d}) \max_{1 \leq j \leq l_d} \boldsymbol{u}_i^T \boldsymbol{u}'_j \tag{3.2}$$

また，$\boldsymbol{u}_i$ と $\boldsymbol{u}'_i$ は式 (3.3)，式 (3.4) の通りである．

$$\boldsymbol{u}_i = \frac{1}{2o+1} \sum_{k=i-o}^{i+o} \boldsymbol{h}_k \tag{3.3}$$

$$\boldsymbol{u}'_j = \frac{1}{2o+1} \sum_{k=i-o}^{i+o} \boldsymbol{h}'_k \tag{3.4}$$

なお，$\boldsymbol{h}_k$ はトークン列 $\boldsymbol{q}$ をエンコーダを用いてエンコードしたベクトル列である $(\boldsymbol{h}_1, ..., \boldsymbol{h}_k, ..., \boldsymbol{h}_{|l_q|}) \in \mathbb{R}^{d \times |l_q|}$ の k 番目ベクトルである．また，$\boldsymbol{h}'_k$ はトークン列 $\boldsymbol{d}$ をエンコーダを用いてエンコードしたベクトル列 $(\boldsymbol{h}'_1, ..., \boldsymbol{h}'_k, ..., \boldsymbol{h}'_{|l_d|}) \in \mathbb{R}^{d \times |l_d|}$ の k 番目ベクトルである．o は窓幅である．

式 (2.57) より，COIL ではトークン列をエンコードしてベクトル列をそのまま用いているが，C-BM25 はクエリと文書で一致したトークンを基準として，左右の窓幅内にあるトークンを表現するベクトルを平均したベクトルを用いている．これは，トークンではなくより文脈を表現するベクトルにするためである．また，密ベクトル検索を平均プーリングを用いて学習するため，使用時もそれに合わせるという点も考慮している．

また，COIL と同様に C-BM25 と密ベクトル検索のスコアを足し合わせる場合も考える．これを 2.5.1 項で述べた BM25 併用との対比で，C-BM25 併用（HC-BM25: Hybrid-C-BM25）と呼ぶことにする．この場合のスコアを $S_{\text{HC-BM25}}$ とすると式 (3.5) のようになる．

$$S_{\text{HC-BM25}}(\boldsymbol{q}, \boldsymbol{d}) = S_{\text{C-BM25}}(\boldsymbol{q}, \boldsymbol{d}) + S_{\text{Dense}}(\boldsymbol{q}, \boldsymbol{d}) \tag{3.5}$$

3.1.1 ドメイン適応としての C-BM25

C-BM25 は COIL-tok を対象データの重みによってドメイン適応したものとみなすことができる．実際，式 (2.57) と式 (3.2) の大きな違いは，BM25 で重み付けていることにある．

検索問題におけるソースドメインと対象ドメインの汎化損失はそれぞれ式 (3.6), 式 (3.7) の様に書ける．

$$R^{\mathrm{ret}}_{\mathcal{S}}(h) = \int l(h(q,d),r)p_{\mathcal{S}}(r,q,d)drdqdd \tag{3.6}$$

$$R^{\mathrm{ret}}_{\mathcal{T}}(h) = \int l(h(q,d),r)p_{\mathcal{T}}(r,q,d)drdqdd \tag{3.7}$$

スコアリングを行う関数は $h(q,d)$ であるので，ソースドメインで訓練した検索モデルのスコアを $S_{\mathcal{S}}(q,d)$ とすると，

$$h_{\mathcal{S}}(q,d) = S_{\mathcal{S}}(q,d) \tag{3.8}$$

と考えられる．いま，COIL を考えているので，$S_{\mathcal{S}}(q,d)$ について

$$S_{\mathcal{S}}(q,d) = \sum_{t \in \boldsymbol{q} \cap \boldsymbol{d}} S'_{\mathcal{S}}(t,q,d) \tag{3.9}$$

と書ける．さらに，文書集合から決まる部分重みがありそれが $w_{D_S}(t,\boldsymbol{d})$ であるとする．ソースドメインでは $w_{D_S}(t,\boldsymbol{d}) = 1$ とすると，

$$S_{\mathcal{S}}(q,d) = \sum_{t \in \boldsymbol{q} \cap \boldsymbol{d}} w_{D_S}(t,\boldsymbol{d}) S'_{\mathcal{S}}(t,q,d) \tag{3.10}$$

と書ける．よって，対象ドメインについて，C-BM25 は式 (3.11) のように考えているとみなせる．

$$h_{\mathcal{T}}(q,d) \sim S_{\mathcal{T}}(q,d) = \sum_{t \in \boldsymbol{q} \cap \boldsymbol{d}} w_{D_T}(t,\boldsymbol{d}) S'_{\mathcal{S}}(t,q,d) \tag{3.11}$$

$$w_{D_T}(t,\boldsymbol{d}) = \mathrm{BM25}(t,\boldsymbol{d}) \tag{3.12}$$

さて，ソースドメインで学習されたモデルを用いた場合の，対象ドメインでの汎化損失は式 (3.14) の様に書ける．

$$R^{\mathrm{ret}}_{\mathcal{T}}(h) = \int l(h_{\mathcal{S}}(q,d),r)p_{\mathcal{T}}(r,q,d)drdqdd \tag{3.13}$$

$$= \int l(h_{\mathcal{S}}(q,d),r)p_{\mathcal{S}}(r,q,d)\frac{p_{\mathcal{T}}(r,q,d)}{p_{\mathcal{S}}(r,q,d)}drdqdd \tag{3.14}$$

これが，C-BM25 の重み付けを行うことで，式 (3.15) の様になる．

$$R^{\mathrm{ret}}_{\mathcal{T}}(h) = \int l(h_{\mathcal{T}}(q,d),r)p_{\mathcal{S}}(r,q,d)\frac{p_{\mathcal{T}}(r,q,d)}{p_{\mathcal{S}}(r,q,d)}drdqdd \tag{3.15}$$

よって，BM25 による重み付けでスコアリングを変えることは，密度比に対する補正と考えられる．

なお，上記の密度比は関連度，クエリ，文書全てのドメインシフトを含んでいるが，本手法は検索対象文書のドメインシフトに対して他の検索モデルより精度が高いことを 3.3.1 項にて実験的に示す．

3.2 実験設定

本節では，C-BM25 の有効性を示す実験に用いるデータセットとエンコーダの学習設定について述べる．

3.2.1 データセット

本実験では，検索モデルを対象データにおいて教師データを用いないで精度検証を行うベンチマークデータセットとして BEIR (Thakur et al. 2021) を用いる．BEIR は既存の 9 種類 18 データセットからなり，広く用いられている．精度指標は，nDCG@10 と Recall@100 が用いられる．今回は，Thakur et al. (2021) が直接配布しているデータセット[1] と Robust04 (Voorhees 2004) を合わせた 8 種類 14

[1]具体的には，ArguAna (Wachsmuth et al. 2018)，Climate-FEVER (C-FEVER) (Leippold and Diggelmann 2020)，DBPedia-Entity Retrieval (DBPedia) (Hasibi et al. 2017)，FEVER (Thorne et al. 2018)，FiQA (Maia et al. 2018)，HotPotQA (Yang et al. 2018)，NFCorpus (Boteva et al. 2016)，Natural Question (NQ) (Kwiatkowski et al. 2019)，

データセットを用いる．

C-BM25 の検証時は計算資源の都合からリランキングを用いる．リランキングは，予め BM25 などの高速で軽量な検索アルゴリズムを用いて上位 K 件を検索し，その K 件を並び替える方法である．Thakur et al. (2021) 等に倣い，BM25 で検索を行い $K = 100$ とした．また，リランキングで評価を行うため，精度評価は nDCG@10 を用いる．

検索モデルを学習するデータセットとしては，MS MARCO を用いる．BEIR においても MS MARCO をソースドメインの学習データとしている．

また，C-BM25 の検索における実効性を確認するため，遅延速度を計測する．遅延速度の計測には Robust04 を用いた．Robust04 は ad-hoc 検索のベンチマークとして広く用いられている．また，Robust04 は BEIR を構成するデータセットの一つである．クエリには title という短いクエリと description という長いクエリがあるが，title をクエリとして使用した．

3.2.2 ベースライン手法

本章の実験では，ベースライン手法として，2 章にて述べた BM25，密ベクトル検索，COIL，クロスエンコーダ，SPLADE，ColBERT を用いた．BM25，密ベクトル検索は C-BM25 で要素として用いられるモデルである．また，COIL は C-BM25 と同種のモデルである．また，クロスエンコーダ，SPLADE，ColBERT はドメイン外でも BM25 を精度で上回るモデルである．BEIR における評価方法について，クロスエンコーダ，COIL，ColBERT はリランキングで評価を行い，密ベクトル検索と SPLADE は検索で評価を行った．

また，C-BM25 と同様に対象データの重みを用いる方法として，密ベクトル検索に対する重み平均プーリングと 2.5.1 項で述べた BM25 併用も併せてベースラインとした．重み平均プーリングの重みは，BM25 を用いている．

表 3.1: 密ベクトル検索のエンコーダ訓練時の学習パラメータ

	密ベクトル検索	COIL-tok
バッチサイズ	20	16
最大クエリ長さ	-	64
最大文書長さ	300	300
学習率	2e-5	5e-6
エポック数	5	5
Warmup ステップ	1000	-
Warmup 率	-	0.1

3.2.3 学習設定と実装

本項では各モデルの実装について述べる．クロスエンコーダは (Thakur et al. 2021) 等の結果を用いた．これは，cross-encoder/ms-marco-MiniLM-L-6-v2 [2] を用いたものである．SPLADE(Formal et al. 2021) と ColBERT(Santhanam et al. 2022) についてはそれぞれFormal et al. (2021) とSanthanam et al. (2022) が配布しているモデル[3] を用いた．密ベクトル検索，COIL は改めて学習したモデルを用いた．密ベクトル検索は学習に SentenceTransformers[4] の実装を用いた．また，COIL[5] にはGao et al. (2021b) の実装を用いた．それぞれ，ハイパーパラメータの詳細を表 3.1に記す．

学習時の損失関数であるが，密ベクトル検索，ColBERT，SPLADE は蒸留を用いた．蒸留元のスコアリングにはクロスエンコーダを用いた．それぞれ著者に従い，ColBERT は KL ダイバージェンスによる蒸留を用い，密ベクトル検索と SPLADE はマージン二乗損失を用いた．一方，C-BM25 に用いるエンコーダは交差エントロピー損失を用いた．これは，3.4.3 項にて述べる通り C-BM25 においては，蒸留を用いない方が高い精度となったためである．COIL についても，Gao et al. (2021b) に従い交差エントロピー損失を用いた．

負例については，SPLADE と密ベクトル検索の学習には Sentence Transform-

SCIDOCS (Cohan et al. 2020)，SciFact (Wadden et al. 2020)，Quora，TREC-COVID (T-COVID) (Voorhees et al. 2021)， Touché-2020 (Bondarenko et al. 2020) である．

[2]https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

[3]SPLADE: https://huggingface.co/naver/splade_v2_distil，ColBERT: https://downloads.cs.stanford.edu/nlp/data/colbert/colbertv2/colbertv2.0.tar.gz

[4]https://github.com/UKPLab/sentence-transformers

[5]https://github.com/luyug/COIL

ers で配布されているもの[6] を用いた．これには，BM25 と他の密ベクトル検索モデルが含まれている．ColBERT には，自己負例を用いている．具体的には，著者に従い交差エントロピー損失と BM25 負例を用いて訓練した ColBERT により負例をサンプリングした．COIL の学習には BM25 負例を用いた．C-BM25 のエンコーダの学習には，BM25 負例を用いた．

事前学習済み言語モデルについては，密ベクトル検索と COIL に対しては MPNet (Song et al. 2020) を用いた[7]．SentenceTransformers では，事前学習済みの埋め込み表現として配布しており[8]，MPNet を事前学習済み言語モデルを用いた場合が最も精度が高いため，MPNet を用いた．

要素として使用するモデルとの差分を確認するため，C-BM25 に対して，BM25，密ベクトル検索と検定を行った．また，C-BM25 併用に対しては，既存手法である BM25 併用及び今回の検証データで最も精度が高かった既存手法である SPLADE と検定を行った．検定は，対応のある t 検定 (酒井 2015) を行い，多重検定の補正は Benjamini-Hochberg 法 (瀬々・浜田 2015) で行った．

BM25 併用と C-BM25 併用の和の取り方であるが，共に上位 100 件の文書のスコアで和を取った．片方にのみ出現する文書については，他方の最低スコアとの和を取った．

BM25 の実装は pyserini(Lin et al. 2021) を使用している．BM25 のパラメータは，(Thakur et al. 2021) 等と同じく，$k_1 = 0.9$，$b = 0.4$ とした．C-BM25 では，$k_1 = 0.82$，$b = 0.65$ とした．また，C-BM25 でベクトルの平均をとる窓幅 $o = 3$ とした．

遅延速度の計測実施時には，(Gao et al. 2021b) 等と同様に，転置インデックスを構築している．また密ベクトル検索や C-BM25 は pytorch (Paszke et al. 2019) を使用して実装おり，遅延速度の計測には jupyter notebook の timeit コマンドを用いて実行した．サーバースペックは，Intel Xeon Gold 6132 Processor2.6GHz の 8 コアを使用し，RAM は 600GB である．

[6] https://huggingface.co/datasets/sentence-transformers/msmarco-hard-negatives/resolve/main/msmarco-hard-negatives.jsonl.gz

[7] ColBERT，SPLADE についても，MPNet を事前学習済みモデルとして訓練を実施したが，BERT の性能に及ばなかった．

[8] https://www.sbert.net/docs/pretrained_models.html

表 3.2: BEIR における評価結果．最も良かった結果を**太字**にしている．C-BM25 について，BM25 から統計的に有意な結果を B，密ベクトル検索から統計的に有意な結果を D の添字をつけている．また，C-BM25 併用について，BM25 併用から統計的に有意な結果を H，SPLADE から統計的に有意な結果を S の添字をつけている．有意水準は 5%としている

	ベースライン手法			対象データの重みを用いたドメイン適応手法		ドメイン外で有効な検索モデル			提案手法	
	BM25	密ベクトル検索	COIL-tok	重み平均	BM25 併用	クロスエンコーダ	ColBERT	SPLADE	C-BM25	HC-BM25
ドメイン内										
MS MARCO	0.506	0.708	0.663	0.673	0.740	0.727	**0.748**	0.729	0.669	0.705
対象データ										
Arguana	0.275	0.487	0.224	**0.490**	0.416	0.302	0.463	0.479	0.449^{B}	0.457^{H}
C-FEVER	0.158	0.206	0.170	**0.264**	0.240	0.243	0.176	0.235	$0.244^{B,D}$	$0.261^{H,S}$
DBPedia	0.285	0.355	0.322	0.365	0.398	0.434	**0.446**	0.435	0.363^{B}	0.396
FEVER	0.577	0.706	0.742	0.622	0.768	0.785	0.785	**0.786**	$0.754^{B,D}$	0.775^{H}
FiQA	0.236	0.315	0.255	0.307	0.351	0.347	**0.356**	0.336	0.322^{B}	0.352
HotpotQA	0.567	0.563	0.605	0.537	0.670	**0.706**	0.667	0.684	$0.663^{B,D}$	0.669
NFCorpus	0.330	0.306	0.304	0.307	0.328	**0.366**	0.338	0.334	$0.343^{B,D}$	0.331
NQ	0.243	0.491	0.353	0.462	0.498	0.459	**0.562**	0.521	0.409^{B}	0.467
Quora	0.789	0.836	0.743	0.863	**0.877**	0.825	0.852	0.838	$0.844^{B,D}$	0.868^{S}
SCIDOCS	0.140	0.140	0.133	0.148	0.164	0.162	0.154	0.158	$0.159^{B,D}$	$\mathbf{0.165}^{S}$
SciFact	0.664	0.576	0.672	0.592	0.712	0.681	0.693	0.693	$0.711^{B,D}$	$\mathbf{0.720}^{S}$
T-COVID	0.530	0.562	0.696	0.473	0.621	0.736	0.738	0.710	$0.724^{B,D}$	$\mathbf{0.762}^{H}$
Robust04	0.409	0.426	0.408	0.434	**0.507**	0.476	0.441	0.469	$0.466^{B,D}$	0.498^{S}
Touchë2020	**0.454**	0.189	0.242	0.192	0.313	0.321	0.263	0.272	0.361	0.353^{S}
対象データ平均	0.404	0.440	0.419	0.433	0.490	0.489	0.495	0.496	0.486	**0.505**

3.3 結果

本節ではまず，BEIR における精度評価について述べる．その後，Robust04 を用いた速度計測を行う．

3.3.1 BEIR における精度評価

評価結果を表 3.2に示す．まず，C-BM25 と BM25，密ベクトル検索を比べる．C-BM25 が BM25 に対して，Touchë2020 を除くすべてのデータに対して，統計的に有意に上回っている．密ベクトル検索に対しては，対象データにおいて Arguana, DBPedia, FiQA, Natural Question を除くデータに対して，統計的に有意に上回っている．また，DBPedia と FiQA では統計的に有意ではないが C-BM25 が密ベクトル検索を上回っている．よって，C-BM25 がドメイン適応手法として効果的であることがわかる．一方で，ソースドメインである MS MARCO においては，密ベクトル検索の方が高い精度を示している．ドメイン内では，本手法は有効ではないと考えられる．なお，先行研究では密ベクトル検索が BM25 を精度で平均的に下回っていた．マージン二乗誤差と負例を工夫したことによって精度が向上したと考えられる．

対象データの重みを用いる既存のドメイン適応手法と提案手法の比較をする．C-BM25 は対象データにおいて重み平均ベクトルを平均的に上回っている．なお，重み平均ベクトルは，密ベクトル検索を平均的に下回っている．密ベクトル検索のスコアは平均プーリングをした場合のものなので，重み平均は有効なドメイン適応方法ではなかったと言える．BM25 併用と比較すると，C-BM25 は BM25 併用より平均的には低い精度となっているが，C-BM25 併用は高い精度となっている．特に，Arguana，Climate-FEVER，FEVER，TREC-COVID では，統計的に有意に上回っている．Arguana は，C-BM25 に対して密ベクトル検索が上回っている．Arguana は同じ議論に対して反対の立場にある主張を関連とする検索タスクであり，Wachsmuth et al. (2018) は単語ベクトルにおいて類似しているが表層の類似度は異なるものを使用してデータの拡張を行っている．よって，そのデータの作り方からも，より文脈が重要なタスクであると考えられる．そのため，文脈を考慮する C-BM25 併用の方が BM25 併用に対して良い結果となったと考えれる．しかし，ソースドメインである MS MARCO においては，BM25 併用の方が有効である．ソースドメインにおいて，BM25 のようなトークンの一致で検索するアルゴリズムと密ベクトル検索の相補性は，Gao et al. (2021a) やLuan et al. (2021) によって指摘されている．この結果は，それを支持するものである．

ドメイン外で有効な既存の検索モデルと比較する．C-BM25 併用が平均的に最も良い精度となっている．しかしながら，各データセットで最も精度が良いモ

表 3.3: Robust04 における検索タスクの結果．最も良い結果は**太字**にしている

	遅延時間/ms	nDCG@10
BM25	**9.13**	0.448
密ベクトル検索	90.9	0.410
C-BM25	61.9	**0.479**

デルはそれぞれ異なっている．各モデルで転移できる側面が異なり，対象データの性質によって性能にばらつきが出ていると推察される．既存のモデルのうち平均的に最も精度が良い SPLADE と C-BM25 併用を比較する．C-BM25 併用は，Climate-FEVER，Quora，SCIDOCS，Scifact，Robust04，Touchë2020 で統計的に有意に上回っている．また，TREC-COVID と FiQA も 5%有意水準で有意ではないものの，基準に近い p 値であった（TREC-COVID が $p = 0.063$，FiQA が $p = 0.058$）．特に SCIDOCS，Scifact，TREC-COVID といった語彙や単語の頻度がソースドメインで大きくことなるデータセットで特に良い結果となっている．一方で，FEVER，DBpedia，Natural Question といったデータセットにおいては，SPLADE が上回っている．これ等は Wikipedia を用いたデータセットになっている．MS MARCO は一般的な Web 検索で作られたデータセットである．また，Santhanam et al. (2022) でも，MS MARCO は映画やスポーツ選手などの Wikipeida に存在するエンティティーの知識を問うクエリが多いという指摘がある．さらに，ColBERT は DBpedia と Natural Question でさらに高い精度となっている．よって，ドメイン外で有効な検索モデルは，より文書集合がソースドメインと類似した対象データにおいて高精度であると考えられる．

3.3.2 速度計測

Robust04 における速度計測の結果を表 3.3に示す．参考値として，検索による精度も示す．表より，最速は BM25 であるものの，C-BM25 がそれに次ぐ速度になっている．Nielsen (1993) によると，ユーザが自由な操作感を得られる遅延速度は 100ms/query から 1s/query とされており，C-BM25 はそれを上回る速度を達成出来ている．C-BM25 の計算は COIL と同様の形式であり，密ベクトル検索よりも高速であることが示されている (Gao et al. 2021b)．COIL はクエリと文書で一致した語彙を持つトークンを表現するベクトルとの内積計算になるた

表 3.4: C-BM25 における重み付け方法による精度比較．最も良い結果を**太字**にしている

	重みなし	IDF のみ	BM25
ソースドメイン			
MS MARCO	0.659	0.657	**0.669**
対象データ			
Arguana	0.341	0.405	**0.449**
C-FEVER	0.229	0.218	**0.244**
DBPedia	0.330	0.339	**0.363**
FEVER	0.784	**0.787**	0.754
FiQA	0.261	0.295	**0.322**
HotpotQA	0.596	0.649	**0.663**
NFCorpus	0.328	0.330	**0.343**
NQ	0.347	0.364	**0.409**
Quora	0.794	0.833	**0.844**
SCIDOCS	0.142	0.149	**0.159**
SciFact	0.674	0.691	**0.711**
T-COVID	0.677	0.656	**0.724**
Robust04	0.438	0.449	**0.466**
Touche2020	0.304	0.326	**0.361**
対象データ平均	0.446	0.464	**0.486**

め，内積計算するベクトルが文書全件より少なくなると考えられる．C-BM25 は，Gao et al. (2021b) の実験設定よりもベクトルの次元が大きいが，密ベクトル検索よりも高速に検索できている．また，リランキングではなく全件を検索する場合でも，C-BM25 は精度において BM25 や密ベクトル検索を上回っている．

3.4 各要素の影響分析

本節では，各要素の影響を分析する．

3.4.1 重みづけ方法の比較

密ベクトル検索は，クエリ中のキーワードが特に教師データ中で低頻度な場合に，そのキーワードと完全一致する文書が上位にランキングづけできない．低頻度なトークンへの重み付けは，BM25 の中で特に IDF による重みによってなされる．

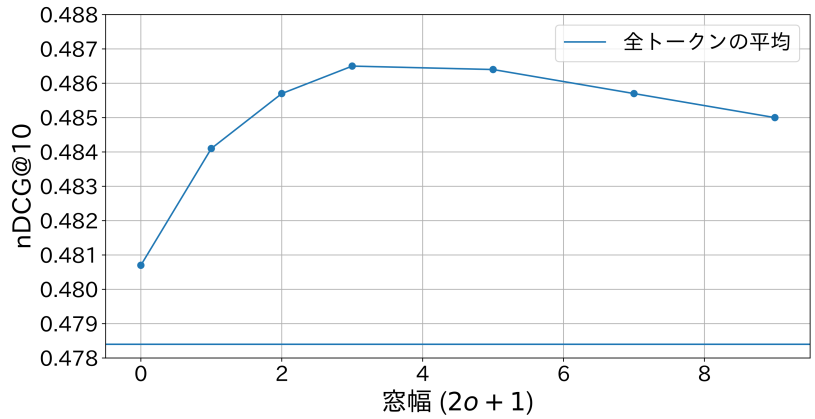


図 3.2: 窓幅の変化による C-BM25 の nDCG@10 の変化

そのため，C-BM25 において重みづけなし，IDF のみで重み付ける場合についても実験を行った．結果を表 3.4に示す．

重みづけを行わない場合よりも IDF で重み付ける場合の方が，対象データにおいて平均的には精度が高くなった．また，IDF のみで重み付ける場合よりも，対象データにおいて BM25 で重み付ける場合の方が，平均的には精度が高くなった．式 (2.35) より BM25 は IDF と TF の部分から成る．TF も対象データにおける精度向上に寄与していることわかる．一方，ソースドメインでは，重みづけを行わない場合からの精度向上は対象データの場合と比べると小さい．ソースドメインでは，BM25 による重みがない場合も十分判別できていると考えられる．

3.4.2 窓幅の影響

C-BM25 では，トークンを表現するベクトルとして，トークン周辺のベクトルを平均したベクトルを用いている．その窓幅を変化させた場合の結果を図 3.2に示す．窓幅 $o = 3$ で最大値となり，その後精度が低くなることがわかる．つまり，文脈を表すベクトルとして，トークンのベクトル $(o = 0)$ や全トークンの平均をとるよりも良い結果となっている．全トークンの平均よりも良い精度となっているため，クエリの文脈が文書の一部の文脈とマッチングすることを考慮するための工夫として，一致したトークン周辺の文脈表現するベクトルを用いることが精度向上に寄与したと考えられる．また，トークンのベクトルより精度が向上して

表 3.5: 密ベクトル検索の学習方法における C-BM25 の精度の差．最も良い結果を**太字**にしている

データセット \ エンコーダ	COIL-tok	COIL-tok（768 次元）	密ベクトル検索	密ベクトル検索（蒸留＋複数負例）
ソースドメイン				
MS MARCO	0.668	**0.674**	0.669	0.605
対象データ				
Arguana	0.445	0.434	0.449	**0.465**
C-FEVER	0.201	0.208	**0.244**	0.230
DBPedia	0.310	0.351	0.363	**0.377**
FEVER	0.714	0.734	**0.754**	0.728
FiQA	0.284	0.311	**0.322**	0.305
HotpotQA	0.607	0.642	**0.663**	0.657
NFCorpus	0.314	0.328	**0.343**	0.340
NQ	0.381	0.402	**0.409**	0.382
Quora	0.760	0.780	**0.844**	0.836
SCIDOCS	0.152	0.154	0.159	**0.162**
SciFact	0.679	0.688	**0.711**	0.705
T-COVID	0.758	0.748	**0.724**	0.688
Robust04	0.443	0.460	**0.466**	0.458
Touche2020	0.299	0.313	0.361	**0.387**
対象データ平均	0.453	0.468	**0.486**	0.480

いるため，より文脈を考慮するために窓幅内の平均をとる操作が精度向上に寄与したと考えられる．

3.4.3 エンコーダの学習方法の比較

C-BM25 では，密ベクトル検索とは異なり負の対数尤度と BM25 の負例を用いてエンコーダを学習している．本項では，このエンコーダが蒸留を用いて学習した密ベクトル検索モデルや COIL よりも精度が良いことを示す．これらの場合についての比較を表 3.5に示す．COIL については，Gao et al. (2021b) では 32 次元の場合であるが，密ベクトル検索と同様に 768 次元の場合も実施した．

まず，COIL をエンコーダとして用いた場合と比較する．ソースドメインでは，エンコーダを COIL-tok にする場合も密ベクトル検索にする場合もほとんど変わらない精度となった．一方，対象データにおいては，平均的には C-BM25 の方が良い結果となった．これは，COIL において次元数を C-BM25 と揃えて 768 次元とした場合も同様である．この結果は，文脈類似度の方がトークンの重要

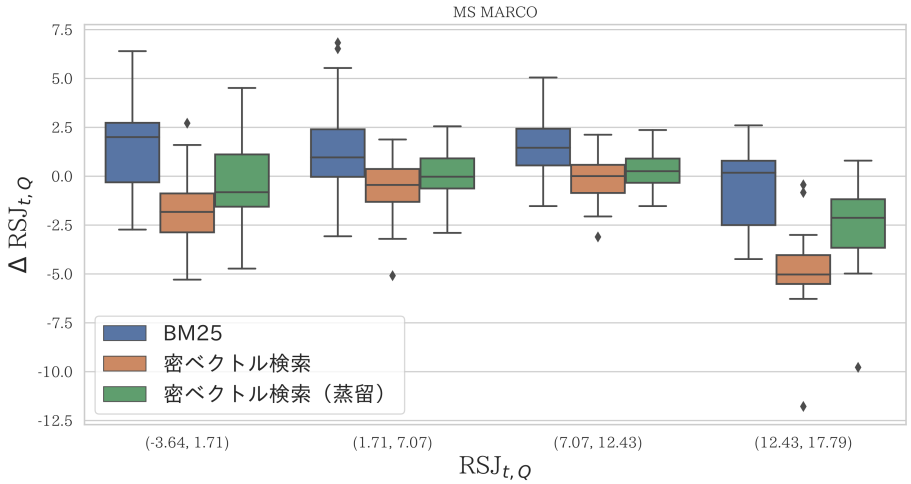


図 3.3: MS MARCO における密ベクトル検索モデルの $\Delta\text{RSJ}_{t,Q}$ の分布．参照のために，BM25 の検索結果についても $\Delta\text{RSJ}_{t,Q}$ を算出している．

度よりドメイン変化による精度低下の影響が小さいことを示唆している．実際，COIL はその次元を 1 次元とした uni-COIL (Lin and Ma 2021) においても，ソースドメインで高い精度となることが知られている．ゆえに，COIL にはトークンの重要度を学習する側面があると考えられる．トークンの重要度を直接学習する DeepCT (Dai and Callan 2020) は，対象データにおいて BM25 より大幅に精度が低下することが知られている (Thakur et al. 2021)．以上から，より文脈を考慮する密ベクトル検索モデルを用いた方が高い精度となったと推察される．

次に，蒸留と多様な負例を用いた密ベクトル検索モデルの場合と比較する．平均的には，負の対数尤度と BM25 の負例を用いた場合の方が平均的に高い精度を示している．蒸留と他の密ベクトル検索の結果を負例に用いた密ベクトル検索モデルは，負の対数尤度と BM25 の負例を用いた場合よりもトークンの一致を考慮した検索結果になっていると考えられる．逆に負の対数尤度と BM25 の負例を用いた場合，負例にトークン一致する文書が多く存在することから，より文脈のみで関連度を学習していると考えれる．C-BM25 においては，トークン一致を明示的に考慮するスコアリングを行うため，より文脈で類似度を判別するモデルの方が高い精度となったと考えられる．

これを分析するために，2.6 節で述べた RSJ 関数を用いた分析を行う．分析の

ため，RSJ 重みの値を区間毎の $\Delta\text{RSJ}_{t,Q}$ 値を図 3.3に示す．図 3.3には，参照先として BM25 の結果も示している．BM25 では RSJ が最大のグループで，$\Delta\text{RSJ}_{t,Q}$ が正になっており，理想よりはトークンの一致を強く考慮した結果になっている．そのため，キーワードをより重視した検索結果であることが伺える．一方，密ベクトル検索は RSJ が最大のグループで，ΔRSJ が負になっており，トークンの一致が考慮されていないことがわかる．その中でも，蒸留と他の密ベクトル検索の結果を負例に用いた密ベクトル検索の方が，正例と BM25 による負例を用いた密ベクトル検索より ΔRSJ が 0 に近い．これは，この学習方法の密ベクトル検索モデルが，トークンの一致を考慮した検索結果となっていることを裏付けている．

3.4.4 教師なし文表現を用いた結果

表 3.6: エンコーダに SimCSE を用いた場合の nDCG@10．最も良い結果を**太字**にしている

	BM25	SimCSE	C-BM25	BM25 併用	C-BM25 併用
MS MARCO	0.506	0.349	**0.527**	0.512	**0.527**
対象データ					
Arguana	0.275	**0.474**	0.465	0.365	0.465
C-FEVER	0.158	0.163	0.197	0.164	**0.199**
DBPedia	0.285	0.162	**0.329**	0.290	0.328
FEVER	0.577	0.257	**0.666**	0.584	0.665
FiQA	0.236	0.159	0.276	0.237	**0.277**
HotpotQA	0.567	0.305	**0.643**	0.572	0.642
NFCorpus	0.330	0.197	**0.336**	0.302	0.317
NQ	0.243	0.175	**0.338**	0.251	**0.338**
Quora	0.789	0.818	0.820	0.805	**0.821**
SCIDOCS	0.140	0.099	0.155	0.141	**0.156**
SciFact	0.664	0.392	**0.696**	0.664	**0.696**
T-COVID	0.530	0.327	0.676	0.535	**0.677**
Robust04	0.409	0.322	0.448	0.416	**0.449**
Touche2020	0.454	0.120	0.370	**0.463**	0.370
対象データ平均	0.404	0.288	**0.463**	0.420	0.462

本節のこれまでの分析により，文脈を考慮するベクトル表現であれば，精度向上に寄与することが考えられる．つまり，文脈類似度はドメイン変化の影響が小さいことが示唆されるため，文脈類似度であれば検索タスクでの学習をしていない

エンコーダでも C-BM25 において有効であると考えられる．そのため，C-BM25 のエンコーダとして，教師なし文表現である SimCSE (Gao et al. 2021) を用いて実験を行った．結果を表 3.4に示す．SimCSE を用いた場合でも，対象データにおいて C-BM25 が BM25 を上回ることがわかる．一方で，SimCSE は対象データにおいて BM25 を大きく下回る．SimCSE による表現が検索タスクでは不適合であることがわかる．さらに，C-BM25 が BM25 併用を大きく上回っており，C-BM25 併用も C-BM25 からの改善が見られない．これらの結果は，C-BM25 は文脈を考慮可能なエンコーダであれば検索に対して不適合な表現ベクトルであっても検索精度の向上に寄与することを支持する．また，教師データを全く用いない場合でも検索精度を向上させられることを示している．MS MARCO は研究用途に限定されたデータセットであり，実際にはソースドメインとなる教師データ構築も困難な場合がある．C-BM25 はそのような場合でも有効な手法である．以上より，一致したトークン周辺の文脈はよりドメイン間で転移が容易であることが示唆される．人間は未知語を含む文の意味を文脈を用いて推定する (Nam et al. 2022)．また，未知語の推定時に文脈語が高頻度であるほど推定精度が高いことが知られている (Schuster et al. 2016)．よって，文脈類似度がドメイン変化の影響が小さいのは，比較的頻度が高くどのドメインにも現れる単語がキーワードの周辺に現れるためと推察される．

3.4.5　併用時の密ベクトル検索との関連度スコアの比率

BM25 併用及び C-BM25 併用において，密ベクトル検索との関連度スコアは単純な和を用いていた．本節では，それぞれ重み付けをすることで，精度向上に対する各モデルの寄与を確認する．重みは，総和が 1 になるようにしつつ，各 0.1 刻みで変動させることで実験を行った．なお，重みが 0 のときは密ベクトル検索のみの精度である．また，重みが 1 のときは，BM25 または C-BM25 の精度である．結果を図 3.4に示す．BM25 併用時は密ベクトル検索の重みが 0.6 で最も nDCG@10 が高く，C-BM25 併用時は 0.4 で最も nDCG@10 が高くなった．どちらの場合も最も精度の良い場合の重みが 0.5 に近く，単純に和をとる場合が比較的有効であることがわかる．

C-BM25 の関連度スコアと密ベクトル検索の関連度スコアの和をとることによる精度の向上について一般性を確認するため，上位 10 件に限らず上位 100 件

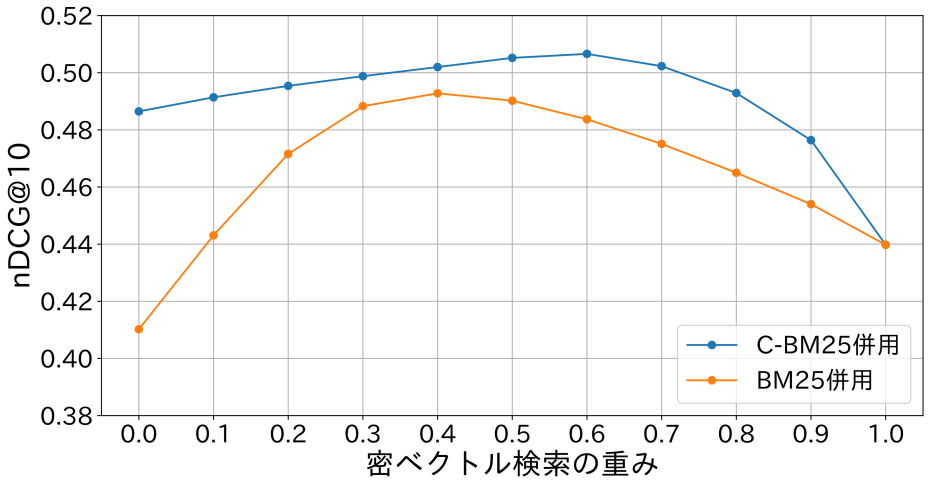


図 3.4: BM25 併用及び C-BM25 併用における密ベクトル検索の重みと nDCG@10 の変化

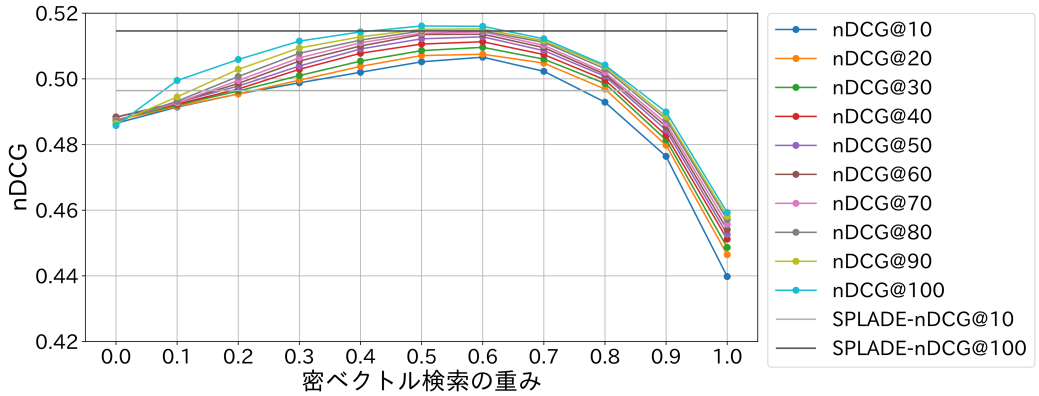


図 3.5: C-BM25 併用における密ベクトル検索の重みと nDCG の変化

まで 10 件ずつ nDCG を評価した．結果を図 3.5に示す．最適な重みについては変動があるものの，概ね上位 10 件の評価と傾向は変わらず，重みが 0.5 付近で最も高い精度を示している．同様に BM25 併用の結果を図 3.6に示す．こちらも上位 10 件の場合の評価と傾向は変わらない．よって，今回使用したモデルにおいて，概ね同等の重みで関連度スコアを足し合わせることが最適に近いことがわかる．C-BM25 併用，BM25 併用のどちらもトークン一致を用いるモデルの関連度スコアの方が，上位と下位の差が大きい方が良い．C-BM25 の全データにおける 1 位

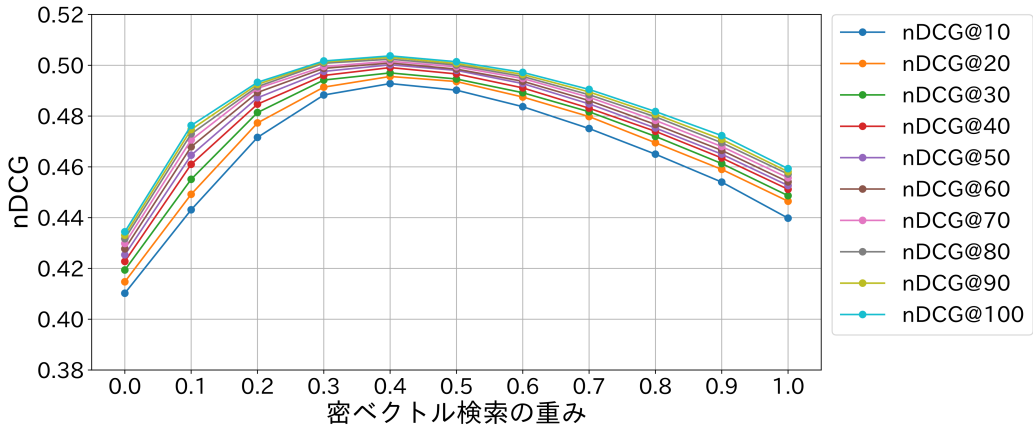


図 3.6: BM25 併用における密ベクトル検索の重みと nDCG の変化

と 100 位の関連度スコアの差は 28.59 であり，BM25 の場合は 10.53 であり，密ベクトル検索の場合は 8.64 であった．そのため，C-BM25 併用や BM25 併用において，C-BM25 及び BM25 というトークン一致を用いるモデルの関連度スコアの方が上位と下位のスコア差が大きい状態を保つ方が良い結果となっていると考えられる．トークン一致を用いるモデルの方が，文書にキーワードを含むかどうかによって，大きく関連度スコア変わるため，トークン一致を用いるモデルにおける上位下位の関連度スコアの差を大きく保つことが重要と思われる．ドメイン外における複数検索アルゴリズムの関連度スコアの統合方法については，今後の研究が求められる．図 3.5にて SPLADE の結果と比較すると，上位 10 件の場合の方が差が大きい．よって，関連度スコアの和をとる方法は，より上位の結果において効果があると考えられる．

3.5 事例を用いた観察

本章では，事例を用いて C-BM25 がどのような場合に精度が向上しているのかを明らかにする．比較対象として，BM25 と平均ベクトルを用いる．成功事例には，両者よりも C-BM25 が nDCG@10 で上回った事例を用いる．失敗事例として，両者よりも C-BM25 が nDCG@10 で下回った事例を用いる．また，事例は各クエリに対するそれぞれの 1 位文書を用いる．

表 3.7: Scifact における各検索手法による 1 位文書の結果．C-BM25 のみが適合と判定された結果を出力している

クエリ	Arginine 90 in p150n is important for interaction with EB1.
BM25 の 1 位文書	CLIP-170 is a "cytoplasmic linker protein" implicated in endosome-microtubule interactions ... In contrast, the CLIP-170/EB1 interaction requires neither metal binding motif. ... We find that CLIP-170 mutants alter p150(Glued) localization without affecting EB1, indicating that EB1 can target microtubule plus ends independently of dynactin.
密ベクトル検索（蒸留）	Listeria monocytogenes is widely used as a model to study immune responses against intracellular bacteria...
C-BM25 のの 1 位文書	Plus-end tracking proteins, such as EB1 and the dynein/dynactin complex, ... Combining crystallography, NMR, and mutational analyses, our studies reveal the critical interacting elements of both EB1 and p150Glued, whose mutation alters microtubule polymerization activity...

表 3.8: 表 3.7のクエリにおける各トークンの IDF と BM25 スコアと文脈類似度．BM25 と C-BM25 の 1 位文書を対象としている．なお，BM25 は pyserini の標準のトークナイザを使用しており，C-BM25 は MPNet のトークナイザを使用しているため，トークン化された結果は異なっている

pyserini のトークナイザ	
クエリの各トークンの IDF スコア	arginin: 5.03, 90: 3.76, p150n: 0.0, import: 1.76, interact: 2.11, eb1: 6.76
BM25 の 1 位文書におけるクエリの各トークンの BM25 スコア	arginin: 0.0, 90: 0.0, p150n: 0.0, import: 0.0, interact: 3.57, eb1: 11.44
MPNet のトークナイザ	
クエリの各トークンの IDF スコア	ar: 3.61, ##gin: 4.79, ##ine: 2.01, 90: 3.33, in: 0.02, p: 1.56, ##15: 4.48, ##0: 2.99, ##n: 1.82, is: 0.31, important: 1.99, for: 0.26, interaction: 2.87, with: 0.26, e: 2.05, ##b: 2.17, ##1: 1.37, .: 0.0
BM25 の 1 位文書におけるクエリの各トークンの BM25 スコア	ar: 0.0, ##gin: 0.0, ##ine: 0.0, 90: 0.0, in: 0.03, p: 1.96, ##15: 5.62, ##0: 3.76, ##n: 0.0, is: 0.39, important: 0.0, for: 0.25, interaction: 4.34, with: 0.4, e: 3.5, ##b: 3.65, ##1: 2.31, .: 0.0
C-BM25 の 1 位文書におけるクエリの各トークンの BM25 スコア	ar: 0.0, ##gin: 0.0, ##ine: 0.0, 90: 0.0, in: 0.02, p: 2.55, ##15: 7.3, ##0: 4.88, ##n: 0.0, is: 0.32, important: 0.0, for: 0.27, interaction: 0.0, with: 0.27, e: 3.57, ##b: 3.77, ##1: 2.39, .: 0.0
BM25 の 1 位文書におけるクエリの各トークンの文脈類似度	ar: 0.0, ##gin: 0.0, ##ine: 0.0, 90: 0.0, in: 0.43, p: 0.45, ##15: 0.46, ##0: 0.47, ##n: 0.0, is: 0.46, important: 0.0, for: 0.45, interaction: 0.49, with: 0.44, e: 0.5, ##b: 0.49, ##1: 0.47, .: 0.46
C-BM25 の 1 位文書におけるクエリの各トークンの文脈類似度	ar: 0.0, ##gin: 0.0, ##ine: 0.0, 90: 0.0, in: 0.49, p: 0.58, ##15: 0.56, ##0: 0.55, ##n: 0.0, is: 0.5, important: 0.0, for: 0.51, interaction: 0.0, with: 0.51, e: 0.55, ##b: 0.55, ##1: 0.54, .: 0.49

3.5.1 成功事例

成功事例として，まず Scifact における，“Arginine 90 in p150n is important for interaction with EB1.” というクエリに対する分析を行った．事例を表 3.7に示す．C-BM25 の 1 位文書は EB1 と p150 の相互作用について言及しており，クエリの内容に関連している．一方，BM25 では，EB1 と p150 について言及されて

いるものの，CLIP-170 が EB1 に影響を与えずに p150 に作用するという話になっており，p150 あるいは Arginine と EB1 の相互作用についての言及ではない．また，密ベクトル検索の結果は蒸留をした場合でも EB1 等の物質への言及がなく，関連がない文書を上位にしている．

より詳細な分析を行うために，BM25 の 1 位文書と C-BM25 の 1 位文書をクエリの各トークンのスコアを用いて比較する．表 3.8にクエリの各トークンの IDF スコアと BM25 スコアおよび C-BM25 の文脈類似度を示す．

まず，BM25 の 1 位文書について，pyserini のトークナイザを用いた場合の BM25 スコアの結果より，EB1 がキーワードとして非常に高いスコアを示している．C-BM25 の 1 位文書については，MPnet のトークナイザを用いた場合の BM25 スコアの結果より，p150n がキーワードとして非常に高いスコアを示している．このように，それぞれキーワードの一致がなされていることがわかる．なぜ，C-BM25 の 1 位文書が BM25 では 1 位文書ではなかったかを考察する．まず，BM25 では，p150n がトークナイズされていない．IDF のスコアが 0 であることから，p150n に一致する文書がないことがわかる．そのため，表記ゆれによって，C-BM25 では 1 位になった関連文書がより上位にならなかったと考えられる．ただし，MPNet のトークナイザを使った場合でも，BM25 の 1 位文書は同じ文書であったため，トークナイザの差異のみとは言えない．次に文脈類似度を見ると，C-BM25 の 1 位文書の方が BM25 の 1 位文書より高くなっている．事例による観察の通り，p150n, EB1, interaction がそれぞれ非常に近くに出現しており，これが寄与していると考えられる．このように，C-BM25 がキーワードの一致と文脈の一致の双方を関連度スコアに反映できていることがわかる．

次に，密ベクトル検索との違いを示す事例として，DBPedia-Entity-Retrieval における"Give me all movies directed by Francis Ford Coppola." というクエリに対する分析を示す．検索結果を表 3.9に示す．C-BM25 と BM25 を比較すると，先ほどと同様にキーワード一致と文脈の一致の双方ができていることで関連文書を上位にランクづけできている傾向が見られる．つまり，共にキーワードである Francis Ford Coppola が監督した作品を出力できている．C-BM25 の一位文書では directed by Francis Ford Coppola と出てくるため，クエリのフレーズと合致しており，文脈類似度が大きくなり，高い関連度スコアになったと推察される．実際，表 3.10より文脈類似度は C-BM25 の 1 位文書の方が BM25 の 1 位文書より高

表 3.9: DBpedia-Entity Retrieval における各検索手法による 1 位文書の結果．C-BM25 と密ベクトル検索（蒸留）が適合と判定された結果を出力している

クエリ	Give me all movies directed by Francis Ford Coppola.
BM25 の 1 位文書	Don't Box Me In Don't Box Me In is a collaboration between Stewart Copeland and Stan Ridgway. It was recorded as part of the soundtrack for the Francis Ford Coppolaś movie Rumble Fish and was subsequently released as a single. Copeland plays guitar, drums, bass and keyboards, and Ridgway sings and plays harmonica.
密ベクトル検索の 1 位文書	All of Me (1984 film) All of Me is a 1984 fantasy comedy film directed by Carl Reiner and starring Steve Martin and Lily Tomlin. This film is based on the novel Me Two by Edwin Davis.
密ベクトル検索（蒸留）の 1 位文書	Francis Ford Coppola Francis Ford Coppola (/'ko ʊpələ/; born April 7, 1939) is an American film director, producer and screenwriter. He was part of the New Hollywood wave of filmmaking. After directing The Rain People in 1969, he won the Academy Award for Best Original Screenplay as co-writer, with Edmund H. North, of Patton in 1970.'
C-BM25 のの 1 位文書	The Godfather The Godfather is a 1972 American crime film directed by Francis Ford Coppola and produced by Albert S. Ruddy from a screenplay by Mario Puzo and Coppola.

表 3.10: 表 3.9のクエリにおける各トークンの IDF と BM25 スコアと文脈類似度．BM25 と C-BM25 の 1 位文書を対象としている．なお，BM25 は pyserini の標準のトークナイザを使用しており，C-BM25 は MPNet のトークナイザを使用しているため，トークン化された結果は異なっている

pyserini のトークナイザ	
クエリの各トークンの IDF スコア	give: 5.55, me: 5.71, all: 3.39, movi: 5.1, direct: 3.57, franci: 5.77, ford: 6.51, coppola: 9.72
BM25 の 1 位文書におけるクエリの各トークンの BM25 スコア	give: 0.0, me: 7.54, all: 0.0, movi: 5.16, direct: 0.0, franci: 5.85, ford: 6.59, coppola: 9.84
MPNet のトークナイザ	
BM25 の 1 位文書におけるクエリの各トークンの文脈類似度	give: 0.0, me: 0.27, all: 0.0, movies: 0.0, directed: 0.0, by: 0.0, francis: 0.34, ford: 0.37, cop: 0.38, ##pol: 0.37, ##a: 0.35, .: 0.22
C-BM25 の 1 位文書におけるクエリの各トークンの文脈類似度	give: 0.0, me: 0.0, all: 0.0, movies: 0.0, directed: 0.48, by: 0.51, francis: 0.53, ford: 0.54, cop: 0.54, ##pol: 0.51, ##a: 0.51, .: 0.49

い．逆に，BM25 の 1 位文書が C-BM25 では 1 位ではないのは，文脈類似度が低くなっているためと考えられる．一方，BM25 の 1 位文書は Francis Ford Coppola が監督した映画ではなく，採用されたサウンドトラックの話になっている．pyserini のトークナイザにおける IDF スコアを見ると，direct のスコアが相対的に小さい．また，me が期待と異なり関連度スコアに大きく寄与してしまっている．

密ベクトル検索と比較する．負の対数尤度と BM25 負例で学習を行った場合の 1 位文書は映画の話であり，誰が監督しているかまで記載されているが，Francis Ford Coppola が監督した作品ではない．密ベクトル検索でも蒸留を用いて学習を行うことで，1 位文書については関連文書とすることができている．図 3.3で示した通り，蒸留や密ベクトル検索における負例を用いることで，キーワードの一致

表 3.11: 表 3.9の密ベクトル検索の 1 位文書およびクエリ中の人名とそのトークンの IDF スコア

文書中の表記	MPNet のトークナイザーによる トークナイズ結果と各トークンの IDF スコア
Francis Ford Coppola	francis: 5.71, ford: 6.43, cop: 6.87, pol: 6.01, a: 2.42
Carl Reiner	carl: 5.98, rein: 6.74, ##er: 3.34
Steve Martin	steve: 5.93, martin: 5.26
Lily Tomlin	lily: 7.8, tom: 5.35, ##lin: 5.65

の問題が緩和されていることがわかる．ただし，この場合においても，nDG@10 において C-BM25 と 0.36 の差がある．実際，２位以降について見てみると，C-BM25 については２位以降にも関連文書が含まれているが，密ベクトル検索については蒸留を用いた場合でも関連文書が含まれていない．キーワードの一致については，依然 C-BM25 の方が確実にできることがわかる．

重み平均プーリングを用いた場合と平均プーリングを用いた場合の比較を行う．差分の比較を行うため，負の対数尤度と BM25 負例で学習を行った場合の事例で分析を行う．重み平均プーリングを用いた場合でもで１位文書同じ結果であった．そこで，Francis Ford Coppola と当該文書に出演する人名のトークンとそのトークンの IDF スコアをみる．結果を表 3.11に示す．IDF スコアは，どの人名でも大きく差がないことがわかる．検索タスクにおいては，知りたい情報を識別するために使用される単語は低頻度になることが想定される．また，この低頻度語と同義と判定される語は少ないことが予想される．よって，このような低頻度語を用いた識別は，ほとんど完全にトークンが一致することと一致した低頻度語に重みを付けることが最も確実な手段となると考えられる．

3.5.2 失敗事例

C-BM25 が BM25 や密ベクトル検索を下回った場合として，DBPedia-Entity Retrieval における，"Give me all launch pads operated by NASA."の事例を示す．表 3.12に，BM25，密ベクトル検索，C-BM25 のそれぞれの１位文書を示す．C-BM25 の１位文書のみ適合文書ではなかった．BM25 と C-BM25 の１位文書を比較すると，キーワードである NASA が共に含まれている．一方，文脈類似度を比

表 3.12: DBpedia-Entity Retrieval における各検索手法による 1 位文書の結果．C-BM25 のみが非適合と判定された結果を出力している

クエリ	Give me all launch pads operated by NASA.
BM25 の 1 位文書	RM-90 Blue Scout II The RM-90 Blue Scout II was an American sounding rocket and expendable launch system which was flown three times during 1961. It was used for two HETS test flights, and the launch of the Mercury-Scout 1 satellite for NASA. It was a member of the Scout family of rockets. The Blue Scout II was a military version of the NASA-operated Scout X-1. All three launches occurred from Launch Complex 18B at the Cape Canaveral Air Force Station, the same launch pad used for the Blue Scout I.
密ベクトル検索（蒸留）	Vandenberg AFB Space Launch Complex 4 Space Launch Complex 4 (SLC-4) is a launch site at Vandenberg Air Force Base with two pads, both of which are used by SpaceX for Falcon 9 launch operations. The complex was previously used by Atlas and Titan rockets between 1963 and 2005. It consisted of two launch pads, SLC-4W and SLC-4E, which were formerly designated PALC2-3 and PALC2-4 respectively. Both pads were built for use by Atlas-Agena rockets, but were later rebuilt to handle Titan rockets.
C-BM25 のの 1 位文書	NASA recovery ship The NASA recovery ships are two ships, the MV Liberty Star and the MV Freedom Star, that are tasked with retrieving spent Solid Rocket Boosters (SRBs) following the launch of Space Shuttle missions. Although owned by NASA, the ships are currently operated by Space Flight Operations contractor United Space Alliance.

表 3.13: 表 3.12のクエリにおける各トークンの IDF と BM25 スコアと文脈類似度．BM25 と C-BM25 の 1 位文書を対象としている．なお，BM25 は pyserini の標準のトークナイザを使用しており，C-BM25 は MPNet のトークナイザを使用しているため，トークン化された結果は異なっている

BM25 の 1 位文書におけるクエリの各トークンの文脈類似度	give: 0.0, me: 0.0, all: 0.37, launch: 0.42, pads: 0.0, operated: 0.42, by: 0.0, nasa: 0.42, .: 0.42
C-BM25 の 1 位文書におけるクエリの各トークンの文脈類似度	give: 0.0, me: 0.0, all: 0.0, launch: 0.47, pads: 0.0, operated: 0.51, by: 0.53, nasa: 0.52, .: 0.53

較すると表 3.13より C-BM25 の方が高い値になっている．特に，NASA について冒頭で繰り返し現れるため高くなっている可能性がある．このように，キーワードの出現場所が近い場合に，文脈類似度が期待とは異なり高くなり，C-BM25 は期待と異なる挙動をする場合がある．

また，密ベクトル検索では，launch pads の部分がマッチングしながら宇宙関連の話題となっているため，関連文書になったと考えられる．関連文書とキーワードが異なる場合に密ベクトル検索の方が優れる場合があると考えられる．しかしながら，この点については，対象ドメインにおいて，平均的にはキーワードの一致を考慮する方が良いことが，3.3.1 項の結果である．クエリ毎に適切なアルゴリズムを選択することは今後の課題と考えられる．

3.6 まとめ

本章では，密ベクトル検索のドメイン適応方法として C-BM25 を提案した．C-BM25 は，トークン一致を用いる検索アルゴリズムでは困難であった文脈の考慮を教師なしで実現すると共に，密ベクトル検索の課題であるクエリと文書におけるキーワード一致を実現する方法である．その有効性を BEIR を用いて検証し，C-BM25 が BM25 や密ベクトル検索から平均的には精度が向上することを明らかにした．さらに，密ベクトル検索の関連度スコアとの和を用いる C-BM25 併用が，他の検索モデルや既存の対象データの重みを用いるドメイン適応手法と比較し，高い精度を示すことを明らかにした．また，応答速度の検証を行い，実用上十分高速に動作することを示した．

さらに，重み付けによるドメイン適応の効果を明らかにするために，BM25 で重み付ける場合，IDF のみで重みづける場合，重み付けを行わない場合を比較した．その結果，BM25 で重み付ける場合が最も良い結果となり，C-BM25 において重み付けに効果があることを明らかにした．また，IDF のみよりも BM25 で重み付ける場合が良い精度となった．BM25 は TF と IDF の双方を考慮していることから，TF も C-BM25 において精度向上に寄与していることを明らかにした．これは，TF はその単語が文書の中心トピックであるかどうかを表すため，文脈とトピックが関連度として双方とも効果的に寄与していることを示唆している．また，クエリと文書で一致したトークンを基準として前後の窓幅内にあるトークンを表現するベクトルを平均することがトークンの表現ベクトルをそのまま用いるよりも効果的であることを確認するために，窓幅の変化による精度の変化を観察した．その結果，一致したトークンを表現するベクトルや全トークンの平均をとったベクトルよりも，一致したトークン周辺で平均を取る場合が最も良い結果となった．

C-BM25 において使用するエンコーダとして密ベクトル検索のモデルが優れていることを確認するためにエンコーダを COIL-tok や蒸留を用いて学習した密ベクトル検索モデルに変えて実験を行った．その結果，負の対数尤度と BM25 負例を用いた密ベクトル検索モデルが比較対象よりも C-BM25 において高い精度となること明らかにした．その要因が，負の対数尤度と BM25 負例を用いた密ベクトル検索モデルがキーワード以外の文脈情報を用いて関連度スコアを算出してい

ることを指摘した．このことから，文脈類似度の方がトークンの重要度よりドメイン変化の影響が小さいことが示唆されるため，文脈類似度であれば検索タスクでの学習をしていないエンコーダでも C-BM25 において有効であると考えられる．そのため，教師なし文表現である SimCSE をエンコーダとして用いた場合についても実験を行った．SimCSE のような，検索では非常に低い精度である文表現も，C-BM25 のエンコーダとして用いることで，BM25 を上回る精度となった．以上より，クエリと文書で一致したトークンの文脈類似度はトークンの重要度よりドメイン変化の影響が小さいことを示唆した．

最後に，事例を用いて具体的にどのような場合に精度が向上し，どのような場合に精度が低下するかを考察した．クエリにキーワードが含まれる場合，クエリと文脈が類似しているものが検索結果の上位となり，本手法が有効であることを示した．一方で，文脈類似度が意図通りではない場合や関連文書にキーワードが含まれていない場合には，本提案手法が有効とは限らないことを示した．

教師なしで精度を向上させる要求は専門的なドメインでドメインより強い．なぜならば，専門性が高いドメインでは，専門家が教師データを作成する必要があるため作成コストが高い場合が多いためである．しかし，密ベクトル検索は，特に専門性の高いドメインでは精度が大きく低下することが指摘されている (Thakur et al. 2021)．そのため，ソースドメイン外でも高精度な検索モデルを用いることが有効と考えられる．しかしながら，C-BM25 は密ベクトル検索にのみ適用対象が限られており，ソースドメイン外でも高精度な検索モデルへの教師なしドメイン適応手法が期待される．

第 4 章

事前学習済み言語モデルに対する語彙追加を用いるドメイン適応

ソースドメイン外でも高精度な検索モデルも，事前学習済み言語モデルを使用している．そのため，事前学習済み言語モデルをドメイン適応することで，密ベクトル検索モデル以外の検索モデルについても精度も向上させられることが期待できる．

事前学習済み言語モデルのドメイン適応において，特に効果的な方法は2.5.2 項で述べた継続事前学習である．さらに，専門的なドメインではソースドメインとで語彙に差があるため，事前学習済み言語モデルに語彙を追加することが効果的と考えられる．特に SPLADE では，その効果が大きいのではないかと考えられる．一つ目として，ドメイン特有の語がサブワードに分解されることを防ぐことで，ノイズを減らすことが可能になる．SPLADE はテキストを事前学習済み言語モデルの語彙サイズ次元のベクトルに変換する．事前学習済み言語モデルは単語より細かい単位の語彙であるサブワードを用いることで，入力から未知語を減らしている．そのため，あるドメイン特有の語は通常の BERT では学習コーパス中で頻度が低いためサブワードに分解される傾向にある．これにより，異なる単語と一致するため，完全一致する単語を含む関連文書が相対的に下位にランクづけられている．例を表 4.1に示す．この例では，“Phytates for the treatment of Cancer” というクエリ中の，Phytates が ph-yt-ates とサブワードに分解される．そのため異なる単語 ph-yt-oche-mic-als と部分一致することで非関連文書が上位にランクづけられている．実際，Formal et al. (2022a) は SPLADE も密ベクトル検

表 4.1: SPLADE においてキーワードが上位文書に含まれていない例

クエリ	Phytates for the Treatment of Cancer
トークナイズされたクエリ	'ph', '##yt', '##ates', 'for', 'the', 'treatment', 'of', 'cancer'
SPLADE の一位文書	Phytochemicals for breast cancer prevention by targeting aromatase. Aromatase is a cytochrome P450 enzyme ...
トークナイズされた文書	'ph', '##yt', '##oche', '##mic', '##als', 'for', 'breast', 'cancer', 'prevention', 'by', 'targeting', 'aroma', '##tase' ...

索同様に対象ドメインで，クエリ中のキーワードと完全一致する文書が上位にランキングされないという傾向があることが報告している．また，上記の課題に対しては，3 章のように対象データにおけるトークンの重要度を用いる方法が効果的と考えれる．なぜならば，ソースドメインと対象ドメインで語彙が異なるのであれば，教師データ中では低頻度なトークンが多いと考えれるためである．二つ目として，ドメイン特有の語に対する同義語拡張が可能になると考えられる．語彙の追加は，継続事前学習の効果を高めることが知られている (Yao et al. 2021)．そのため，語彙を追加して継続事前学習を行うことで，SPLADE がドメイン特有の語に対して同義語拡張を行うことが可能になると期待できる．

よって，本章では語彙や単語頻度がソースドメインから大きく変化するドメインへの適応手法として事前学習済み言語モデルに対する語彙追加を用いることを提案する．概要を図 4.1に示す．具体的には，まず事前学習済みモデルのドメイン適応を行う．方法としては AdaLM (Yao et al. 2021) を用いた．AdaLM は，事前学習済み言語モデルに対して，対象ドメインのコーパスを用いて語彙を追加し，そのコーパスを用いて継続事前学習を行う方法である．そして，AdaLM によってドメイン適応された事前学習済み言語モデルを用いて SPLADE の学習を行う．その後，SPLADE に対して対象データにおけるトークンの重要度を用いるドメイン適応を行う．具体的には，SPLADE でエンコードされたベクトルの各要素に IDF で重みづけをする IDF 重みを提案し，前章で用いた BM25 併用の二つの方法を併せて用いた．

本ドメイン適応手法の有効性を BEIR を用いて検証を行った．AdaLM は分類や固有表現抽出などのタスクで対象データに教師がある場合に有効であることが

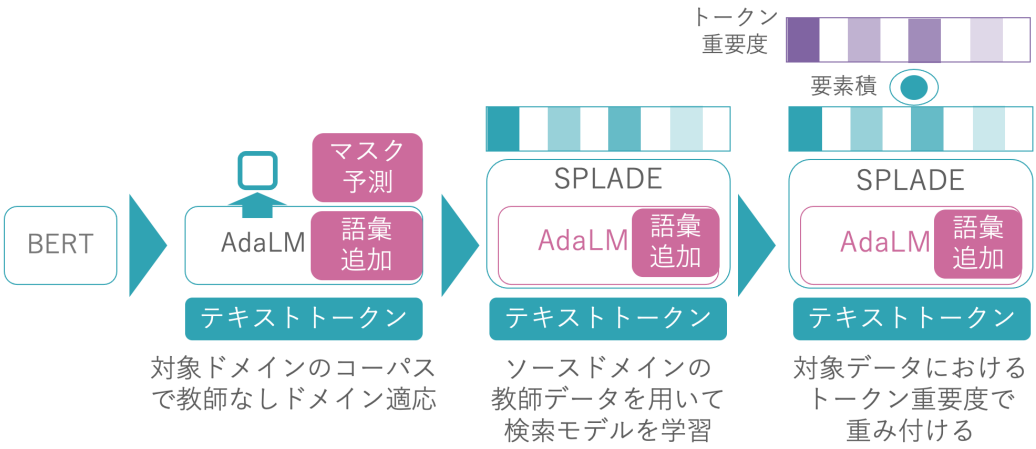


図 4.1: 事前学習済み言語モデルに対する語彙追加を用いるドメイン適応の概要．対象データにおけるトークン重要度を併せて用いる．図は IDF 重みを用いている場合．

知られている (Yao et al. 2021)．しかし，Wang et al. (2022) にて，分類タスクで有効な教師なしドメイン適応手法である UDALM (Karouzos et al. 2021) が検索では有効ではない場合が示されており，検索タスクにおける AdaLM の有効性は自明ではない．検証では，教師データである MS MARCO から語彙と単語頻度の点で最も離れたバイオ・医療と科学の 2 ドメイン 5 データセットを対象データとした．比較対象として，疑似クエリを用いるドメイン適応手法である GPL と比較を行った．GPL は密ベクトル検索で有効な手法であるとされており，SPLADE にも適用可能である．実験の結果，AdaLM を用いた SPLADE は，GPL よりも，高い精度を達成した．さらに，AdaLM を用いた SPLADE に対して，IDF 重みと BM25 併用を用いるを行うことでさらに精度が向上し，既存の最高精度手法である，LaPraDoR (Xu et al. 2022) を平均的には上回る精度となった．また，本手法をクエリに対してエンコーダを用いずトークナイズのみを行った Bag-of-Words 表現のベクトルを用いる場合である SPLADE-Doc に対しても適用した．Bag-of-Words 表現のベクトルを用いる場合，従来のトークン一致を用いる検索と同等の速度やコンピューテーションリソースで検索を実行できることが期待できる．この場合においても提案手法は有効であった．また，IDF 重みは SPLADE-Doc に対して特に有効であり，BM25 併用時も有効であることを示した．

事前学習済み言語モデルのドメイン適応としては，対象ドメインのコーパス

を用いて事前学習する方法があり，クロスエンコーダを用いた実験で精度が向上することが知られている (MacAvaney et al. 2020)．そこで，バイオ・医療と科学のドメインのコーパスを用いて事前学習された言語モデルを用いて，SPLADE の学習を行った．その結果，対象ドメインのコーパスを用いた場合，SPLADE の学習が成功しないことがあり，AdaLM を用いた方が安定して学習を行えることがわかった．

最後に，密ベクトル検索と ColBERT においても本手法を適用し効果の検証を行い，どの検索モデルにおいても効果があることを示した．また，BM25 のみならず，すべての検索モデルをアンサンブルすることで，さらに精度が向上することを示した．

本章では，4.1 節にて提案手法について述べた後，4.2 節にて，有効性を検証する実験設定を述べ，その結果を 4.3 節にてを述べる．各要素の分析を 4.4 節にて述べた後，実際にクエリ中のキーワードと完全一致する文書が上位に来ているかを 4.5 節にて分析する．最後に，4.7 節にて複数の検索モデルに対する AdaLM の効果とアンサンブルの結果を述べる．

4.1 提案手法

本節では，提案手法について述べる．提案手法は，事前学習済み言語モデルに対して AdaLM を用いてドメイン適応したのち，さらに対象データにおけるトークンの重要度を用いる．以下では，まず AdaLM について説明したのち，対象データの重みを用いる方法について説明する．

4.1.1 AdaLM

AdaLM は，事前学習済み言語モデルに対する教師なしドメイン適応手法である．模式図を図 4.2に示す．具体的な手続きとしては，対象ドメインの語彙を追加すると共に，対象ドメインのコーパスを用いて事前学習済み言語モデルの継続事前学習を行う．このとき，追加する語彙も対象ドメインのコーパスを用いて選定する．

AdaLM には以下の三つの効果があることを期待している．一つ目は，本章冒頭の通り，ドメインの語彙を追加することで，ドメイン特有の単語がトークナイズされることを防ぐことである．事前学習済み言語モデルでは，事前学習を行

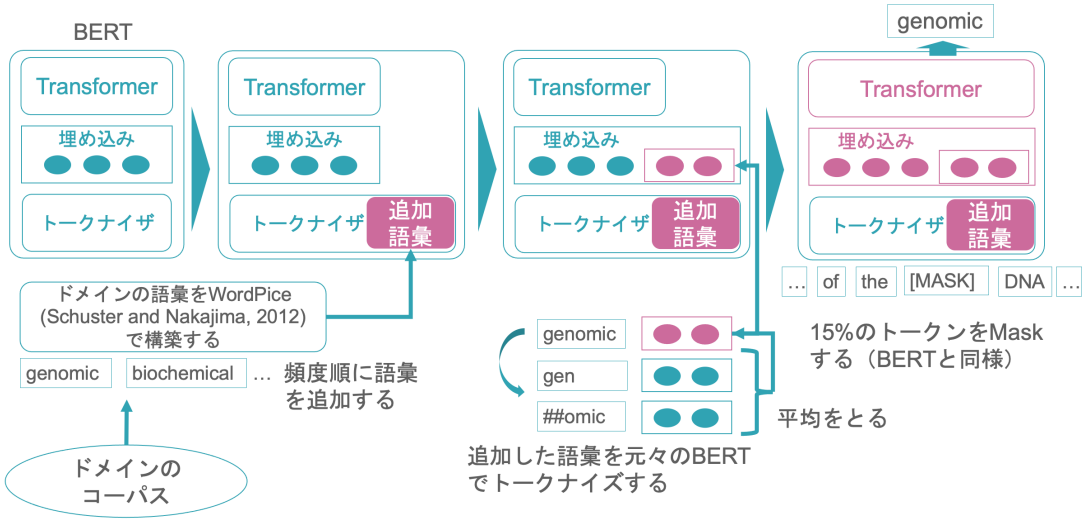


図 4.2: AdaLM の模式図

うコーパス中低頻度な単語については，単語よりもさらに小さい単位であるサブワードに分割することで，入力に未知語が発生することを防ぐ．しかしながら，SPLADE においてはサブワードに分割した結果クエリ中の単語が異なる単語と部分一致することで，非関連文書が検索結果の上位にランキングづけられる．ドメインの語彙を追加することでこれが防止されることを期待している．二つ目は，ドメインの語彙を追加することで，より正確なクエリ拡張・文書拡張を行うことである．SPLADE は，テキストを事前学習済み言語モデルの語彙サイズ次元のベクトルに変換する．そのため，元々のトークン以外の非ゼロ成分は，クエリ拡張・文書拡張を行ったものとみなせる．SPLADE によるテキストのベクトル化は，単語予測層によってなされるため，ドメインの語彙を追加し事前学習を行うことで，より正確な同義語・関連語の予測が可能になると期待できる．結果として，クエリ拡張・文書拡張がより正確になることが期待できる．Bai et al. (2020) は精度向上に寄与するクエリ拡張・文書拡張のうち，同義語・関連語は事前学習によって得られると考察しており，継続事前学習が検索で効果を持つことが期待できる．三つ目は，語彙追加により次元を大きくすることで，精度が向上することである．Jang et al. (2021) は事前学習済み言語モデルを用いてテキストを疎ベクトルに変換する際，次元を増やすことで検索における適合性指標の一つである MRR@10 が上昇すると報告している．語彙追加は SPLADE において，次元を増

やすことに相当しており，同様の効果が期待できる．

それでは，以下に語彙追加と事前学習の実施方法を述べる．

語彙追加　語彙追加では，ドメイン特有の語彙の追加を試みる．そのために，追加する語彙を取得する．まず対象ドメインのコーパスにおいて語彙サイズを指定して，WordPiece (Mike and Kaisuke 2012) を用いてトークナイザを構築することで語彙を得る．次に，構築したトークナイザの語彙を事前学習済み言語モデルの語彙に追加する．語彙の追加は，対象コーパスにおいて頻度の高い語から行い，指定した語彙サイズになったら停止する．この時，すでに事前学習済み言語モデルに存在する語はスキップする．追加する際に，数字や記号のみのトークンはノイズとして除去した．この手続きを，指定した語彙サイズを 3k ずつ増やし，指定サイズに語彙数が到達しなくなったら停止した．以上をアルゴリズム 1に示す．WordPiece はスペース区切りで得られた各単語を文字毎に数え上げ，自己相互情報量が高いサブワードペアマージする[1]．よって，このアルゴリズムで，対象ドメインにおいては頻度が比較的高い語彙を追加することが期待される．比較的頻度が高い語であれば，継続学習によって同義関係を獲得することも期待できる．

継続事前学習　継続事前学習 (Gururangan et al. 2020) は，事前学習済み言語モデルに対する有効な教師なしドメイン適応手法として知られている．方法は，事前学習済み言語モデルに対して，対象ドメインのコーパスを用いて事前学習を行うことである．今回は，マスク言語モデルの事前学習を行なっている．つまり，ランダムにトークンをマスクトークンに置き換え，元々のトークンをモデルが予測することで学習する．マスク方法は BERT に従い 15%とした．

4.1.2　対象データにおけるトークン重要度を用いるドメイン適応

AdaLM を用いてドメイン適応した事前学習済み言語モデルに対して，SPLADE を学習したのち，対象データの重みを用いてドメイン適応を行う．対象データの重みを用いることで，ソースドメインの教師データ中で低頻度なキーワードを含む文書を上位にランキングすることが期待される．本章で用いた方法は BM25 併用

[1] https://huggingface.co/learn/nlp-course/chapter6/6?fw=pt (2023 年 10 月にアクセス)

アルゴリズム 1 語彙追加の手続

INPUT: Original vocabulary $\mathcal{V}_0$, a domain corpus C, incremental vocabulary size ΔV
OUTPUT: $\mathcal{V}_{\text{final}}$
Set iterating index $i = 0$
repeat
  $i = i + 1$
  $\mathcal{V}_i = \mathcal{V}_{i-1}$
  Set target vocabulary size $V_i = |\mathcal{V}_0| + i * \Delta V$
  Build WordPiece tokenizer $\mathcal{T}_i$ at the vocabulary size of V_i on C.
  Get vocabulary $\acute{\mathcal{V}}_i$ from $\mathcal{T}_i$
  Tokenize C by $\mathcal{T}_i$ and count tokens
  Sort $\acute{\mathcal{V}}_i$ by frequency
  Set new vocabulary $\mathcal{V}_i$ by adding words to $\mathcal{V}_0$ from frequent words until $|\acute{\mathcal{V}}_i| < V_i$ except for duplicate words and words consisting of only number of mark.
until $|\mathcal{V}_i| - |\mathcal{V}_{i-1}| < \Delta V$
return $\mathcal{V}_{\text{final}} = \mathcal{V}_i$

と IDF 重みである．以下では，IDF 重みについて説明する．BM25 併用は 2.5.1 項を参照されたい．

IDF 重み　IDF 重みは，SPLADE がエンコードしたベクトルの各要素に，対象データの文書を用いて計算した IDF 値をかける方法である．数式で表すと以下の通りである．まず，文書数を N とし，トークン t を含む文書数を N_t とする．IDF の値を要素にもつベクトルを $\boldsymbol{w}^{\text{IDF}} \in \mathbb{R}^{|\mathcal{V}|}$ とすると

$$\boldsymbol{w}_t^{\text{IDF}} = \begin{cases} \log \frac{N}{N_t} & \text{if } N_t \neq 0 \\ 1 & \text{otherwise} \end{cases}. \tag{4.1}$$

となる．なお，$N_t = 0$ の場合は SPLADE の重みをそのまま用いるため IDF 重みは 1 とした．式 (2.51) により，SPLADE によってエンコードされた文書のベクトルをここでは，$\boldsymbol{v}'_{\text{SPLADE}}$ とすると，重み付けられたベクトル $\hat{\boldsymbol{v}}'_{\text{SPLADE}}$ は式 (4.2) のようになる．

$$\hat{\boldsymbol{v}}'_{\mathrm{SPLADE}} = \boldsymbol{w}^{\mathrm{IDF}} \odot \boldsymbol{v}'_{\mathrm{SPLADE}}. \tag{4.2}$$

なお，$\odot$ は要素ごとの積を表す．なお，上記では文書のベクトルに重み付けしているが，クエリのベクトルに重み付けをしても良い．

4.1.3 検索モデルおけるドメイン適応としての AdaLM

AdaLM 及び継続事前学習が検索におけるドメイン適応としてどのように機能するかを説明する．まず，ソースドメインにおける検索モデルの学習は，式 (3.6) を用いると式 (4.3) のように定式化される．

$$R(h) = \int_{R,Q,D} l(h(q,d),r) p_{\mathcal{S}}(r,q,d)\, drdqdd \tag{4.3}$$

今，式 (2.48) の記号を援用して，ドメイン適応前のエンコーダを $\eta_{\mathcal{S}}$ とすると，

$$h(q,d) = \phi(\eta_{\mathcal{S}}(\boldsymbol{q}), \eta_{\mathcal{S}}(\boldsymbol{d})) \tag{4.4}$$

と書ける．AdaLM や継続事前学習でドメイン適応したエンコーダを $\eta_{\mathcal{T}}$ と書くと，検索モデルの学習は式 (4.5) のように書ける．

$$R(\eta) = \int_{R,Q,D} l(\phi(\eta_{\mathcal{T}}(\boldsymbol{q}), \eta_{\mathcal{T}}(\boldsymbol{d}),r) p_{\mathcal{S}}(r,q,d)\, drdqdd \tag{4.5}$$

よって，直観的にも明らかなように，検索モデルの学習としてはエンコーダの初期値のみしか変化していない．エンコーダの変化は事前学習によるものであり，事前学習によってソースドメインの入力 $p_{\mathcal{S}}(q,d)$ から対象ドメインの入力 $p_{\mathcal{T}}(q,d)$ への変化に適応したとみなせる．また，$p(q,d)$ のドメインシフトであるということは，$p(d)$ のドメインシフトでもあるため，対象文書のドメインシフトにも適応していると考えられる．

4.2 実験設定

提案手法の有効性を検証するために，実験を行った．本節では，ベースライン手法を述べたのち，評価を行うデータセットと評価方法について述べる．最後に，モデルの学習方法の詳細を述べる．

4.2.1 ベースライン手法

提案手法の有効性を検証するために，他の検索モデル及び他のドメイン適応方法と比較した．他の検索モデルとして，密ベクトル検索，ColBERT，クロスエンコーダと比較した．各手法の詳細は 2.3 節を参照のこと．また，Bag-of-words 表現を用いる検索モデルとして BM25 及び DocT5query(Nogueira et al. 2019) を用いた．DocT5query はクエリ生成モデルを用いて文書拡張をする方法で，検索のランキングには BM25 を用いる．なお，これ等の結果は BEIR(Thakur et al. 2021) から引用した．これ等は，同様に BoW 表現を用いる SPLADE-Doc と比較する．さらに，トークン一致を用いる検索アルゴリズムとの併用を用いる既存の手法をして，C-BM25 併用と LaPraDor (Xu et al. 2022) と比較を行った．LaPraDor は，教師なし事前学習を行ったのち MS MARCO で学習した密ベクトル検索モデルのスコアと BM25 のスコアのスコアを掛け合わせる手法である．他のドメイン適応方法として，GPL (Wang et al. 2022) と比較した．GPL は密ベクトル検索において，最も有効な教師なしドメイン適応手法である．今回は，GPL を SPLADE に対しても適用する実験を行った．

4.2.2 データセットと評価方法

本実験では，ベンチマークデータセットとして前章同様に BEIR を用いた．BEIR の実験設定同様に，教師データとして MS MARCO を用いる．また，検索精度は nDCG@10 と Recall@100 を用いて評価を行った．検定は，対応のある t 検定 (酒井 2015) を行い，多重検定の補正は Benjamini-Hochberg 法 (瀬々・浜田 2015) で行った．対象データとして，BEIR のうち重みつき Jaccard 係数の点で教師データである MS MARCO から最も離れたデータである 5 つのデータセットを用いた．ソースドメインのデータセットを A，対象ドメインのデータセットを B とすると，重

表 4.2: BEIR の各データセットと MS MARCO の重みつき Jaccard 係数

データセット	重みつき Jaccard 係数
Natural Question	0.523
Robust04	0.475
Touche-2020	0.410
FiQA	0.407
Quora	0.395
Arguana	0.385
Climate-FEVER	0.384
FEVER	0.384
HotpotQA	0.342
DBPedia-Entity	0.334
SCIDOCS	0.327
BioASK	0.317
TREC-COVID	0.315
NFCorpus	0.285
SciFact	0.273

みつき Jaccard 係数 $J(A, B)$ は式 (4.6) で定義される．

$$J(A, B) = \frac{\sum_t \min(A_t, B_t)}{\sum_t \max(A_t, B_t)}. \tag{4.6}$$

なお，A_t はデータセット A におけるトークン t の正規化した頻度である．また B_t はデータセット B におけるものである．

5つのデータセットは，具体的には BioASQ (BASK)，NFCorpus (NFC)，TREC-COVID (TCOVID)，SCIDOCS (SDOCS)，Scifact (Sfact) を用いた．BEIR 中の対象データと MS MARCO の重みつき Jaccard 係数を表 4.2に示す．BioASK，NFCorpus，TREC-COVID は，バイオ・医療ドメインのデータセットである．また，SCIDOCS，Scifact は，科学ドメインのデータセットである．

AdaLM を行うためのドメインコーパスとして，バイオ・医療ドメインに対しては PubMed[2] の要旨を用いた．その際，PubmedBERT と同様に 128 単語以上の文書を用いた．結果としてコーパスサイズは 17GB となった．科学ドメインのデータセットには S2ORC (Lo et al. 2020) の要旨を用いた．バイオ・医療ドメイ

[2]https://pubmed.ncbi.nlm.nih.gov/

表 4.3: 各検索モデルの学習時のハイパーパラメータ

	SPLADE	密ベクトル検索	ColBERT
バッチサイズ	40	64	32
最長文書長	256	300	220
学習率	2e-5	2e-5	3e-6
エポック	30	30	1
Warmup ステップ数	1000	1000	-
最大学習ステップ数	-	-	500,000

ン同様に 128 単語以上の文書を用いた．結果としてコーパスサイズは 7.3GB となった．なお，バイオ・医療ドメインにおいて AdaLM を行った際の語彙サイズ $\mathcal{V}_{\text{final}}$ は $71,694$ となった．また，科学ドメインでは $62,783$ となった．

4.2.3 モデルの訓練方法

SPLADE の学習にはマージン二乗損失による蒸留を用いた．蒸留を行うためのスコア関数として，クロスエンコーダ[3] を用いた．負例は，BM25 と既存の密ベクトル検索モデルによる負例[4] を用いた．学習時のハイパーパラメータは表 4.3に記す．SPLADE-Doc は，SPLADE で学習を行なったモデルに対して，検索時のみそのスコアリング方法を用いた．本章では，4.7 節にて密ベクトル検索と ColBERT の学習も行なっている．学習時のハイパーパラメータも併せて表 4.3に記す．密ベクトル検索の学習は SPLADE と同様に行っており，損失関数にはマージン二乗損失を用いた．負例にも同じものを使用した．ColBERT の学習はSanthanam et al. (2022) と同様に，損失関数には KL ダイバージェンスによる蒸留を行った．蒸留元のクロスエンコーダは，SPLADE と同じものを用いた．負例については，BM25 による負例と負の対数尤度で学習した ColBERT による検索結果の上位を負例として用いた．SPLADE と密ベクトル検索の学習に使用した GPU は NVDIA A100 40GB を 1 枚用いている．ColBERT には NVIDIA A100 40GB を 8 枚用いている．

継続事前学習に対しては，bert-base-uncased[5] から開始し，実装は Transform-

[3]https://huggingface.co/BeIR/query-gen-msmarco-t5-base-v1を使用した
[4]Sentence Tansformers で配布されている負例 （https://huggingface.co/datasets/sentence-transformers/msmarco-hard-negatives/resolve/main/msmarco-hard-negatives.jsonl.gz)
[5]https://huggingface.co/bert-base-uncased

表 4.4: GPL 使用時の学習時のハイパーパラメータ

バッチサイズ	24
最大文書長	350
学習率	2e-5
最大学習ステップ数	140000
Warmup ステップ数	1000

ers[6] を用いた．NVDIA A100 40GB を 8 枚用いて学習を行なった．バッチサイズは 1GPU あたり 32 とした．また，最大長を 512 とした．その他の設定は，Transformers の Trainer[7] のデフォルト値を使用した．

GPL では，クエリ生成が必要となる．クエリ生成モデルは (Wang et al. 2022) に従い，MS MARCO で学習した T5[8] を用いた．生成時には top-k サンプリングと累積確率サンプリング（top-k: 25; top-p: 0.95）を用いた．各文書ごとに 3 クエリ生成し，計算コスト削減のため生成対象の文書は 1M までとした．GPL の学習時のハイパーパラメータは表 4.4に記す．

4.3 結果

本節では，提案手法の有効性を検証するため，4.2.1 項で述べたベースラインとの比較及び GPL との比較結果を示す．

4.3.1 ベースラインとの比較

nDCG@10 の結果を表 4.5に示す．まず，SPLADE に AdaLM を適用することで，SPLADE よりも全データで精度が向上しており，4 つのデータセットで統計的に有意である．また，IDF 重みを用いることでさらに精度が向上している．その結果，既存手法で最も精度の良いクロスエンコーダと平均的に同等な精度を達成している．ただし，IDF 重みは Scifact において AdaLM を適用した SPLADE の精度を下げており，統計的に有意なデータは三つとなっている．次に，クエリが Bag-of-words 表現である SPLADE-Doc の結果を見ると，提案手法適用前は

[6]https://github.com/huggingface/transformers
[7]https://huggingface.co/docs/transformers/main_classes/trainer
[8]https://huggingface.co/BeIR/query-gen-msmarco-t5-base-v1

表 4.5: ベースラインと提案手法の比較．評価指標は nDCG@10．最も良い結果を**太字**にしている．また，同カテゴリで最も良い結果を斜体にしている．ドメイン適応前のモデルに対して統計的に有意である場合はに#を数値の右上に記載している．また，トークン一致との併用の手法に対して，LaPraDor に対して有意である場合は L を，C-BM25 に対して有意である場合は C を記載している

	バイオ・医療ドメイン			科学ドメイン		
	BASK	NFC	TCOVID	SDOCS	SFact	平均
密ベクトル検索	0.377	0.301	0.716	0.144	0.571	0.422
SPLADE	0.503	0.336	0.627	0.155	0.691	0.462
ColBERT	0.500	0.327	0.723	0.154	0.645	0.462
Cross Encoder	0.523	0.350	*0.757*	*0.166*	0.688	*0.497*
SPLADE+AdaLM	$0.521^{\#}$	$0.348^{\#}$	$0.715^{\#}$	0.156	*0.716*$^{\#}$	0.491
SPLADE+AdaLM +IDF 重み	*0.544*$^{\#}$	*0.353*$^{\#}$	$0.719^{\#}$	0.161	0.708	*0.497*
クエリが Bag-of-words 表現						
BM25	0.515	0.335	0.581	0.148	0.674	0.451
docT5query	0.431	0.328	*0.713*	*0.162*	0.675	0.462
SPLADE-Doc	0.488	0.323	0.539	0.147	0.678	0.435
SPLADE-Doc +AdaLM+IDF 重み	*0.551*$^{\#}$	*0.342*$^{\#}$	$0.633^{\#}$	*0.162*$^{\#}$	*0.715*$^{\#}$	*0.480*
トークン一致を用いる検索アルゴリズムとの併用						
LaPraDor	0.511	0.347	**0.779**	**0.185**	0.697	0.504
C-BM25 併用	0.526	0.331	0.762	0.165	**0.720**	0.499
SPLADE-Doc+AdaLM +IDF 重み+BM25 併用	$0.573^{\#\mathrm{LC}}$	$0.348^{\#\mathrm{C}}$	$0.653^{\#}$	$0.163^{\#}$	$0.714^{\#}$	0.490
SPLADE+AdaLM +IDF 重み+BM25 併用	**0.577**$^{\#\mathrm{LC}}$	**0.356**$^{\#\mathrm{C}}$	$0.746^{\#}$	$0.165^{\#}$	$0.718^{\#}$	**0.512**

SPLADE-Doc が BM25 より精度が低かった．しかし，提案手法を適用することで，SPLADE-Doc の精度が全てのデータで SPLADE-Doc から統計的に有意に改善している．また，BM25 や提案手法適応前の SPLADE を上回っている．よって，本提案手法は Bag-of-words 表現のクエリに対しても有効であることがわかった．最後にトークン一致を用いる検索アルゴリズムとの併用を用いる結果を見ると，提案手法を適用した SPLADE が C-BM25 併用や BM25 併用を用いる最高精度の手法である，LaPraDor を平均的には上回っている．一方で，LaPraDor や C-BM25 の方が精度が高いデータセットも存在しており，提案手法が LaPraDor に対して統計的に有意なデータセットは BioASK のみであり，C-BM25 に対して統計的に有意なデータセットは BioASK と NFCorpus のみであった．LaPraDor

表 4.6: ベースライン検索モデルと提案手法の比較．評価指標は Recall@100．最も良い結果を**太字**にしている．また，同カテゴリで最も良い結果を斜体にしている．ドメイン適応前のモデルに対して統計的に有意である場合はに#を数値の右上に記載している．また，トークン一致との併用の手法に対して，LaPraDor に対して有意である場合は L を，C-BM25 に対して有意である場合は C を記載している．なお，TREC-COVID のみ関連文書が多いため，BEIR に従い Capped Recall@100 を用いている．また，クロスエンコーダは BM25 の Top100 のリランキングであり，BM25 と同じであるため除いた

	バイオ・医療ドメイン			科学ドメイン		
	BASK	NFC	TCOVID	SDOCS	SFact	平均
密ベクトル検索	0.589	0.258	0.481	0.330	0.879	0.507
SPLADE	0.733	0.279	0.515	0.362	0.924	0.563
ColBERT	0.716	0.265	0.538	0.354	0.888	0.552
SPLADE+AdaLM	0.774$^{\#}$	0.294$^{\#}$	0.597$^{\#}$	0.360	0.945	0.594
SPLADE+AdaLM +IDF 重み	*0.788*$^{\#}$	*0.297*$^{\#}$	*0.618*$^{\#}$	*0.372*$^{\#}$	*0.949*$^{\#}$	*0.605*
クエリが Bag-of-words 表現						
BM25	0.714	0.250	0.498	0.356	0.908	0.545
docT5query	0.646	0.253	*0.541*	0.360	0.914	0.543
SPLADE-Doc	0.723	0.257	0.433	0.346	0.917	0.542
SPLADE-Doc+AdaLM +IDF 重み	*0.785*$^{\#}$	*0.283*$^{\#}$	0.531$^{\#}$	*0.368*$^{\#}$	*0.952*$^{\#}$	*0.584*
トークン一致を用いる検索アルゴリズムとの併用						
LaPraDor	0.716	0.273	0.506	**0.410**	0.921	0.565
C-BM25 併用	0.740	0.298	0.496	0.359	0.945	0.568
SPLADE-Doc+AdaLM +IDF 重み+BM25 併用	**0.826**$^{\#LC}$	0.284$^{\#L}$	0.529$^{\#LC}$	0.371$^{\#C}$	0.948$^{\#L}$	0.591
SPLADE+AdaLM +IDF 重み+BM25 併用	0.821$^{\#LC}$	**0.298**$^{\#L}$	**0.693**$^{\#LC}$	0.377$^{\#C}$	**0.955**$^{\#L}$	**0.611**

も C-BM25 併用も密ベクトル検索を用いることから，密ベクトル検索と SPLADE でソースドメインでの学習を通じて獲得した特徴のうち，汎化できる側面が異なることが示唆される．これを確認するため，4.7.2 項にて BM25 のみならす，異なる検索モデル同士のアンサンブルを行い更なる精度向上が見られるか検証する．

次に Recall@100 の結果を表 4.6に示す．傾向は概ね nDCG@10 と同様である．なお，Recall@100 の方がベースラインからの精度向上幅が nDCG@10 大きいことがわかる．SPLADE に AdaLM と IDF 重みを適用した場合，全てのデータで SPLADE から統計的に有意に精度が向上している．また，さらにトークン一致を

表 4.7: AdaLM と GPL の比較．評価指標は nDCG@10． 最も良い結果を**太字**にしている．また，同カテゴリで最も良い結果を斜体にしている

	バイオ・医療ドメイン		科学ドメイン		
	NFC	TCOVID	SDOCS	SFact	平均
SPLADE					
ドメイン適応なし	0.336	0.627	0.155	0.691	0.452
GPL	0.319	0.708	**0.171**	0.676	0.469
AdaLM	**0.348**	0.715	0.156	*0.716*	**0.484**
AdaLM+GPL	0.330	**0.719**	0.169	0.677	0.474
SPLADE-Doc					
ドメイン適応なし	0.323	0.539	0.147	0.678	0.422
GPL	0.305	0.562	*0.153*	0.649	0.417
AdaLM	*0.332*	*0.593*	0.148	*0.686*	*0.440*
AdaLM+GPL	0.319	0.566	0.146	0.621	0.413

用いる検索アルゴリズムとの併用した場合についても，全てのデータで LaPraDor か C-BM25 併用のどちらかに対しては統計的に有意に精度が向上している．

4.3.2 GPL との比較

4.2.1 項にて，SPLADE に対する提案手法の有効性を示した．本項では，ドメイン適応手法の比較として，GPL と AdaLM の比較を行う．なお，計算コストの都合から，BioASK は除いている．また，nDCG@10 と Recall@100 の傾向が類似しているため，nDCG@10 のみを示す．

結果を表 4.7に示す．GPL でも SPLADE からの精度向上が見られているが，平均的には AdaLM が GPL よりも高い精度を示している．ただし，SCIDOCS については，GPL の方が精度が高くなっている．SPLADE-Doc においては，GPL によって精度が低下している．よって，AdaLM の方が語彙や単語頻度によるドメインシフトに適応する方法として優れていることがわかる．

AdaLM と GPL を共に適用することで，更なる精度向上が見られるか実験を行った．併せて結果を表 4.7に示す．平均的には精度が低下しており，SPLADE に対しては AdaLM のみを適用することが有効であると考えられる．TREC-COVID に注目すると，ドメイン適応を行わない場合から GPL のみを適用した場合の精度の向上よりも AdaLM のみを適用した場合から AdaLM と GPL を共に適用した

表 4.8: 事前学習済み言語モデルに対するドメイン適応手法の比較．評価指標は nDCG@10．ベースモデルは SPLADE．最も良い結果は**太字**にしている

	バイオ・医療ドメイン		科学ドメイン		
ドメイン適応方法	NFC	TCOVID	SDOCS	SFact	平均
ドメイン適応なし	0.336	0.627	0.155	0.691	0.452
継続事前学習のみ	0.335	0.681	0.152	0.700	0.467
語彙拡張のみ	0.346	0.631	0.156	0.676	0.452
ドメイン特化	0.000	0.000	**0.162**	**0.729**	0.223
AdaLM	**0.348**	**0.715**	0.156	0.716	**0.484**

場合の精度の向上幅が小さくなっている．2.5.4 項にて汎化誤差と確率分布を用いて議論を行った通り，GPL は文書集合のドメインシフトに対する手法であるとみなせる．一方，AdaLM は主に語彙や単語頻度におけるドメインシフトに適応する方法である．そして，4.1.3 項より語彙や単語頻度におけるドメインシフトはクエリ集合及び文書集合のドメインシフトの 1 つである．今回，検証の対象としたデータセットは検索対象文書のドメインシフトが大きい．このことから，GPL と AdaLM はデータセットによっては同様に検索対象文書のドメインシフトに対して有効な適応方法であることが伺える．一方，その他のデータセットについては SPLADE に GPL を適用した場合と AdaLM と SPLADE の双方を用いた場合で，傾向が変わらなかった．具体的には，NFCorpus と Scifact では精度が低下し，SCIDOCS では精度が向上した．そのため，全てのデータセットで同様であるとは限らない．4.7.1 項にて示すように，AdaLM と GPL の双方を用いることで精度改善がするデータセットは検索モデルによって異なる．そのため，これらの挙動については確率分布による議論から予測される，GPL は検索対象文書のドメインシフトで精度向上させる，GPL と AdaLM で同様のドメインシフトに有効であり共に用いても効果が小さい，といったこととは異なる要因があると考えられる．

4.4 各要素の影響分析

4.4.1 事前学習済み言語モデルに対するドメイン適応手法の比較

AdaLM における個々の要素影響を調べるため，継続事前学習のみ行なった場合と語彙追加のみ行なった場合の実験を行なった．また，対象のドメインに合わせ

表 4.9: FiQA における SPLADE への AdaLM の効果．最も良い結果を**太字**にしている

	nDCG@10	Recall@100
SPLADE		
ドメイン適応なし	**0.336**	0.609
継続事前学習	0.325	0.613
AdaLM	0.334	**0.615**

た言語モデルを作成する方法として，対象ドメインのコーパスを用いて事前学習を行う方法がある (Beltagy et al. 2019, Gu et al. 2021)．そのため，対象ドメインのコーパスを用いて事前学習を行ったモデルについても実験を行なった．なお，対象ドメインのコーパスで事前学習を行ったモデルとして，PubmedBERT[9] と SciBERT[10] を使用した．

結果を表 4.8に示す．継続事前学習のみでも平均的には精度が向上している．しかし，AdaLM には及ばなかった．一方，語彙追加のみを行なっても平均的には精度の向上は見られなかった．語彙追加は継続事前学習の効果を高めていると考えられる．また，次元が大きいこと自体は対象データに教師データが存在しない場合に，検索精度の改善に寄与しないと考えられる．

対象ドメインのコーパスで事前学習を行なったモデルを用いた場合は，科学ドメインでは AdaLM より SciBERT が高い精度を示した．しかし，PubmedBERT では SPLADE の学習に失敗している[11]．PubmedBERT は医療ドメインの語彙が中心であると考えられる．ソースドメインと事前学習済み言語モデルの語彙が大きく異なる場合，SPLADE の学習が困難になると考えられる．よって，AdaLM はより安定して検索モデルの学習が可能であると言える．

4.4.2 語彙や単語頻度の変化が小さい場合

提案手法の効果が語彙や単語頻度の変化によるドメインシフト限定的であるか調べるため FiQA にて評価を行なった．FiQA は BEIR の対象データの一つであり，

[9]https://huggingface.co/microsoft/BiomedNLP-PubMedBERT-base-uncased-abstract
[10]allenai/scibert_scivocab_uncased
[11]FLOPS の正則化係数を小さくしたり，pubmedbert で microsoft/BiomedNLP-PubMedBERT-base-uncased-abstract-fulltext を使用するなどしたが，共に学習に失敗した

金融ドメインの QA データである．一方で，表 4.2より語彙や単語頻度の面での変化は小さい．

AdaLM を学習する金融ドメインのコーパスとして，TRC2 コーパス (Lewis et al. 2004) を使用した．FiQA における実験結果を表 4.9に示す．SPLADE では，AdaLM でドメイン適応を行なっても，FiQA において nDCG@10 は向上しなかった．また，Recall@100 もその向上幅は小さい．AdaLM はドメインシフトが語彙や単語頻度によって引き起こされる場合に有効であると考えられる．

4.4.3 語彙数による精度の変化

4.4.1 項において語彙を追加することで精度が向上することが明らかになった．ここでは，語彙数の変化による精度の変化を調べる．

AdaLM において追加する語彙数の影響を調べるため，より小さい語彙数での実験を行った．小さい語彙数の選択基準として，ユニグラム言語モデルのエントロピーを用いた．ユニグラム言語モデル $p(t)$ は，コーパス C 内の全トークンの頻度を $N \in \mathbb{Z}$，トークン t の頻度 $c(t) \in \mathbb{Z}$ とすると，$p(t) = \frac{c(t)}{N}$ である．このエントロピーは，テキストのトークン列を $\boldsymbol{t} = (t_1, t_2, ..., t_L)$ とすると，

$$I(\boldsymbol{t}) = \sum_{i=1}^{L} -\log p(t_i) \tag{4.7}$$

となる．また，コーパス全体のユニグラム言語モデルのエントロピーは，

$$I(C) = \sum_{\boldsymbol{t} \in C} I(\boldsymbol{x}) \tag{4.8}$$

となる．

さて，4.1.1 項にて述べたように，語彙数を BERT のトークナイザからステップ毎に増やしている．各ステップでの語彙数を $\mathcal{V}_i$ とし，各ステップ毎のエントロピーを $I_i(C)$ とする．語彙数の変化によるエントロピーの変化を図 4.3に示す．

語彙数の選択基準として三つの基準を用いた．一つ目は，Yao et al. (2021) で用いられた基準であり，$\frac{I_i(C) - I_{i-1}(C)}{I_{i-1}(C)} < \epsilon_1$ である．なお，Yao et al. (2021) と同様に ϵ_1 は 0.01 とした．結果として語彙数は $42,522$ となった．また別の基準として，$I_i(C) - I_{i-1}(C) < \epsilon_2(I_1(C) - I_0(C))$ とした．なぜならば，図 4.3より，$I_1(C) - I_0(C)$

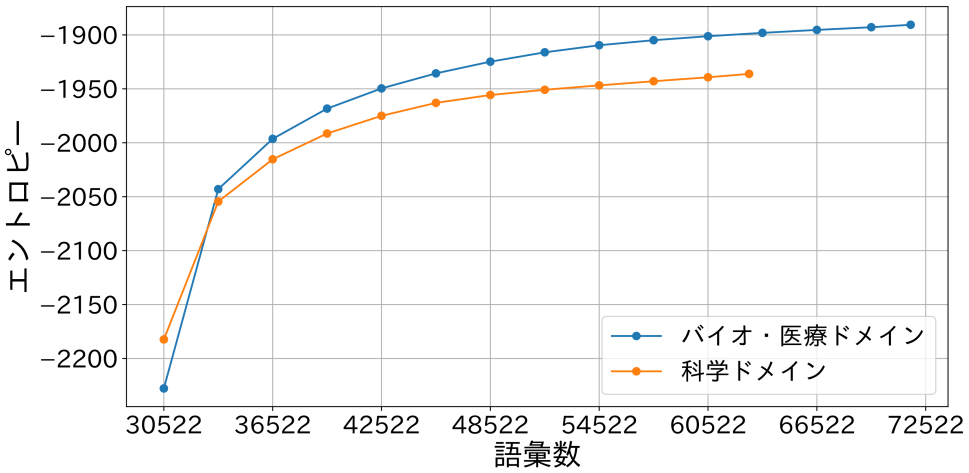


図 4.3: 語彙数によるユニグラム言語モデルのエントロピーの変化

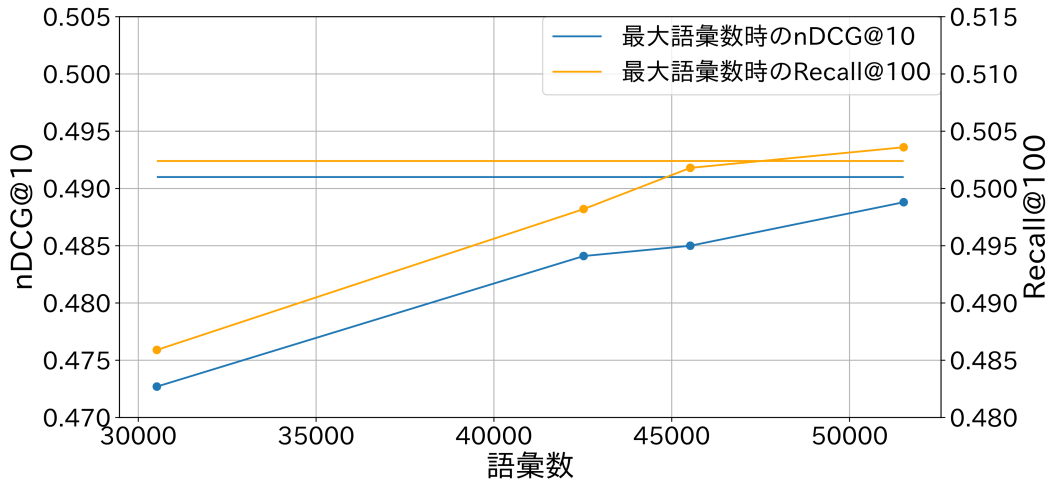


図 4.4: 語彙数による検索精度の変化

が最も変化が大きいため，その変化より十分小さくなる場合を採用した．二つ目の基準として $\epsilon_2 = 0.1$ を用いた．結果的に語彙数は $45,522$ となった．また，三つ目の基準として $\epsilon_2 = 0.05$ を用いた．結果的に語彙数は $51,522$ となった．

各語彙数における，nDCG@10 の結果と Recall@100 の五つのデータの平均を図 4.4に示す．nDCG@10 においては，語彙数が増えるにつれて精度が向上しており，最大語彙数の時に最も高い精度になっている．これはJang et al. (2021) で

より高次元のベクトルにすることで，精度向上が見られることと同様である．一方，Recall@100 は語彙数が 51,522 で最大値を取っており，最大語彙数の場合には精度が下がっている．日本語検索において，再現性の指標においては単語が短くなるように分割を行うことが良いとされている (工藤 2018)．一方，短い単語単位はノイズを増加させることから，適合性の指標においては 1 単語が長くなるように分割することが良いということが推察される．語彙数を増加させることは，1 トークンが長くなるようなトークナイズを行うことに相当すると考えられる．よって，図 4.4に見られる追加語彙数による適合性指標と再現性指標の挙動の違いは，この日本語検索の現象と類似すると考えられる．

4.4.4 教師データがある場合の効果

提案手法は対象データに教師がない場合であった．しかし，教師データが存在する場合もある．その際，AdaLM を行うことで更なる効果が期待できる．なぜならば，Bai et al. (2020) によると，事前学習済み言語モデルを用いてテキストを疎ベクトルに変換する検索アルゴリズムにおいて，同義語や関連語への拡張が精度向上の要素であり，それらは事前学習から得られることが期待できるためである．そのため，教師データが存在する場合に，提案手法の効果があるかを検証する．

検証データとして NFCorpus を用いた．SPLADE の学習において，損失関数は負の対数尤度を用いた．負例には，BM25 の検索結果と二種類の密ベクトル検索の結果を用いた．一つ目は，MS MARCO で学習した密ベクトル検索であり，4.2 節と同様に学習した．二つ目は，BM25 負例と負の対数尤度を用いて，NFCorpus にて一つ目のモデルからさらに学習を行ったものである．負例には上位 100 件のうち正例ではないものを用いた．マージン二乗損失は，この設定よりも精度が悪化したため，使用しなった．そのため，SPLADE の正則化の係数は $\lambda_Q = 0.0006$ と $\lambda_D = 0.0008$ に変更した．これは，Formal et al. (2021) における負の対数尤度を用いる SPLADE-max の設定である．

結果を表 4.10に示す．まず，元々の SPLADE では教師ありの場合でも精度の向上がわずかである．しかし，AdaLM を適用することで，教師ありの精度がさらに向上している．教師ありの場合でも，AdaLM には精度向上効果があることがわかる．これは，Bai et al. (2020) の事前学習済み言語モデルを用いて，テキストを疎ベクトルにエンコードする方法において，同義語・関連語は事前学習で

表 4.10: NFCorpus における教師ありの場合の AdaLM の効果．評価指標は nDCG@10

	NFCorpus
SPLADE	
教師なし	0.336
教師あり	0.339
SPLADE+AdaLM	
教師なし	0.348
教師あり	0.378
SPLADE+AdaLM+IDF 重み	
教師なし	0.353
教師あり	0.377
SPLADE+AdaLM+IDF 重み+BM25 併用	
教師なし	0.356
教師あり	0.381

得られるという考察を支持している．

一方で，IDF 重みや BM25 併用をさらに適用した場合は，教師なしの場合は精度が向上しているが教師ありの場合は，IDF 重みのみでは精度が向上していない．BM25 併用も行った場合も精度向上幅が小さくなっている．対象データにおけるトークンの重要度を用いる手法は，教師ありの場合に精度向上への寄与が小さいことがわかる．

4.4.5 対象データにおけるトークン重要度の使用方法

表 4.5において，IDF 重みと BM25 併用を共に使用した．しかし，BM25 でも IDF が考慮されているため，重複していると考えられる．そこで，BM25 併用を用いた場合に，IDF 重みを使用するか否かの実験を行なった．

結果を表 4.12に示す．SPLADE，SPLADE-Doc 共に平均的な精度は改善している．しかし，SPLADE は TREC-COVID 以外のデータセットでほとんど改善が見られない．一方，SPLADE-Doc では全データセットで精度が改善している．表 4.5より，BM25 併用を用いない場合にも，SPLADE-Doc の方が IDF 重みによる精度改善の幅が大きい．BM25 併用は二つの検索モデルの関連度スコアの足し合わせであるため，一方の検索精度の改善の影響を受けていると考えられる．ま

表 4.11: BM25 併用時の IDF 重みの効果．評価指標は nDCG@10

	バイオ・医療ドメイン			科学ドメイン		
	BASK	NFC	TCOVID	SDOCS	Sfact	平均
SPLADE						
AdaLM+BM25 併用	0.577	0.354	0.727	0.164	0.725	0.509
AdaLM+IDF 重み +BM25 併用	0.577	0.356	0.744	0.165	0.718	0.512
SPLADE-Doc						
AdaLM+BM25 併用	0.551	0.343	0.641	0.157	0.703	0.479
AdaLM+IDF 重み+ +BM25 併用	0.573	0.348	0.653	0.163	0.714	0.490

表 4.12: IDF 重みと BM25 重みの比較．評価指標は nDCG@10

	バイオ・医療ドメイン			科学ドメイン		
	BASK	NFC	TCOVID	SDOCS	Sfact	平均
SPLADE						
AdaLM+IDF 重み	0.544	0.353	0.719	0.161	0.708	0.497
AdaLM+BM25 重み	0.496	0.352	0.680	0.161	0.701	0.478
SPLADE-Doc						
AdaLM+IDF 重み	0.551	0.342	0.633	0.162	0.715	0.481
AdaLM+BM25 重み	0.492	0.341	0.684	0.168	0.715	0.480

た，SPLADE によってエンコードされたベクトルが IDF と同様の情報を持っていると推察される．

4.4.6 SPLADE の重み付け方法の比較

3 章では，IDF よりも BM25 で重みづけた方が良い精度となっていた．そこで，IDF 重みの代わりに BM25 で重みづける BM25 重みの場合と比較する．結果を表 4.12に示す．これより，データセットにより違いはあるが，平均的には IDF 重みの方が高い精度となっていることがわかる．IDF 重みを BM25 重みの差は単語頻度の項の有無にある．単語頻度は 2.1.3 項にて述べた通り，eliteness を表す．このことから，SPLADE でエンコードしたベクトルにおいては，eliteness についての情報が含まれており，単語頻度でさらに重み付けることで，過大な重み付けとなっていると推察される．

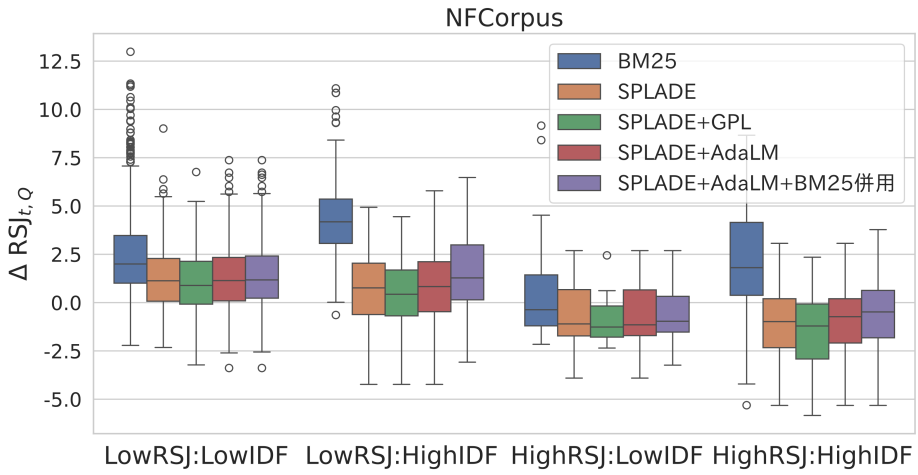


図 4.5: NFCoupus におけるドメイン適応による $\Delta\text{RSJ}_{t,Q}$ の変化．参照のために，BM25 についても $\Delta\text{RSJ}_{t,Q}$ を載せている

4.5　クエリ中のキーワードとの完全一致に関する分析

本節では，事前学習済み言語モデルを用いた検索モデルをドメイン外で用いる場合の課題の一つである，クエリ中のキーワードを含む文書を上位にランニングづけできないという課題が，提案手法によってドメイン適応により改善しているかどうかを分析する．

3.4.3 項と同様に，RSJ 関数を用いて分析を行った．分析のためにトークンの RSJ 重みと IDF の値で区分した．RSJ 関数の値が 75%点より値が大きいトークンを HighRSJ とし，それ以外を LowRSJ とした．また，IDF は中央値より値が大きいトークンを HighIDF とし，それ以外を LowRSJ とした．この分析においてトークナイザには pyserini[12] の analyzer を用いた．このトークナイザは，ポーターステマーによる語幹化と不要語削除を行う．

結果を図 4.5に示す．まず，BM25 に比べて SPLADE の方が $\Delta\text{RSJ}_{t,Q}$ が小さい値になっている．特に，HighIDF のトークンで差が大きい．BM25 の方がキーワード一致による検索を行っていることがわかる．次に，GPL を用いた場合に，SPLADE より $\Delta\text{RSJ}_{t,Q}$ の中央値が小さくなっている．HighRSJ:HighIDF のトークンに注目すると，中央値 $\Delta\text{RSJ}_{t,Q}$ の値が負である．実際，HighRSJ:HighIDF

[12]https://github.com/castorini/pyserini

表 4.13: NFcorpus における HighRSJ:HighIDF であるトークンを含むクエリの一位文書．AdaLM を適用した SPLADE の一位文書は関連文書である．HighRSJ:HighIDF である単語は**太字**にしている

クエリ	**Phytate**s for the Treatment of Cancer
一位文書	
SPLADE+AdaLM	Dietary suppression of colonic cancer. Fiber or **phytate**? The incidence of colonic cancer differs widely ...
SPLADE	Phytochemicals for breast cancer prevention by targeting aromatase. Aromatase is a cytochrome P450 enzyme ...

の平均値は SPLADE が –0.401 であり，GPL を適用した場合，–0.586 であった．RSJ 関数が大きいということは，キーワードとして重要ということであるため，HighRSJ のトークンについて，$\Delta\mathrm{RSJ}_{t,Q}$ が負になることは精度に影響があると考えられる．実際，GPL を適用することによって，NFCorpus は nDCG@10 の値が小さくなっている．よって，GPL はキーワードに関する課題を解決できていない．他方，AdaLM を適用した場合は，SPLADE より $\Delta\mathrm{RSJ}_{t,Q}$ の値が大きくなっている．また，HighRSJ:HighIDF の値も大きくなっている．具体的には，HighRSJ:HighIDF の平均値は，AdaLM を適用した場合，–0.292 であった．よって，AdaLM を適用することで，キーワードを含む文書をより上位にランキングできていることがわかる．さらに，BM25 併用を用いると $\Delta\mathrm{RSJ}_{t,Q}$ の値がさらに大きくなる．そのため，BM25 併用もキーワード一致の課題を解決することがわかる．一方，BM25 併用は LowRSJ:HighIDF のトークンの $\Delta\mathrm{RSJ}_{t,Q}$ も大きくしている．これは，精度の悪化につながる可能性がある．BM25 の関連度スコアと SPLADE の関連度スコアをの和をとることは，アンサンブル手法の一種であるが，今後より良いアンサンブル手法が求められる．

4.6 事例分析

本節では事例分析を行う．まず，事例を用いて実際にキーワード，つまり HighRSJ:HighIDF である単語を含む文書を上位にできているかを確認する．表 4.13にキーワードに HighRSJ:HighIDF な単語を含むクエリと，そのクエリに対する AdaLM を適用した SPLADE の一位文書を示す．ここでのキーワードは phytate である．また，phytate は MS MARCO では下位 2%の頻度でしか出現しておら

表 4.14: NFcorpus おいて AdaLM により SPLADE がドメインの同義語拡張で関連文書の検索順位を上昇させた例．AdaLM を適用した SPLADE の一位文書は関連文書である．

クエリ	amnesia
一位文書	
SPLADE+AdaLM	Amnesic shellfish poisoning (ASP) is caused by ...
SPLADE	The prevalence of dementia is increasing with ...

ず，教師データ中では低頻度である．それぞれの文書を見ると，AdaLM を適用した SPLADE の一位文書は，phytate が含まれており関連文書である．そして，phytate は，AdaLM の語彙が 48522 から 51522 に増加させる際に追加されていた．しかし，SPLADE の一位文書は phytate が含まれておらず，関連文書でもない．また，SPLADE では表 4.1にて述べたように，phytate が語彙に含まれておらず，分割されている．以上より，AdaLM によって事前学習済み言語モデルに語彙が追加されることで，キーワードの分割を防ぎ，キーワードを含む文書を上位にすることができていることがわかる．

さらに，AdaLM により SPLADE がドメインの同義語拡張で関連文書の検索順位を上昇させた例を表 4.14に示す．クエリは amnesia であるが，同義語拡張の結果 amnesic とマッチして検索結果の上位に関連文書をランキングできている．amnesia は元々の BERT のトークナイザの語彙に含まれている．しかし，amnesic は AdaLM の語彙が 42522 から 45522 に増加させる際に追加されている．よって，事前学習済み言語モデルに語彙を追加し，継続事前学習を行うことで，ドメインの同義語拡張を行うことができたと思われる．また，このクエリにおいて SPLADE は密ベクトル検索より nDCG@10 が約 80 ポイント高く，密ベクトル検索では困難なことがわかる．

4.7 複数検索モデルを用いた更なる精度向上

AdaLM は，2.3 節で述べたすべての検索モデルに対して用いることが可能である．本節ではまず，密ベクトル検索と ColBERT に適用しその効果を述べる．次に，BM25 併用は事前学習済み言語モデルを用いた検索モデルとのアンサンブルと考えることが可能である．そのため，BM25 に加えて事前学習済み言語モデル

表 4.15: 密ベクトル検索と ColBERT における AdaLM と GPL の適用結果．評価指標は nDCG@10．同カテゴリで最も良い結果は **太字**にしている

	バイオ・医療ドメイン		科学ドメイン		
	NFC	TCOVID	SDOCS	Sfact	平均
密ベクトル検索					
ドメイン適応なし	0.301	0.716	0.144	0.571	0.433
GPL	0.325	0.723	**0.162**	**0.654**	0.466
AdaLM	0.328	**0.773**	0.148	0.648	**0.475**
AdaLM+GPL	**0.329**	0.733	0.160	0.652	0.469
ColBERT					
ドメイン適応なし	0.327	0.723	0.154	0.645	0.462
GPL	0.327	0.696	0.153	0.668	0.461
AdaLM	**0.332**	**0.768**	0.153	0.677	**0.482**
AdaLM+GPL	0.331	0.744	**0.156**	**0.691**	0.480

を用いた検索モデル同士を用いてアンサンブルを行った場合に更なる精度向上をもたらすかどうかを明らかにする．

4.7.1 密ベクトル検索と ColBERT における AdaLM の効果

4.3.2 項にて，SPLADE に対して AdaLM を適用した結果と GPL を適用した結果を比較した．本項では，事前学習済み言語モデルを用いた他の検索モデルである，密ベクトル検索と ColBERT にて同様の比較を行い，検索モデル毎に差があるかどうかを検証した．なお，COIL は ColBERT の特殊な場合と考えられるため検証から除いた．

結果を表 4.15に示す．どの検索モデルにおいても，AdaLM を適用した場合が平均的にもっとも精度が高い．一方，GPL は密ベクトル検索では精度を向上させたものの，ColBERT を精度の向上させることができていない．よって，平均的には AdaLM の方が，語彙や単語頻度によるドメインシフトに対して，多くの検索モデルで有効であることがわかる．さらに，AdaLM と GPL の双方を適用した場合についても，同様に実験をおこなった．GPL での精度向上が見られる密ベクトル検索においても，4.3.2 項の SPLADE の場合と同様に双方を適用しても AdaLM を適用した場合からの精度の向上はみられなかった．その要因の一つとして，GPL のみ適用した場合には精度が向上していたが，AdaLM に加えて

表 4.16: 複数の検索モデルをアンサンブルした結果．評価指標は nDCG@10．表中で最も良い結果を**太字**にしている

	バイオ・医療ドメイン			科学ドメイン		
	BASK	NFC	TCOVID	SDOCS	SFact	平均
AdaLM を用いない場合						
SPLADE + BM25	0.575	0.344	0.663	0.162	0.716	0.492
+密ベクトル検索	0.548	0.345	0.727	0.169	0.712	0.500
+ ColBERT	0.573	0.341	0.711	0.165	0.715	0.501
+密ベクトル検索 &ColBERT	0.570	0.341	0.743	0.172	0.713	0.508
AdaLM を用いる場合						
SPLADE + BM25	0.577	0.354	0.727	0.164	0.725	0.509
+密ベクトル検索	0.577	**0.360**	0.806	**0.174**	**0.733**	0.530
+ ColBERT	**0.592**	0.358	0.775	0.169	0.723	0.523
+密ベクトル検索 &ColBERT	0.580	**0.360**	**0.817**	0.173	0.729	**0.532**

GPL を適用した場合に精度向上幅が小さくなっていることが挙げられる．例えば，SPLADE においては TREC-COVID で，密ベクトル検索では，NFCourpus，TREC-COVID，Scifact でそのような状況が見られた．そのため，AdaLM と GPL が同じような効果が伺える．このことは，どちらも検索対象文書に関するドメイン適応の手法として解釈可能であることを裏付ける．

一方，AdaLM と GPL で異なる効果を持つことが示唆される場合もある．SCIDOCS において密ベクトル検索や SPLADE では AdaLM ではほとんど精度が向上していないが，GPL では精度が向上している．また，Scifact において ColBERT で AdaLM に加えて GPL を適用することで更に精度が向上している．また，一貫して精度低下をもたらすデータセットと検索モデルの組み合わせもある．これらの現象のより詳細な解明は今後の研究としたい．

4.7.2 複数の検索モデルによるアンサンブル

4.7.1 項にて，AdaLM が密ベクトル検索，SPLADE，ColBERT の全てのモデルで効果があることが明らかになった．BM25 併用は，BM25 併用は事前学習済み言語モデルを用いた検索モデルと BM25 のアンサンブルである．そのため，事前学

習済み言語モデルを用いた検索モデル同士のアンサンブルも精度向上の効果が期待できる．また，AdaLM が複数の検索モデルで効果があるため，AdaLM を適用した場合，アンサンブル時にさらに精度が向上することが期待できる．そのため，AdaLM を行った BERT で学習を行った密ベクトル検索，SPLADE，ColBERT と BM25 でアンサンブルを行った．アンサンブルの方法としては，BM25 併用と同様に上位 100 件の文書のスコアの和を用いた．なお，他の検索モデルで現れない文書のスコアについては，最低スコアとの和を取った．結果を表 4.16に示す．

密ベクトル検索，ColBERT のどちらを足し合わせた場合も，精度が向上している．さらに，どちらも足し合わせた場合にも，平均的にはさらに精度が向上している．ただし，データセットによって最良の場合は変化している．潜在的には全てアンサンブルした場合に，各データセットの最高精度の結果が得られるため，単純な和をとるよりも良いアンサンブル方法が求められる．

次に，AdaLM を用いずに学習した検索モデルでアンサンブルを行った場合の結果と比較するこちらも同様にアンサンブルによって精度が向上している．ただし，全ての場合について AdaLM を用いた場合の方が精度が高い．よって，BM25 のみならず他の検索モデルとのアンサンブルする場合でも AdaLM は効果的であると言える．

4.8 まとめ

本章では，教師なしで専門的なドメインのような語彙や単語頻度がソースドメインと大きく異なるドメインに対して精度を向上させる手法として，事前学習済み言語モデルに対する教師なしドメイン適応を行う，AdaLM を SPLADE に用いることを提案した．実験を通じて，代表的な既存のドメイン適応手法である GPL よりも，AdaLM を用いる方が当該のドメインシフトに対して有効であることを示した．さらに，SPLADE において AdaLM に加えて対象データにおけるトークンの重要度を用いるドメイン適応手法を用いるとさらに精度が向上することを示した．一方で，AdaLM と GPL を共に用いた場合に平均的には精度が向上しないことを明らかにした．その一因として，双方が対象文書のドメインシフトに対して有効なドメイン適応手法であり，同様の効果を持つ場合があるため，共に用いても精度向上が見られなくなることを考察した．

事前学習済み言語モデルに対するドメイン適応手法間での比較として，語彙を追加しない場合と対象ドメインのコーパスで事前学習する場合を AdaLM と比較した．語彙を追加しない場合と比較し，AdaLM は大きく精度を向上させていることから語彙追加を事前学習前に行うことが検索においても効果的であることを明らかにした．SPLADE はクエリ拡張と文書拡張を行う方法であると考えらえる．そのため，語彙追加が事前学習を通じてより精度の高い同義語や関連語の拡張を促進していることが示唆される．また，対象ドメインのコーパスで事前学習する場合と AdaLM の比較した場合，対象ドメインのコーパスのみで事前学習する場合，MS MARCO を教師データとして SPLADE を学習することに失敗する場合が見られた．このことから，AdaLM を用いる方がより安定的に SPLADE を学習できると考えらえる．一方で，対象ドメインとソースドメインで語彙や単語頻度の変化が小さい場合についても実験を行い，SPLADE からの精度向上幅が小さいことを明らかにした．AdaLM は語彙や単語頻度がソースドメインと大きく異なる場合に有効と考えられる．

AdaLM によるドメイン適用の効果をさらに調べるため，追加する語彙数による精度の変化，対象ドメインに教師データが存在する場合の精度の差，対象ドメインとソースドメインで語彙や単語頻度の変化が小さい場合について検証を行った．まず，追加する語彙数による精度の変化であるが，適合性の指標においては，今回の実験では語彙を追加するほど高い値を示した．一方で，再現性の指標においては，ある一定の値を超えたところで語彙を追加すると精度の低下が見られた．この点について，適合性指標と再現性指標の挙動の違いは，この日本語検索におけるトークナイズ単位との関連を述べた．対象ドメインに教師データが存在する場合については，AdaLM を用いることで SPLADE に更なる精度向上が見られた．疎ベクトルを用いる検索モデルの場合，同義語や関連語に関しては事前学習によって得られるという主張 (Bai et al. 2020) をサポートするものである．最後に，対象ドメインとソースドメインで語彙や単語頻度の変化が小さい場合は，AdaLM は精度改善にほとんど寄与しないことを確かめた．

さらに，本研究のアプローチでクエリ中のキーワードを含む文書が上位にランキングできているかを，RSJ 関数を用いて分析した．提案手法によりクエリと関連文書におけるトークン一致の問題が解決していること確認した．また，事例を通じてトークン一致の問題が実際に解決することを見た．さらに，事例から AdaLM

を適用することにより，ドメインの同義語を獲得していることを確認した．

最後に，AdaLM は事前学習済み言語モデルを用いる全ての検索モデルで使用可能なことから，事前学習済み言語モデルを用いる複数の検索モデルに対して AdaLM を適用し関連度スコアの和をとるアンサンブルを行った．結果，BM25 も含めアンサンブルを行うことで，さらに精度が向上することを示した．また，AdaLM がアンサンブル時にも有効なドメイン適応手法であることを示した．

第 5 章

結論

本論文では，事前学習済み言語モデルを用いた検索モデルをゼロショットで精度向上させることを目的に，教師なしドメイン適応する手法を二つ提案した．いずれの手法も，ゼロショット検索の精度低下要因の一つである，クエリ中のキーワードが完全一致する文書を上位にできないという課題を解決するものである．また，いずれの手法も検索対象文書のドメインシフトに有効な手法であることを実験的に示した．

一つ目の手法は，対象データのトークン重要度を用いる方法である．その際，密ベクトル検索がもつ文脈を考慮したスコアリングが可能な方法として，C-BM25 を提案した．C-BM25 は，クエリと文書で一致したトークン毎に文脈の類似度計算し，同じ語をもつトークンのうち最大値を BM25 で重み付けた和を関連度スコアとした．文脈類似度には，トークンの文脈を表現するベクトルを用いた．また，文脈を表現するベクトルは，クエリや文書を密ベクトル検索モデルによってエンコードすることで得た．実験を通じて，C-BM25 が BM25 や密ベクトル検索から平均的には精度が向上することを明らかにした．さらに，密ベクトル検索の関連度スコアと C-BM25 の関連度スコアの和を用いる C-BM25 併用が，ソースドメイン外でも高精度な検索モデルや既存の対象におけるトークンの重要度を用いるドメイン適応である BM25 併用と比較し，高い精度を示すことを明らかにした．さらに，応答速度の検証を行い，実用上十分高速に動作することを示した．

二つ目の手法は，事前学習済み言語モデルに語彙追加を行う方法である．教師なしで精度を向上させる要求が強い，専門的なドメインでは，検索モデルを学習したドメインから語彙や単語頻度が大きく変化することが多い．．そのため，語

彙や単語頻度に関するドメインシフトに対するドメイン適応手法として AdaLM を用いることを提案した．AdaLM は，BERT に対するドメイン適応手法であり，対象ドメインのコーパスを用いて，BERT に対して語彙の追加と継続事前学習を行う．AdaLM は，語彙や単語頻度に関するドメインシフトの大きい対象ドメインに対して，既存の教師なしドメイン適応手法である GPL を上回る精度を示した．さらに，対象データのトークン重要度を併せて用いる方法と相補的であることを示した．AdaLM は事前学習済み言語モデルを用いる全ての検索モデルで使用可能なことから，複数の検索モデルに対して AdaLM を適用し関連度スコアの和をとるアンサンブルを行った．結果，BM25 も含めアンサンブルを行うことで，さらに精度が向上することを示した．

以下，今後の展望について述べる．まず，本研究の課題であるが，C-BM25 は，現状インデックス時の使用メモリが非常に多い点がある．また，複数の検索モデルでアンサンブルを行う場合も使用メモリが増える．今後の研究として，同程度の精度を保ちながら，インデックス時のメモリを減らす方法が求められる．また，本研究で用いたアンサンブルは単純な和をとるというものであった．今後の研究として，さらに高度なアンサンブル方法またはモデル統合方法が求められる．本研究の提案手法は，主に検索対象文書のドメインシフトに対する有効なドメイン適応手法であった．しかし，ドメインシフトは検索対象文書に限られない．2.5.5 項で述べた通り，クエリ及び関連度に関するドメインシフトへの適応手法もすでに提案されている．今後の研究の一つとして，どちらのドメインシフトに対しても効果的な手法が求められる．

検索における未解決の課題として，クエリと文書におけるトークンの一致がノイズとなるクエリに対して，関連文書を上位にすることが挙げられる．本研究からも明らかなように，基本的にはトークンの一致は検索において重要である．一方で，例えば，一般にキーワードと考えられる固有名詞が，単に状況説明のために入力されており，ユーザの意図としてはキーワードではないが，検索システムにはキーワードとして判定される場合がある．この様な場合でも，精度の高い検索方法が求められる．

最後に，大規模言語モデルによって検索は大きく変化している．大規模言語モデルを用いることで，モデル内の知識のみで回答可能なことが増えたためである．また，大規模言語モデルは，より指示に従った応答を返すことが可能である．

しかしながら，検索の役割は依然として大きい．大規模言語モデルは，パラメータ更新に多大なコストがかかるため，知識の更新が困難である．そのため，検索結果を大規模言語モデルに入力して知識を補う試みが行われている (Lewis et al. 2020)．特に，Mallen et al. (2023) は，珍しいエンティティーがクエリに含まれるものは，検索と併せて用いた方が良いことを示した．また，Jiang et al. (2023) は，生成時に尤度の低い文に対して検索を用いることで，回答の精度が上がることを示している．今後，Nakano et al. (2021) やLiu et al. (2023) など，探査やクエリの書き換えも含めたより高度な検索及び大規模言語モデルの活用が研究されると思われる．

業績リスト

ジャーナル論文

1. 飯田 大貴, 岡崎 直観. Zero-shot ニューラル検索のための語彙一致と文脈の類似度による関連度スコアリング. 情報処理学会論文誌データベース (TOD), 15(3):20–35 (ダブルカラム, 16 ページ）, 2022 年 10 月.

国際学会発表

1. Hiroki Iida and Naoaki Okazaki. Unsupervised Domain Adaptation for Sparse Retrieval by Filling Vocabulary andWord Frequency Gaps. In Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), AACL-IJCNLP, pages 752–765 (double column, 14page), Novermber 2022.

2. Hiroki Iida, Naoaki Okazaki. Incorporating Semantic Textual Similarity and Lexical Matching for Information Retrieval In Proceedings of the 35th Pacific Asia Conference on Language, Information and Computation, PACLIC, pages 582–591, Shanghai, China. Association for Computational Lingustics 2021.

国内口頭発表

1. 飯田 大貴, 岡崎 直観. 事前学習済みモデルに基づく検索モデルにおけるドメイン適応手法の比較と相乗効果の検証 言語処理学会第 29 回年次大会 (NLP2023), pp. 176-181, 2023 年 3 月.

参考文献

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih (2023) "Task-aware Retrieval with Instructions," in *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 3650–3675, Toronto, Canada: Association for Computational Linguistics, July, DOI: 10.18653/v1/2023.findings-acl.225.

Yang Bai, Xiaoguang Li, Gang Wang et al. (2020) "SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval," October.

Iz Beltagy, Kyle Lo, and Arman Cohan (2019) "SciBERT: A Pretrained Language Model for Scientific Text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3615–3620, Hong Kong, China: Association for Computational Linguistics, November, DOI: 10.18653/v1/D19-1371.

John Blitzer, Mark Dredze, and Fernando Pereira (2007) "Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification," in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 440–447, Prague, Czech Republic: Association for Computational Linguistics, June.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif et al. (2020) "Overview of Touché 2020: Argument Retrieval," in *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pp. 384–395.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler (2016) “A Full-Text Learning to Rank Dataset for Medical Information Retrieval,” in *Advances in Information Retrieval*, pp. 716–722.

Stefan Büttcher・Charles L A Clarke・Gordon V Cormack 他 (2020) 「情報検索 : 検索エンジンの実装と評価」.

ZeFeng Cai, Chongyang Tao, Tao Shen, Can Xu, Xiubo Geng, Xin Alex Lin, Liang He, and Daxin Jiang (2023) “HypeR: Multitask Hyper-Prompted Training Enables Large-Scale Retrieval Generalization,” in *The Eleventh International Conference on Learning Representations*.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld (2020) “SPECTER: Document-level Representation Learning using Citation-informed Transformers,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2270–2282, Online, July.

Zhuyun Dai and Jamie Callan (2019) “Deeper Text Understanding for IR with Contextual Neural Language Modeling,” in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR’19, p. 985–988, New York, NY, USA: Association for Computing Machinery, DOI: 10.1145/3331184.3331303.

Zhuyun Dai and Jamie Callan (2020) “Context-Aware Term Weighting For First Stage Passage Retrieval,” in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’20, pp. 1533–1536, New York, NY, USA: Association for Computing Machinery, July.

Zhuyun Dai, Vincent Y Zhao, Ji Ma et al. (2023) “Promptagator: Few-shot Dense Retrieval From 8 Examples,” in *The Eleventh International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019) “BERT: Pre-training of Deep Bidirectional Transformers for Language Un-

derstanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota: Association for Computational Linguistics, June, DOI: 10.18653/v1/N19-1423.

Yixing Fan, Jiafeng Guo, Xinyu Ma, Ruqing Zhang, Yanyan Lan, and Xueqi Cheng (2021) "A Linguistic Study on Relevance Modeling in Information Retrieval," in *Proceedings of the Web Conference 2021*, WWW '21, pp. 1053–1064, New York, NY, USA: Association for Computing Machinery, April.

Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn (2021) "Efficiently Identifying Task Groupings for Multi-Task Learning," in A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan eds. *Advances in Neural Information Processing Systems*.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant (2021) "SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval," September.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant (2022a) "Match Your Words! A Study of Lexical Matching in Neural Information Retrieval," in Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty eds. *Advances in Information Retrieval*, pp. 120–127, Cham: Springer International Publishing.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant (2022b) "From Distillation to Hard Negative Sampling: Making Sparse Neural IR Models More Effective," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pp. 2353–2359, New York, NY, USA: Association for Computing Machinery, July.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo

Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky (2017) "Domain-adversarial training of neural networks," in *Domain Adaptation in Computer Vision Applications*, pp. 189–209, Cham: Springer International Publishing.

Luyu Gao and Jamie Callan (2021) "Condenser: a Pre-training Architecture for Dense Retrieval," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 981–993, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, November, DOI: 10.18653/v1/2021.emnlp-main.75.

Luyu Gao and Jamie Callan (2022) "Unsupervised Corpus Aware Language Model Pre-training for Dense Passage Retrieval," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2843–2853, Dublin, Ireland: Association for Computational Linguistics, May, DOI: 10.18653/v1/2022.acl-long.203.

Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan (2021a) "Complement Lexical Retrieval Model with Semantic Residual Embeddings," in *Advances in Information Retrieval*, pp. 146–160: Springer International Publishing.

Luyu Gao, Zhuyun Dai, and Jamie Callan (2021b) "COIL: Revisit Exact Lexical Match in Information Retrieval with Contextualized Inverted List," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3030–3042, Online: Association for Computational Linguistics, June.

Tianyu Gao, Xingcheng Yao, and Danqi Chen (2021) "SimCSE: Simple Contrastive Learning of Sentence Embeddings," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6894–6910, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, November.

Yu Gu, Robert Tinn, Hao Cheng et al. (2021) "Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing," *ACM Trans. Comput. Healthcare*, Vol. 3, No. 1, October, DOI: 10.1145/3458754.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft (2016) "A Deep Relevance Matching Model for Ad-hoc Retrieval," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, Vol. 24-28-Octo of CIKM '16, pp. 55–64, New York, NY, USA: Association for Computing Machinery, October.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith (2020) "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360, Online: Association for Computational Linguistics, July, DOI: 10.18653/v1/2020.acl-main.740.

Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork (2020) "Learning-to-Rank with BERT in TF-Ranking."

Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan (2017) "DBpedia-Entity v2: A Test Collection for Entity Search," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pp. 1265–1268, August.

John Hewitt and Christopher D Manning (2019) "A Structural Probe for Finding Syntax in Word Representations," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4129–4138, Minneapolis, Minnesota: Association for Computational Linguistics, June.

Sebastian Hofstätter and Allan Hanbury (2019) "Let's measure run time! Extending the IR replicability infrastructure to include performance aspects," in *OSIRRC@SIGIR*.

Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury (2020) "Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation," October.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury (2021) "Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 113–122, New York, NY, USA: Association for Computing Machinery.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave (2022) "Unsupervised Dense Information Retrieval with Contrastive Learning," *Transactions on Machine Learning Research*.

Kyoung-Rok Jang, Junmo Kang, Giwon Hong, Sung-Hyon Myaeng, Joohee Park, Taewon Yoon, and Heecheol Seo (2021) "Ultra-High Dimensional Sparse Representations with Binarization for Efficient Text Retrieval," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1016–1029, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, November, DOI: 10.18653/v1/2021.emnlp-main.78.

Kalervo Järvelin and Jaana Kekäläinen (2002) "Cumulated Gain-Based Evaluation of IR Techniques," *ACM Trans. Inf. Syst.*, Vol. 20, No. 4, p. 422–446, October, DOI: 10.1145/582415.582418.

Zhengbao Jiang, Frank F Xu, Luyu Gao et al. (2023) "Active Retrieval Augmented Generation," May.

Constantinos Karouzos, Georgios Paraskevopoulos, and Alexandros Potamianos (2021) "UDALM: Unsupervised Domain Adaptation through Language Modeling," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2579–2590, Online: Association for Computational Linguistics, June, DOI: 10.18653/v1/2021.naacl-main.203.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-Tau Yih (2020) "Dense Passage Retrieval for Open-Domain Question Answering," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online: Association for Computational Linguistics, November.

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu (2023) "Continual Pre-training of Language Models," in *The Eleventh International Conference on Learning Representations*.

Omar Khattab and Matei Zaharia (2020) "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pp. 39–48, New York, NY, USA: Association for Computing Machinery, July.

Weize Kong, Jeffrey M Dudek, Cheng Li, Mingyang Zhang, and Michael Bendersky (2023) "SparseEmbed: Learning Sparse Lexical Representations with Contextual Embeddings for Retrieval," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, pp. 2399–2403, New York, NY, USA: Association for Computing Machinery, July.

Wouter M Kouw and Marco Loog (2018) "An introduction to domain adaptation and transfer learning," December.

Taku Kudo (2018) “Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 66–75, Melbourne, Australia: Association for Computational Linguistics, July, DOI: 10.18653/v1/P18-1007.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield et al. (2019) “Natural Questions: A Benchmark for Question Answering Research,” *Transactions of the Association for Computational Linguistics*, Vol. 7, pp. 452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova (2019) “Latent Retrieval for Weakly Supervised Open Domain Question Answering,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6086–6096, Florence, Italy: Association for Computational Linguistics, July.

Markus Leippold and Thomas Diggelmann (2020) “Climate-FEVER: A Dataset for Verification of Real-World Climate Claims,” in *NeurIPS 2020 Workshop on Tackling Climate Change with Machine Learning*, December.

Omer Levy and Yoav Goldberg (2014) “Linguistic Regularities in Sparse and Explicit Word Representations,” in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pp. 171–180, Stroudsburg, PA, USA: Association for Computational Linguistics.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li (2004) “RCV1: A New Benchmark Collection for Text Categorization Research,” *J. Mach. Learn. Res.*, Vol. 5, p. 361–397, December.

Patrick Lewis, Ethan Perez, Aleksandra Piktus et al. (2020) “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS’20, Red Hook, NY, USA: Curran Associates Inc.

Hang Li and Jun Xu (2014) “Semantic Matching in Search,” *Foundations and Trends in Information Retrieval*, Vol. 7, No. 5, pp. 343–469.

Xiang Lisa Li and Percy Liang (2021) "Prefix-Tuning: Optimizing Continuous Prompts for Generation," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, Online: Association for Computational Linguistics, August.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang (2023) "Towards General Text Embeddings with Multi-stage Contrastive Learning," *arXiv [cs.CL]*, August.

Jimmy Lin and Xueguang Ma (2021) "A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques," June.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates (2020) "Pretrained Transformers for Text Ranking: BERT and Beyond," October.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira (2021) "Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, p. 2356–2362, New York, NY, USA: Association for Computing Machinery, DOI: 10.1145/3404835.3463238.

Xiao Liu, Hanyu Lai, Hao Yu et al. (2023) "WebGLM: Towards An Efficient Web-Enhanced Question Answering System with Human Preferences," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, p. 4549–4560, New York, NY, USA: Association for Computing Machinery, DOI: 10.1145/3580305.3599931.

Yinhan Liu, Myle Ott, Naman Goyal et al. (2019) "RoBERTa: A Robustly Optimized BERT Pretraining Approach."

Zheng Liu, Shitao Xiao, Yingxia Shao, and Zhao Cao (2023) "RetroMAE-2: Duplex Masked Auto-Encoder For Pre-Training Retrieval-Oriented Language Models," in *Proceedings of the 61st Annual Meeting of the Associ-*

ation for Computational Linguistics (Volume 1: Long Papers), pp. 2635–2648, Toronto, Canada: Association for Computational Linguistics, July, DOI: 10.18653/v1/2023.acl-long.148.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld (2020) "S2ORC: The Semantic Scholar Open Research Corpus," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4969–4983, Online: Association for Computational Linguistics, July, DOI: 10.18653/v1/2020.acl-main.447.

Shuqi Lu, Di He, Chenyan Xiong et al. (2021) "Less is More: Pretrain a Strong Siamese Encoder for Dense Text Retrieval Using a Weak Decoder," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 2780–2791, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, November, DOI: 10.18653/v1/2021.emnlp-main.220.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins (2021) "Sparse, Dense, and Attentional Representations for Text Retrieval," *Transactions of the Association for Computational Linguistics*, Vol. 9, pp. 329–345.

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald (2021) "Zero-shot Neural Passage Retrieval via Domain-targeted Synthetic Question Generation," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 1075–1088, Online: Association for Computational Linguistics, April.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian (2019) "CEDR: Contextualized Embeddings for Document Ranking," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, p. 1101–1104, New York, NY, USA: Association for Computing Machinery, DOI: 10.1145/3331184.3331317.

Sean MacAvaney, Arman Cohan, and Nazli Goharian (2020) "SLEDGE-Z: A

Zero-Shot Baseline for COVID-19 Literature Search," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4171–4179, Online: Association for Computational Linguistics, November.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur (2018) "WWW'18 Open Challenge: Financial Opinion Mining and Question Answering," in *Companion Proceedings of the The Web Conference 2018*, WWW '18, pp. 1941–1942, Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, April.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi (2023) "When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9802–9822, Toronto, Canada: Association for Computational Linguistics, July, DOI: 10.18653/v1/2023.acl-long.546.

Yusuke Matsui, Yusuke Uchida, and Shin'ichi Satoh (2018) "A survey of product quantization," *ITE Transactions on Media Technology and Applications*, Vol. 6, No. 1, pp. 2–10.

Schuster Mike and Nakajima Kaisuke (2012) "Japanese and Korean voice search," *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5149–5152.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013) "Distributed Representations of Words and Phrases and their Compositionality," in C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger eds. *Advances in Neural Information Processing Systems*, Vol. 26: Curran Associates, Inc.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers (2023)

"MTEB: Massive Text Embedding Benchmark," in *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2014–2037, Dubrovnik, Croatia: Association for Computational Linguistics, May.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji et al. (2021) "WebGPT: Browser-assisted question-answering with human feedback," December.

Sungjin Nam, David Jurgens, and Kevyn Collins-Thompson (2022) "An Attention-Based Model for Predicting Contextual Informativeness and Curriculum Learning Applications," *ArXiv*, Vol. abs/2204.09885.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng (2016) "MS MARCO: A Human Generated MAchine Reading COmprehension Dataset," in *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, Vol. 1773 of CEUR Workshop Proceedings.

Jakob Nielsen (1993) "Response Times: The 3 Important Limits," https://www.nngroup.com/articles/response-times-3-important-limits/, Accessed: 2023-09-10.

Rodrigo Nogueira and Kyunghyun Cho (2019) "Passage Re-ranking with BERT," January.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho (2019) "Document Expansion by Query Prediction," *arXiv Preprint*, Vol. arXiv:1904.08375.

Biswajit Paria, Chih-Kuan Yeh, Ian E.H. Yen, Ning Xu, Pradeep Ravikumar, and Barnabás Póczos (2020) "Minimizing FLOPs to Learn Efficient Sparse Representations," in *International Conference on Learning Representations*.

Adam Paszke, Sam Gross, Francisco Massa et al. (2019) *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, Red Hook, NY, USA: Curran Associates Inc.

Fabio Petroni, Aleksandra Piktus, Angela Fan et al. (2021) "KILT: a Benchmark for Knowledge Intensive Language Tasks," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2523–2544, Online: Association for Computational Linguistics, June.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu (2019) "Understanding the Behaviors of BERT in Ranking."

Ori Ram, Liat Bezalel, Adi Zicher, Yonatan Belinkov, Jonathan Berant, and Amir Globerson (2023) "What Are You Token About? Dense Retrieval as Distributions Over the Vocabulary," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2481–2498, Toronto, Canada: Association for Computational Linguistics, July, DOI: 10.18653/v1/2023.acl-long.140.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen (2021) "RocketQAv2: A Joint Training Method for Dense Passage Retrieval and Passage Re-ranking," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 2825–2835, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, November, DOI: 10.18653/v1/2021.emnlp-main.224.

S E Robertson and S Walker (1994) "Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval," in *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '94, pp. 232–241, Berlin, Heidelberg: Springer-Verlag, August.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia (2022) "ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3715–3734, Seattle, United States: Association for Computational Linguistics, July, DOI: 10.18653/v1/2022.naacl-main.272.

Sarah Schuster, Stefan Hawelka, Florian Hutzler, Martin Kronbichler, and Fabio Richlan (2016) "Words in Context: The Effects of Length, Frequency, and Predictability on Brain Responses During Natural Reading," *Cereb. Cortex*, Vol. 26, No. 10, pp. 3889–3904, October.

Rico Sennrich, Barry Haddow, and Alexandra Birch (2016) "Neural Machine Translation of Rare Words with Subword Units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany: Association for Computational Linguistics, August, DOI: 10.18653/v1/P16-1162.

Tao Shen, Xiubo Geng, Chongyang Tao, Can Xu, Xiaolong Huang, Binxing Jiao, Linjun Yang, and Daxin Jiang (2023) "LexMAE: Lexicon-Bottlenecked Pretraining for Large-Scale Retrieval," in *The Eleventh International Conference on Learning Representations*.

Hidetoshi Shimodaira (2000) "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of Statistical Planning and Inference*, Vol. 90, No. 2, pp. 227–244, DOI: https://doi.org/10.1016/S0378-3758(00)00115-4.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu (2020) "MPNet: Masked and Permuted Pre-Training for Language Understanding," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA: Curran Associates Inc.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014) "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.*, Vol. 15, pp. 1929–1958.

Hongjin Su, Weijia Shi, Jungo Kasai et al. (2023) "One Embedder, Any Task: Instruction-Finetuned Text Embeddings," in *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 1102–1121, Toronto, Canada: Association for Computational Linguistics, July, DOI: 10.18653/v1/2023.findings-acl.71.

Weng Lam Tam, Xiao Liu, Kaixuan Ji et al. (2022) "Parameter-Efficient Prompt Tuning Makes Generalized and Calibrated Neural Text Retrievers," July.

Ian Tenney, Dipanjan Das, and Ellie Pavlick (2019) "BERT Rediscovers the Classical NLP Pipeline," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, Florence, Italy: Association for Computational Linguistics, July.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych (2021) "BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal (2018) "FEVER: a Large-scale Dataset for Fact Extraction and VERification," pp. 809–819, June.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin (2017) "Attention is All you Need," in I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett eds. *Advances in Neural Information Processing Systems*, Vol. 30: Curran Associates, Inc.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick et al. (2021) "TREC-COVID:

constructing a pandemic information retrieval test collection," *SIGIR Forum*, Vol. 54, No. 1, pp. 1–12, February.

Ellen M Voorhees (2004) "Overview of the TREC 2004 robust retrieval track," in *TREC*.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein (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)*, pp. 241–251, Melbourne, Australia, July.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi (2020) "Fact or Fiction: Verifying Scientific Claims," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7534–7550, Online, November.

Kexin Wang, Nils Reimers, and Iryna Gurevych (2021) "TSDAE: Using Transformer-based Sequential Denoising Auto-Encoderfor Unsupervised Sentence Embedding Learning," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 671–688, Punta Cana, Dominican Republic: Association for Computational Linguistics, November, DOI: 10.18653/v1/2021.findings-emnlp.59.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych (2022) "GPL: Generative Pseudo Labeling for Unsupervised Domain Adaptation of Dense Retrieval," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2345–2360, Seattle, United States: Association for Computational Linguistics, July, DOI: 10.18653/v1/2022.naacl-main.168.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei (2022) "Text Embeddings by Weakly-Supervised Contrastive Pre-training," *arXiv preprint arXiv:2212.03533*.

Jason Wise (2023) “Read more at EarthWeb: How Many Google Searches per Day in 2023? (Full Statistics),” https://earthweb.com/how-many-google-searches-per-day/ Accessed: 2023-09-10.

Shitao Xiao, Zheng Liu, Yingxia Shao, and Zhao Cao (2022) “RetroMAE: Pre-Training Retrieval-oriented Language Models Via Masked Auto-Encoder,” in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 538–548, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, December, DOI: 10.18653/v1/2022.emnlp-main.35.

Ji Xin, Chenyan Xiong, Ashwin Srinivasan, Ankita Sharma, Damien Jose, and Paul Bennett (2022) “Zero-Shot Dense Retrieval with Momentum Adversarial Domain Invariant Representations,” in *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 4008–4020, Dublin, Ireland: Association for Computational Linguistics, May, DOI: 10.18653/v1/2022.findings-acl.316.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk (2021) “Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval,” in *International Conference on Learning Representations*.

Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley (2022) “LaPraDoR: Unsupervised Pretrained Dense Retriever for Zero-Shot Text Retrieval,” in *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 3557–3569, Dublin, Ireland: Association for Computational Linguistics, May.

Wei Yang, Haotian Zhang, and Jimmy Lin (2019) “Simple Applications of BERT for Ad Hoc Document Retrieval,” March.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning (2018) “HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering,” in *Proceedings of the*

2018 Conference on Empirical Methods in Natural Language Processing, pp. 2369–2380.

Yunzhi Yao, Shaohan Huang, Wenhui Wang, Li Dong, and Furu Wei (2021) "Adapt-and-Distill: Developing Small, Fast and Effective Pretrained Language Models for Domains," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 460–470, Online: Association for Computational Linguistics, August, DOI: 10.18653/v1/2021.findings-acl.40.

Yue Yu, Chenyan Xiong, Si Sun, Chao Zhang, and Arnold Overwijk (2022) "COCO-DR: Combating Distribution Shift in Zero-Shot Dense Retrieval with Contrastive and Distributionally Robust Learning," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 1462–1479, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, December, DOI: 10.18653/v1/2022.emnlp-main.95.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma (2021) "Optimizing Dense Retrieval Model Training with Hard Negatives," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1503–1512, New York, NY, USA: Association for Computing Machinery.

Rong Zhang, Revanth Gangi Reddy, Md Arafat Sultan et al. (2020) "Multi-Stage Pre-training for Low-Resource Domain Adaptation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5461–5468, Online: Association for Computational Linguistics, November, DOI: 10.18653/v1/2020.emnlp-main.440.

岡崎直観・荒瀬由紀・鈴木潤・鶴岡慶雅・宮尾祐介 (2022) 『IT Text 自然言語処理の基礎』，オーム社.

岡谷貴之 (2022) 『深層学習』，MLP 機械学習プロフェッショナルシリーズ，講談社，第改訂第 2 版版.

工藤拓 (2018) 『形態素解析の理論と実装』，実践・自然言語処理シリーズ ; 第 2 巻，近代科学社.

酒井哲也 (2015) 『情報アクセス評価方法論 : 検索エンジンの進歩のために』，コロナ社.

瀬々潤・浜田道昭 (2015) 『生命情報処理における機械学習 : 多重検定と推定量設計 = Machine learning in bioinformatics』，MLP 機械学習プロフェッショナルシリーズ，講談社.