事前学習言語モデルに基づく質問応答のプログラミング言語リファレンスへの適用

仲地 優登† 中野 優†† 加藤 誠†††

† 筑波大学 知識情報・図書館学類 〒 305-–8550 茨城県つくば市春日 1-2
†† 筑波大学大学院 人間総合科学学術院 〒 305–8550 茨城県つくば市春日 1–2
††† 筑波大学 図書館情報メディア系 〒 305–8550 茨城県つくば市春日 1-2
E-mail: †nakachi@klis.tsukuba.ac.jp, ††{yunakano,mpkato}@acm.org

あらまし 本論文では，与えられた質問に対してプログラミング言語リファレンスを検索し，取得された文書から質問に対する回答を抽出するような，プログラミング言語リファレンスに対する質問応答の提案を行う．質問応答の手法としては，事前学習言語モデルに基づいた，Retriever と Reader を用いる方法を採用する．この質問応答手法をプログラミング言語リファレンスへ適用するにあたって，文書構造を考慮したパッセージ分割，および，半自動的に生成された質問-適合パッセージ-回答の組によるドメイン適応を提案し，これらの工夫による精度向上の有無について評価を行った．実験では，プログラミング言語リファレンスで回答可能な質問を作成し，提案手法の評価を行なった．

キーワード 質問応答，ドメイン指向アプリケーション，自然言語処理応用，プログラミング支援，ExtractiveQA

1 はじめに

情報技術の発展に従い, 情報技術人材の育成需要は増大している．情報技術人材育成の中で，特にプログラミング学習は重要視されている．代表的な例として，文部科学省は 2017 年に小学校学習指導要領 [1]の改訂を公示し, 情報活用能力を言語能力と同様に「学習の基盤となる資質・能力」として位置付け, 児童がプログラミング学習活動を行うこと義務づけた．プログラミング学習を重要視する動きに伴い, プログラミングを学び始めたばかりの人であるプログラミング初学者は増加している．プログラミング学習支援として, stackoverflow や teratail などの QA サイトの活用がある．QA サイトはユーザが質問を投稿し，他のユーザから回答が得られるインターネットサービスのことである．プログラミング初学者は, プログラミング学習における質問に回答してもらうことを期待して QA サイトに質問を投稿し, 他のユーザから回答を得る．

しかしながら, QA サイトではプログラミングの専門的知識を持ったユーザに回答を頼っている．そのため，回答が得られるまでに時間的コストと人的コストがかかり，実際にプログラミング学習における質問に対して迅速に，適切な回答が得られるかどうかは他のユーザ次第である．また, Stack Overflow が 2019 年に公開した “The Loop #2: Understanding Site Satisfaction, Summer 2019” [2]において，Stack Overflow のユーザコミュニティが非歓迎的であるという項目の不満意見が他の項目と比較して最も多く，これは，プログラミング初学者が QA サイトでコミュニティに加わりにくい要因の一つとして考えられる．

本研究では，与えられた質問に対してプログラミング言語リファレンスを検索し，取得された文書から質問に対する回答を抽出するような質問応答の提案を行う．これにより，プログラミング初学者が他のユーザからの影響を受けることなく，プログラミングに対する質問を解決することを目指す．ここでいうプログラミング言語リファレンスとは，プログラミング言語の構文やセマンティクス，プログラミング言語と共に配付されている標準ライブラリなどの説明が記載されている文書のことである．質問応答に用いる手法として，事前学習言語モデルに基づいた，*Retriever* と *Reader* を用いる方法を採用する．これは Karpukhin らのモデル [10] と同様のものである．予備実験の結果，事前学習言語モデルに「パッセージ分割によって構造依存の暗黙的な意味を捉えきれない」,「ドメイン特有の知識を持っていない」という 2 つの課題があることがわかった．これらの課題に対して 2 つの提案を行った．1 つ目は，構造依存パッセージ分割と構造依存パッセージ分割拡張の 2 つのパッセージ分割手法である．2 つ目は動詞に着目したテンプレートから作成した学習事例を用いたドメイン適応手法である．構造依存パッセージ分割は，文書を用語とその説明文ごとに分割する分割器によってセグメントに分割し，セグメントに対して特定の単語数でパッセージに分割を行うパッセージ分割手法である．構造依存パッセージ分割拡張は，構造依存パッセージ分割に加えて，分割されたパッセージに対応する用語名の追加を行うパッセージ分割手法である．これにより，事前学習言語モデルが構造依存の暗黙的な意味を捕捉できるよう工夫する．例えば，図 1 左側のような文書に対しては，構造依存パッセージ分割を用いた場合では，図 1 の水色部分のように分割が行われ，構造依存パッセージ分割拡張では，図 1 のピンク色部分のように分割が行われる．動詞に着目したテンプレートから学習事例

1：https://www.mext.go.jp/a_menu/shotou/new-cs/youryou/syo/index.htm(2022-12-12 確認)
2：https://stackoverflow.blog/2020/01/22/the-loop-2-understanding-site-satisfaction-summer-2019/（2022-12-12）確認

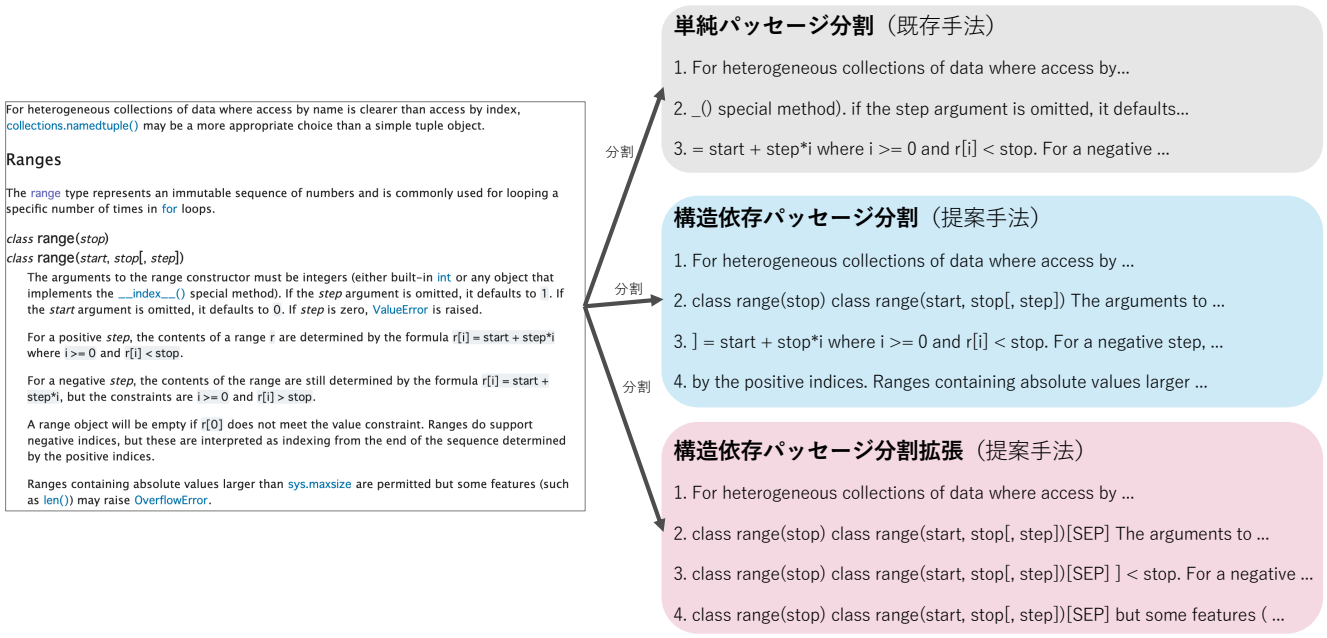

図 1　左側の図は「Built-in Types」から Range 型についての文書，右側は左側の文書を手法ごとに分割した結果．

を作成する方法は，プログラミング言語リファレンスにおいて頻出の，「Return an item which …」のように，動詞から始まる関数や，クラスの説明文に注目し，この説明文から質問-適合パッセージ-回答の組を作成するような手法である．この作成された質問-適合パッセージ-回答の組を用いて，事前学習言語モデルに対して追加の学習を行い，プログラミング言語リファレンスに対するドメイン適応を行う．

実験では，自身で作成した質問-適合パッセージ-回答の組からなる評価データセットを用いて，提案するパッセージ分割手法とドメイン適応手法の *Retriever* における検索性能と *Reader* におけるリランキング性能に対する有効性を評価した．実験の結果，提案手法のドメイン適応を行わなかった場合の構造依存パッセージ分割拡張が *Reader* のリランキング性能に対して統計的有意に高い MRR を示した．

この論文における我々の貢献を以下に示す：

（1） プログラミング言語リファレンスに対する質問応答タスクの提案

（2） 事前学習言語モデルへの入力に対するパッセージ分割手法とテンプレートから作成した事例を用いたドメイン適応手法の提案と実験の実施

本論文の構成は以下の通りである． 2 節では質問応答およびドメイン適応，事前学習言語モデルのプログラミングドメインにおける活用に関する関連研究について述べる． **??**節では問題設定を説明し，および．その主問題への適用方法と，提案手法について説明する． 4 節では実験結果を示す．最後に， 5 節では今後の課題と共に本論文の結論を述べる．

2　関 連 研 究

本節では，まず，質問応答について説明した後，近年盛んに研究されている事前学習言語モデルを用いた質問応答に対する手法について説明する．次に，事前学習言語モデルのドメイン適応について説明する．最後に，プログラミングドメインにおける，事前学習言語モデルの活用について説明する．

2.1　質 問 応 答

質問応答（Question Answering）は質問に対して自然言語で的確な回答を提供することを目的としている．質問応答は，回答が得られる情報源の種類によって Textual QA と Knowledge-Base QA の 2 種類に大別される [22] ．Textual QA は情報源が非構造化テキスト文書であり，Knowledge-Base QA は情報源が知識ベースである．さらに Textual QA には文脈の文章が与えられるか与えられないかという 2 種類のタスク設定があり，前者は機械読解（Machine Reading Comprehension）と呼ばれ，後者はオープンドメイン QA（Open-Domain Question Answering）と呼ばれ研究されている．機械読解は言語能力試験からヒントを得られたもので，与えられた質問に回答するために，指定された文脈の文章から，機械読解モデルが質問に対する回答を抽出あるいは生成することを目的としている．オープンドメイン QA は，指定された文脈の文章が存在せず，文書群から適合文書を検索し，取得された文書から回答を抽出あるいは生成することを目的としている．本論文が対象としているのは，オープンドメイン QA である．

近年，オープンドメイン QA では，事前学習言語モデルを活用した手法が盛んに研究されている．事前学習言語モデルを用いることで，文章を意味的な情報を含有する密なベクトル表現に埋め込み，従来の適合文書の検索手法に用いられてきた，TF-IDF や BM25 のように単語の出現頻度ベースの特徴量の疎なベクトル表現のみを用いた検索手法では検索できなかった適合文書の取得が可能となる．事前学習言語モデルを活

用した手法として Guu らによる REALM [7] や，Lee らによる ORQA [13] 等がある．これらの手法は，Wikipedia 等の通常の文章を対象として文章中のマスクされたトークンを回答するような事前学習を行う．しかしながら，この事前学習がプログラミング言語リファレンスで頻出する構造による暗黙的な意味を含む文章に対して有効的であるかは不明である．

2.2 ドメイン適応

ドメイン適応とは，あるソースドメインのデータ分布と，ソースドメインと別のターゲットドメインのデータ分布に対して，そのデータ分布が異なること（ドメインシフト）から生じる機械学習モデルの性能低下を抑えることに取り組む技術のことを指す．

事前学習言語モデルを用いた密ベクトルによる検索手法（Dense Retrieval）は，ドメインシフトによるモデルの汎化性能の低下により，TF-IDF や BM25 等の単語の出現頻度ベースの特徴量を用いた疎なベクトル表現を用いた手法より性能が低いことが Thakur らの先行研究 [19] や Chen らの先行研究 [2] にて指摘されている．この指摘点に対して，先行研究には，Sciavolino らによる，レアなエンティティに対して知識ベースからテンプレートを活用することによって事前学習言語モデルに対してレアなエンティティの情報を学習させることでドメイン適応を行なった研究 [17] や，Gangi らによる，COVID-19 ドメインに対して，sequence-to-sequence の生成モデルである BART [14] を使用することで質問-適合文書-回答のペアを作成し，ドメイン適応を行った研究 [6] がある．先行研究は，本論文が対象としているドメインが異なり，プログラミングドメインにおいても有効な手法であるかは不明である．

2.3 プログラミングドメインにおける事前学習言語モデル

プログラミングドメインにおける，事前学習言語モデルを活用した先行研究について説明する．

Feng らによる研究 [5] では，ソースコードの説明文とソースコードの両方に対して事前学習を行い，自然言語で記述された文章とソースコードの両方に対して密ベクトル空間に埋め込むことのできる，マルチモーダルな RoBERTa [15] ベースの事前学習言語モデル CodeBERT を提案し，入力を自然言語として，ソースコードを検索するタスクと，コードのドキュメンテーションを生成するタスクの評価を行なった．Huang らによる研究 [8] では，自然言語のクエリからソースコードを検索するためのクエリとソースコードの組で構成されたデータセット CoSQA と，クエリとソースコードのペアを前述した CodeBERT により密ベクトル空間に埋め込み，対照学習により，CodeBERT を学習するフレームワークである CoCLR を提案した．以上の研究ではクエリに対してソースコードを検索するタスクを対象としており，本論文は質問に対して自然言語で記述された文書から回答を抽出するという点で異なる．

Chen らによる研究 [1] では，自然言語での質問とそれに対して適合する GitHub リポジトリの組から構成されるデータセット Repo4QA について説明し，Huang らによる研究 [8] と同様に，CodeBERT により質問-リポジトリを密ベクトル空間に埋め込み，対照学習により，CodeBERT を学習するフレームワークである QuReCL を提案した．この研究と本論文では回答対象が，リポジトリかプログラミング言語リファレンスか，という点で異なり，プログラミング言語リファレンスにおいても有効な手法であるかは不明である．

3 提 案 手 法

本節では，まず，問題設定について説明する．次に，問題設定に対して用いる既存手法について説明し，既存手法の課題点を予備実験を通して示す．最後に，既存手法の課題点に対する提案手法であるパッセージ分割手法とドメイン適応手法について説明する．

3.1 問 題 設 定

本論文は，D をプログラミング言語リファレンスで構成される文書集合とし，与えられた質問 q に対する回答 a を，ある文書 d ($d \in D$) の部分文字列から抽出する形式の質問応答 [20] 問題を解く．例えば,「What is the method to lowercase a string?」という質問が与えられた時，質問に対する回答を，文書集合 D に含まれる文書「Built-in Types」の中から,「str.lower()」という部分文字列を抽出する．

3.2 密パッセージ検索による質問応答

本論文で提案するプログラミング言語リファレンスに対する質問応答を実現するための手法を図 2 に示す．この手法は Karpukhin らの先行研究 [10] で提案された検索モジュールとして Dense Passage Retrieval（密パッセージ検索）を用いた手法である．この手法は，予め，文書集合 D に含まれる全ての文書を特定の単語数ごとに分割を行い，パッセージ集合 P を作成する．質問 q が与えられた時に，パッセージ p_i（$p_i \in P$）の質問 q に対する適合度 s_i を算出し，適合度 s_i に基づいてパッセージ p_i のランク付けを行い，適合する上位 k 件を返す検索モジュールの *Retriever* と，*Retriever* が返した上位 k 件の適合パッセージをリランキングし，リランキングされた上位 1 件の適合パッセージから，質問 q に対する回答箇所 a として部分文字列を抽出する読解モジュールの *Reader* の 2 つから構成される．

3.2.1 Retriever

Retriever は，パッセージエンコーダ $E_P(\cdot)$ を用いて，パッセージの文字列を h 次元の実ベクトル空間に埋め込み，検索に用いる M 個の全パッセージを索引付けする．推論実行時にはパッセージエンコーダと異なる，質問エンコーダ $E_Q(\cdot)$ を用いて，入力された質問を h 次元の実ベクトルに埋め込み，質問ベクトルから最も適合するベクトルの k 件のパッセージ $P_{\mathrm{rel}} = \{p_1, p_2, \ldots, p_k\}$ を取得する．(P_{rel} の集合要素の添字は順序を表す.）質問とパッセージの適合度は以下の質問エンコーダの出力とパッセージエンコーダの出力を内積を計算する式で表される．

$$s_i = \mathrm{sim}(q, p_i) = E_Q(q)^\intercal E_P(p_i)$$

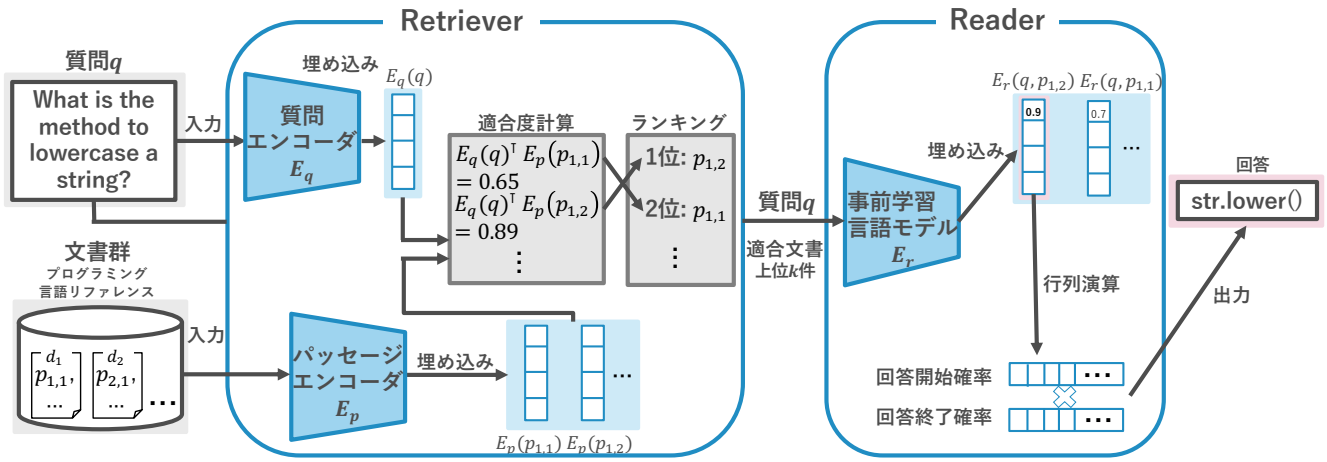

図 2　密パッセージ検索による質問応答 概要図

3.2.2　Retriver の学習

学習は，質問と質問に対して適合しないパッセージの組の距離よりも，質問と質問に対する適合パッセージの組の距離が近くなるような埋め込みベクトル空間を，質問エンコーダ E_Q とパッセージエンコーダ E_P が作成できるよう行う．学習データは $\{\langle q_i, p_i^+, p_{i,1}^-, ..., p_{i,n}^-\rangle\}_{i=1}^m$ で定義される．ここで m は学習データの事例数である．$n+1$ は学習時のミニバッチ中の学習事例の数である．p_i^+ は質問 q_i に対する正例である．$p_{i,n}^-$ は質問に対する n 個の負例である． 損失関数は以下の負の対数尤度で定義され，この損失関数から算出された各事例の損失の平均を最小化するように学習を行う．

$$L(q_i, p_i^+, p_{i,1}^-, ..., p_{i,n}^-) = -\log \frac{e^{\mathrm{sim}(q_i, p_i^+)}}{e^{\mathrm{sim}(q_i, p_i^+)} + \sum_{j=1}^{n} e^{\mathrm{sim}(q_i, p_{i,j}^-)}}$$

質問 q_i に対する正例は，文書集合 D のうち，質問に対する回答が含まれる正例パッセージ p_i^+ として定義する． 質問 q_i に対する負例は，学習時に，質問 q_i と同一ミニバッチ内に存在する正例パッセージ以外の全ての質問回答ペアの正例パッセージを負例パッセージとして定義した．

3.2.3　Retriever の推論

推論時には，質問 q が与えられた時，その埋め込み $\boldsymbol{v_q} = E_Q(q)$ を算出し，$\boldsymbol{v_q}$ に対して近似最近傍探索により，質問に対して最も近い埋め込みを持つと推定された上位 k 件の文書を取得する．

3.2.4　Reader

Reader は，まず，*Retriever* が検索した上位 k 件のパッセージ集合 $P_{\mathrm{rel}} = \{p_1, p_2, \ldots, p_k\}$ に対して，パッセージ選択スコアを付与する．*Reader* はこのパッセージ選択スコアに基づいて，*Retriever* が取得した上位 k 件のパッセージを並べ替える．次に，それぞれの文書から部分文字列を抽出し，部分文字列に対して，回答スパンスコアを割り振る．そして，パッセージ選択スコアが最も高い文書から，回答スパンスコアが最も高い部分文字列を抽出する．

パッセージ選択スコアと回答スパンスコアの導出には，機械読解モデルとして，事前学習言語モデルの BERT [3]（base，uncased）を用いる．BERT に対して，質問 q と *Retriever* が検索した上位 k 件の文書 P_{rel} の i 番目の文書を [CLS]q[SEP]p_i[SEP] というように結合して入力を行い，分散表現 $\mathbf{P}_i \in \mathbb{R}^{L\times h}$ を獲得する．ただし，L は入力する系列の最大の長さで，h は BERT が出力する最終隠れ層の次元数である．i 番目の文書の獲得した分散表現に対して，トークンが回答スパンの開始または終了位置である確率と文書が選択される確率を以下の式で定義する．

$$P_{\mathrm{start},i}(s) = \mathrm{softmax}(\mathbf{P}_i \mathbf{w}_{\mathrm{start}})_s \quad (1)$$

$$P_{\mathrm{end},i}(t) = \mathrm{softmax}(\mathbf{P}_i \mathbf{w}_{\mathrm{end}})_t \quad (2)$$

$$P_{\mathrm{selected}}(i) = \mathrm{softmax}(\hat{\mathbf{P}}^\intercal \mathbf{w}_{\mathrm{selected}})_i \quad (3)$$

ただし，$\hat{\mathbf{P}} = [\mathbf{P}_1^{[\mathrm{CLS}]}, \mathbf{P}_2^{[\mathrm{CLS}]}, \ldots, \mathbf{P}_k^{[\mathrm{CLS}]}] \in \mathbb{R}^{h\times k}$ （$\mathbf{P}_n^{[\mathrm{CLS}]}$ は BERT の [CLS] トークンに対応する分散表現という意味）であり，$\mathbf{w}_{\mathrm{start}}, \mathbf{w}_{\mathrm{end}}, \mathbf{w}_{\mathrm{selected}} \in \mathbb{R}^h$ は学習可能なパラメータである．また，softmax 関数の添字 s,t,i は softmax 関数の出力から添字 s,t,i の行を取得するという意味である．i 番目のパッセージに対して，s 番目から t 番目のトークンの回答スパンスコアを $P_{\mathrm{start},i}(s) \times P_{\mathrm{end},i}(t)$ として算出し，パッセージ選択スコアを $P_{\mathrm{selected}}(i)$ とする．

3.3　予 備 実 験

本節では， 3.2 にて説明した密パッセージ検索による質問応答を用いた，予備実験について説明する．

3.2 で説明した既存手法の密パッセージ検索を用いた手法では Natural Questions [12] 等の主に Wikipedia から作成された質問回答のペアを対象に手法が用いられているため，本論文が対象としているプログラミング言語リファレンスに対する質問応答では既存手法では回答できない事例があるのではないかと考え，既存手法の密パッセージ検索で回答できない事例を探索するために，予備実験を行った．

3.3.1　予備実験のデータセット

プログラミング言語リファレンスにはプログラミング言語 Python とともに配付されている標準ライブラリについての説明

が記述されている The Python Standard Library[3]（バージョンは 3.10.5 で，言語は英語）を使用した．プログラミング言語リファレンスから回答可能な質問を自身で作成した後，質問に対する回答と回答の含まれる適合パッセージを自身で選択し，質問-適合パッセージ-回答の組を作成した．

質問は，Factoid 型質問と呼ばれる，単文で記述された単純な事実が回答となるような質問のみを採用した．回答には，モジュール名や関数名，クラス名，返り値のような単文で記述された単純な事実を採用した．適合パッセージの元の文書群として，The Python Standard Library のそれぞれの文書に対して，前処理を行い，HTML 形式の文書から文字列部分を抽出し，検索モジュールへの基本単位に分割を行ったものを使用した．前処理では，表やコードブロックのような半構造化データと，ライブラリについての概要のみが記載されている文書を除外した．前処理後，各文書を Wang らの先行研究 [21] に従って，100 単語単位のパッセージに分割を行った．各パッセージに対して，パッセージ元の文書の h1 タグ中の文字列をタイトルとし，タイトルに加えて，[SEP] トークンを文頭に追加した．最終的に 9,906 個のパッセージを得た．

3.3.2 予備実験の質問応答システム

予備実験では，質問応答システムとして， 3.2 項で述べた *Retriever* と *Reader* の 2 つのモジュールで構成されるモデルを使用した．

Retriever の質問エンコーダとパッセージエンコーダには，BERT [3]（base，uncased）に基づいた学習済みモデル [4][5]を使用し，BERT の [CLS] トークンに対応する分散表現をそれぞれのエンコーダの出力とした．そのため，$h = 768$ である．また，文書の索引付けには，パッセージエンコーダを用いて，文書全てを分散表現に埋め込み，分散表現の索引づけを FAISS [9] により行った．検索実行時には，質問エンコーダを用いて質問を分散表現に埋め込み，索引付けと同様に FAISS を使用して，近似最近傍探索により質問の分散表現に近い分散表現を持つ文書を検索する．また $k = 100$ とした．

Reader の機械読解モデルには，*Retriever* の 2 つのエンコーダと同様に，BERT に基づいた学習済みモデル [6]を使用した．

3.3.3 予備実験の事例分析

予備実験の結果，以下のような事例が確認された．

（1） パッセージ分割により，暗黙的な構造による意味が失われ，回答不可能な事例

（2） 事前学習言語モデルがドメイン特有の知識を所有せずに，誤った回答が抽出される事例

本項では，上の事例について，例を挙げながら説明を行う．

まず，パッセージ分割によって，暗黙的な構造による意味が失われ，回答不可能な事例について説明する．

質問として，「What is the default value of the argument step in the range type？」という文章を質問応答システムに入力を行った．その結果，質問応答システムは，range 型について関係のないパッセージから，「zero」と出力を行った．しかし，設定した適合パッセージは range 型の引数 step についての説明記述のある文書 Bulit-in Types の表 1 中の “構造依存の意味” の行の適合パッセージであり，この適合パッセージを *Retriever* が取得できなかった．*Retriever* が取得できなかった原因として，設定した適合パッセージには引数 step についての記述はあるが，引数 step が range 型のものであるという説明が含まれていないことが考えられる．プログラミング言語リファレンスでは，図 3 のような記述が多くあり，このような構造を用いた記述により，Range 関数とその説明文の対応関係が表現されている．しかしながら，このような構造を用いた暗黙的な用語とその説明文の対応関係は，単純なパッセージ分割により失われしまう．このように，パッセージ分割によって，構造による暗黙的な意味が失われ，回答不可能な事例が確認された．

次に事前学習言語モデルがドメイン特有の知識を持っていないことから，誤った回答が抽出される事例について説明する．質問として，「What is the default value of for “end” in print function?」という文章を質問応答システムに入力した．その結果，質問応答システムは，“end” や “default” が多く含まれる文書を検索し，適合パッセージの表 1 中の “ドメイン特有の知識” の行の適合パッセージを上位 100 件中に取得することができなかった．この結果から，質問応答システムは，プログラミングドメインにおいて一般的な，文字列等を出力する処理を行う “print function” についての知識を持ち合わせていないということが考えられる．この失敗事例の原因として，Thakur らの研究 [19] でも指摘されているように，予備実験で使用した学習済みの事前学習言語モデルは，主に Wikipedia 由来のデータで学習が行われており，プログラミング言語リファレンスから作成された質問-適合パッセージ-回答の組と学習データの分布が異なることから，検索や抽出ができなかったということが考察できる．

3.4 パッセージ分割手法

本項では，3.3.3 で説明した失敗事例のうち，「パッセージ分割によって，暗黙的な構造による意味が失われ，回答不可能な事例」に対する提案手法のパッセージ分割手法について説明する．

本項では，3.3.3 予備実験の事例分析にて説明した失敗事例のうち，「パッセージ分割によって，暗黙的な構造による意味が失われ，解答不可能な事例」に対する提案手法のパッセージ分割手法について説明する．提案するパッセージ分割手法の疑似コードを Algorithm 1 に示す．最初に，提案手法の**構造依存パッセージ分割**について説明する．この分割手法は，プログラミング言語リファレンスの HTML 文書 D が与えられたとき，その文書から `p` 要素と `dl` 要素で構成されたノードを抽出し，`dl` 要素のノード（$node \in D$ に対して Algorithm 1 の処理を

3：https://docs.python.org/3.10/library/index.html (2022/07/11 確認)

4：https://huggingface.co/facebook/dpr-question_encoder-single-nq-base (2022/12/26 確認)

5：https://huggingface.co/facebook/dpr-ctx_encoder-single-nq-base (2022/12/26 確認)

6：https://huggingface.co/facebook/dpr-reader-single-nq-base (2022/12/26 確認)

class range(*stop*)
class range(*start*, *stop*[, *step*])

The arguments to the range constructor must be integers (either built-in int or any object that implements the __index__() special method). If the *step* argument is omitted, it defaults to 1. If

（中略）

start

The value of the *start* parameter (or 0 if the parameter was not supplied)

stop

The value of the *stop* parameter

step

The value of the *step* parameter (or 1 if the parameter was not supplied)

図 3　構造依存な意味を含む文書例

表 1　適合パッセージ例

事例の種類	適合パッセージ
構造依存の意味	of the stop parameter step The value of the step parameter (or 1 if the parameter was not supplied) The advantage of the range type over a regular list or tuple is that a range object will always take the same (small) amount of memory, no matter the size of the range it represents (as it only stores the start, stop and step values, calculating individual items and subranges as needed). Range objects implement the collections.abc.Sequence ABC, and provide features such as
ドメイン特有の知識	(*objects, sep=' ', end=' n', file=sys.stdout, flush=False) Print objects to the text stream file, separated by sep and followed by end. sep, end, file, and flush, if present, must be given as keyword arguments. All non-keyword arguments are converted to strings like str() does and written to the stream, separated by sep and followed by end. Both sep and end must

行う．まず，*node* に対して，DELCHILDDL メソッドを使用し，*node* に含まれる dl 要素を削除する．これにより，*node* 中の dd 要素に含まれる dl 要素が重複してパッセージ分割されることを防ぐ．次に CHILDREN メソッドを用いて，*node* の dt 要素と dd 要素で構成された子要素 n_c 全てに対して，TEXT メソッドを使用し，テキスト部分を抽出，*text* にテキストを追加する．その後，NAME メソッドを用いて，n_c のタグ名を取得し，そのタグ名が dd であり，かつ，NEXT メソッドと NAME を組み合わせて，n_c の次のノードのタグ名を取得し，そのタグ名が dt である．或いは，NEXT メソッドを使用し，次のノードが存在しないならば，リストである *segment* に *text* を追加し，*text* を CLEAR メソッドにより空の文字列にする．ただし，構造依存パッセージ分割では (∗) と右にコメントのある行の処理は実行しない．最後に，SPLIT メソッドを用いて，*segment* の各要素をある単語数 n_t で分割し，その結果を返す．以上のように，各 *node* に対して処理を行う．しかしながら，構造依存パッセージ分割のみでは，用語とその説明文が分割されることを防止できるが，その説明文が分割されてしまうと，用語とその説明文の構造により示される暗黙的な対応関係が失われてしまう．続いて，この課題に対するパッセージ分割手法である**構造依存パッセージ分割拡張**を説明する．基本的には，構造依存パッセージ分割とほとんど同じ処理を行うが，(∗) とコメントのある行の処理も構造依存パッセージ分割拡張では行う．(∗) のある行では，GETDTPARENTS 関数を用いて，n_c と *text* を入力し，n_c の親となる dt 要素のテキストを全て取得し，*text* の先頭に取得した親 dt 要素のテキストと [SEP] トークンを追加する．これにより，用語とその説明文の構造依存の暗黙的な対応関係を失うことなく，文書をパッセージに分割する．

3.5　ドメイン適応手法

本節では，3.3.3 で説明した失敗事例のうち，「事前学習言語モデルがドメイン特有の知識を持っていないことから，誤った回答が抽出される事例」に対する提案手法のドメイン適応手法について説明する．

プログラミング言語リファレンスで回答可能な質問で構成された質問応答データセットは我々が調査した限り存在しない．そのため，プログラミング言語リファレンスから，テンプレートを用いることで質問-適合パッセージ-回答の組を作成し，作成した組を用いることによって，プログラミング言語リファレンスに対するドメイン適応を行う．

テンプレートは，質問用と回答用を用意し，それぞれプログラミング言語リファレンスで多く見られる，図 4 のような，用語とその用語の説明文において，動詞から始まる文章に着目し，

Algorithm 1 提案手法のアルゴリズム

Require: n, a `dl` HTML node
Ensure: The child nodes of *node* consist of `dt` or `dd` HTML nodes.
Ensure: *node* has one or more `dt` and `dd` HTML nodes as children.

```
function 構造依存パッセージ分割拡張 (node)
    delChildDl(node)
    segment ← []
    for n_c ∈ children(node) do
        text ← text + text(n_c)
        if name(n_c) = dd and name(next(n_c)) = dt then
            text ← getDtParents(n_c, text) + [SEP] + text ▷ (*)
            append(segment, text)
            clear(text)
        else if next(n_c) ≠ null then
            text ← getDtParents(n_c, text) + [SEP] + text ▷ (*)
            append(segment, text)
            clear(text)
        end if
    end for
    return Split(segment)
end function
```

sorted(*iterable*, */*, ***, *key=None*, *reverse=False*)

Return a new sorted list from the items in *iterable*.

Has two optional arguments which must be specified as keyword arguments.

key specifies a function of one argument that is used to extract a comparison key from each element in *iterable* (for example, key=str.lower). The default value is None (compare the elements directly).

図 4 「Built-in Functions」より，組み込み関数 sorted の説明文

以下のように設定した．

- 質問用テンプレート：What {directive} {verb} {sentence}?
- 回答用テンプレート：{term}

ここで，中括弧中に表されたものは変数である．term は関数名やメソッド名，クラス名等を表す．directive は，term が関数なのかメソッドなのかクラスなのかを明示する [7]HTML のクラス名を表す．また，directive は自身で設定したクラス名のみを使用し，省略可能である．verb は自身で設定した ’ Return ’ 等の動詞を表す．sentence は verb の後の単語から，.（ドット），:（コロン），;（セミコロン）までの単語である．図 4 において，term は “sorted(iterable, /, *, key=None, reverse=False)” であり，directive は “function” であり，verb は “Return” であり，sentence は “a new sorted list from the items in iterable” である．

以上のように設定した質問用テンプレートと回答用テンプレートに基づいて作成した質問と回答のペアを用いて，単純パッセージ分割，構造依存パッセージ分割，構造依存パッセージ分割拡張のそれぞれの手法により作成した各々のパッセージ集合から，term と sentence の含まれるパッセージをランダムに選択し，適合パッセージとして，質問-適合パッセージ-回答の組を作成した．

以上のように設定したテンプレートを用いて半自動的に質問-適合パッセージ-回答の組を作成し．獲得した組を用いて，3.2.1 項で述べた *Retriever* に対する学習を行う [8]．

3.5.1 逆翻訳によるデータ拡張手法

逆翻訳はある言語の文章を別の言語に翻訳した後，元の言語に再度翻訳することであり，自然言語処理タスクにおいてデータ拡張の手法として利用されている [4] [18]．本項では，逆翻訳を活用し，前述したテンプレートを用いて作成した質問-適合パッセージ-回答の組のデータを増やす手法について説明する．

逆翻訳は質問-適合パッセージ-回答の組のうち，質問と適合パッセージに対して行い，元の回答と逆翻訳後の質問と適合パッセージを組とした．また，質問と適合パッセージのどちらか一方でも，逆翻訳前と逆翻訳後の文章が同じ場合，その逆翻訳後の質問-適合パッセージ-回答の組は破棄した．

逆翻訳は，次の手順によって行われる．(1) ある言語（ソース言語）の系列 s_{raw} が与えられたとき，ソース言語からターゲット言語へ順翻訳を行う翻訳モデル T_{forward} を用いて翻訳を行い，ターゲット言語の系列 s_{via} を獲得する．(2) 系列 s_{via} をソース言語への逆翻訳を行う翻訳モデル T_{back} を用いて翻訳を行い，ソース言語の系列 s_{back} を獲得する．

順翻訳モデル T_{forward} には，英語からドイツ語への翻訳を行う Ng らの [16] の機械翻訳モデル [9]を使用し，逆翻訳モデル T_{back} には，順翻訳モデルと同様のモデルのドイツ語から英語への翻訳を行う機械翻訳モデル [10]を使用した．

4 実　　験

本節では，3.4 項と 3.5 項にて述べた提案手法に対する実験について説明する．

まず使用するデータセットについて説明する．次に，実験設定について説明する．最後に実験結果について考察を踏まえながら述べる．

4.1 データセット

実験において使用するデータの事例数を表 4.1 に示す．(1 事例は質問-適合パッセージ-回答の 1 組である．)

3.5 項で説明した質問-適合パッセージ-回答の組で構成されたデータセットを学習データと検証データと評価データに 8:1:1 の割合で分割した．分割された内の学習データを使用して，質問応答システムの学習を行う．また，逆翻訳によるデータ拡張手法にて作成した，データセットに対しては，逆翻訳を行う前の学習データの質問-適合パッセージに対応したもの使用した．

検索対象となる文書集合としては，3.3.1 項にて説明した手法（単純パッセージ分割）をにより作成された文書集合 P_{base}，

7：https://devguide.python.org/documentation/markup/#information-units(2022/12/26 確認)

8：予備実験において，ほとんどの事例が *Retriever* による検索に失敗した一方，*Reader* に対して適合パッセージを入力したところ，ほとんどの事例において抽出が成功したため，今回は *Retriever* に対してのみ学習を行なった．

9：https://huggingface.co/facebook/wmt19-en-de (2022/12/26 確認)

10：https://huggingface.co/facebook/wmt19-de-en (2022/12/26 確認)

表 2　パッセージ数

	P_{base}	P_{split}	P_{extend}
パッセージ数	9,906	15,377	14,940

表 3　データセット事例数

	事例数
テンプレート	2,052
逆翻訳	1,942
テンプレート +逆翻訳	3,994
検証テンプレート	257
検証逆翻訳	247
検証テンプレート +検証逆翻訳	504
評価テンプレート D_{temp}	257
評価データセット D_{own}	70

構造依存パッセージ分割により作成された文書集合 P_{split}，構造依存パッセージ分割拡張により作成された文書集合 P_{extend} の 3 つを使用した．それぞれの文書数は表 4.1 に示す．

4.2 実験設定

本実験における実験設定について説明する．

本実験で用いる質問応答システムは，3.3 項にて説明したものと同一である．本実験は *Retriever* モジュールの質問エンコーダとパッセージエンコーダに 3.2.1 項で説明した学習を行い，評価を行う．また，*Retriever* が返すパッセージ数は，$k = 100$ とした．質問エンコーダとパッセージエンコーダのバリエーションとして，3.4 項にて述べた提案手法を評価するために，以下の 3 つのモデルを用意する．

（1） 単純パッセージ分割 *Retriever*

（2） 構造依存パッセージ分割 *Retriever*

（3） 構造依存パッセージ分割拡張 *Retriever*

さらに，3 つのモデルに対して，学習データのバリエーションを以下の 3 つ用意した．

（1） 学習用テンプレートデータ（テンプレート）

（2） 学習用テンプレート逆翻訳データ（逆翻訳）

（3） 学習用テンプレートデータ+学習用テンプレート逆翻訳データ（テンプレート+逆翻訳）

ここで，括弧中の “テンプレート”，“逆翻訳”，“テンプレート+逆翻訳” は 4.1 のデータセットに対応している．また，テンプレートと逆翻訳の両方で学習を行わない，つまり，配布されている学習済みのモデルをそのまま使用した場合の評価も行う．

また，モデルはエポック毎に検証データの損失の値を計算し，学習終了後，最も検証データの損失の値が小さいものを保存し，評価に使用する．そこで，検証データのバリエーションとして，以下の 3 つを用意した．

（1） 検証用テンプレートデータ（検証テンプレート）

（2） 検証用テンプレート逆翻訳データ（検証逆翻訳）

（3） 検証用テンプレートデータ+検証用テンプレート逆翻訳データ（検証テンプレート+検証逆翻訳）

ここで，括弧中の “検証テンプレート”，“検証逆翻訳”，“検証テンプレート+検証逆翻訳” は 4.1 のデータセットに対応している．

学習の際のバッチサイズは 32，エポック数は 100 と設定した．learning rate は 10^{-5} として optimizer には Adam [11]，warm-up rate と dropout rate は共に 0.1 として線形スケジューラを用いた．

評価用データセットには，自身で作成したデータセット D_{own} と，4.1 項で説明したテンプレートから作成した評価用データセット D_{temp} を用いた．

評価指標には，精度（accuracy）と MRR（Mean Reciprocal Rank）の 2 つを用いる．精度は検索結果の上位 k 件に，質問（クエリ）に対する適合パッセージが含まれた件数を全事例数で割った指標であり，本実験では *Retriever* が評価データ中の質問に対してどれくらい上位 k 件中に適合パッセージを検索できたかを評価する．MRR は質問に対する上位 k 件のランキング中の，適合パッセージの順位の逆数を全ての質問に対して平均した指標であり，本実験では *Reader* がリランキングしたランキング結果に対して，可能な限り上位に適合パッセージが順位づけられていることを評価する．

4.3 実験結果

表 4 の列名 D_{own} と表 5 に D_{own} に対する 3.4 項と 3.5 項で説明した提案する手法の実験結果である精度と MRR を示す．結果として，単純パッセージ分割を用いた，モデルを学習せずにそのまま使用した結果が最も良い精度であり，パッセージ分割手法として，構造依存パッセージ分割拡張を用いた，配布されている学習済みモデルが最も良い MRR であった．また，繰り返しなしの 2 元配置分散分析を算出された MRR に対して行なった．その結果，前述した 30 種類のバリエーションの F 値 $F(29, 2001) = 1.90$ であり，MRR において，30 種類のバリエーションによる効果が統計的に有意であることが分かった（$p < 0.05$）．さらに MRR は *Retriever* がより良い精度で検索結果を返すと，向上すると考えられるため，パッセージ分割手法以外の同条件で比較的精度のスコアが良い，パッセージ分割手法の配布されている学習済みモデルを学習せずにそのまま使用した結果の MRR に対して，Tukey HSD 検定を行い，パッセージ分割手法がそれぞれの手法との間で統計的に有意な差が認められた（$p < 0.05$）．この結果から，パッセージ分割手法によって，単純なパッセージ分割を行うよりも，提案手法である構造依存パッセージ分割拡張を用いた場合の方が，*Reader* のリランキング性能に対して有効であるということが考えられる．しかしながら，学習を行わなかった場合の方が，精度と MRR の両方に対して良い結果となったため，学習データに用いたテンプレートから作成したデータの分布では，自身で作成した評価セットの分布を近似することができなかったのではないかと考えられる．また，どのパッセージ分割手法を用いた場合でも，学習データにテンプレートのみを用いる場合よりも，逆翻訳のみを使用またはテンプレートに逆翻訳を追加した場合がより良い精度を出していることが分かった．*Retriever* に対して，逆翻訳を用いて作成した事例を追加することで，検索性能が向上する可能性を示唆していると考えることができる．また，検証データのバリエーションにおいて，検証逆翻訳を使用し，パッセージ分割手法として構造依存パッセージ分割拡張を用いて，

表 4　配布されているモデルそのままでの評価

	D_{own}		D_{temp}	
手法	Accuracy@100	MRR	Accuracy@100	MRR
単純パッセージ分割	**25.714**	0.0577	9.339	0.0573
構造依存パッセージ分割	11.429	0.0238	**16.732**	0.0667
構造依存パッセージ分割拡張	24.286	**0.0693**	4.280	**0.1695**

表 5　データセット D_{own} に対する評価

	Accuracy@100			MRR		
手法	検証テンプレート	検証逆翻訳	検証テンプレート+検証逆翻訳	検証テンプレート	検証逆翻訳	検証テンプレート+検証逆翻訳
単純パッセージ分割（テンプレート）	2.857	2.857	5.714	0.0156	0.0152	0.0042
〃（逆翻訳）	5.714	7.143	7.143	0.0196	0.0210	0.0239
〃（テンプレート+逆翻訳）	10.000	8.571	10.000	0.0090	0.0266	0.0170
構造依存パッセージ分割（テンプレート）	2.857	4.286	4.286	0.0025	0.0030	0.0037
〃（逆翻訳）	11.429	11.429	8.571	0.0177	0.0096	0.0123
〃（テンプレート+逆翻訳）	14.286	15.714	11.429	0.0122	0.0117	0.0113
構造依存パッセージ分割拡張 （テンプレート）	2.857	4.286	2.857	0.0191	0.0090	0.0032
〃（逆翻訳）	14.286	12.857	11.429	0.0136	0.0094	0.0078
〃 （テンプレート+逆翻訳）	12.857	**20.000**	14.286	0.0407	**0.0470**	0.0235

表 6　データセット D_{temp} に対する評価

	Accuracy@100			MRR		
手法	検証テンプレート	検証逆翻訳	検証テンプレート+検証逆翻訳	検証テンプレート	検証逆翻訳	検証テンプレート+検証逆翻訳
単純パッセージ分割（テンプレート）	63.424	51.362	49.805	0.1266	0.1198	0.1216
〃（逆翻訳）	52.918	44.747	51.751	0.1312	0.0977	0.1266
〃（テンプレート+逆翻訳）	75.097	73.541	74.319	0.1556	0.1556	0.1538
構造依存パッセージ分割（テンプレート）	66.148	71.206	69.650	0.1601	0.1600	0.1417
〃（逆翻訳）	59.533	58.366	58.366	0.1423	0.1683	0.1420
〃（テンプレート+逆翻訳）	**82.101**	80.545	**82.101**	0.1613	0.1659	0.1628
構造依存パッセージ分割拡張 （テンプレート）	70.428	59.533	71.595	0.1554	0.1536	**0.1919**
〃（逆翻訳）	54.086	59.922	59.144	0.1327	0.1755	0.1636
〃 （テンプレート+逆翻訳）	73.933	77.043	76.265	0.1700	0.1845	0.1884

学習データをテンプレート+逆翻訳とした場合の精度と MRR が最も高い結果となった．データセット D_{own} の事例数は表 5 に示した通り，70 件となっており，非常に小さい．今後は，評価用セットの事例数を複数人でのアノテーションを施す等の手法を用いることによって増加させ，提案手法であるパッセージ分割手法とドメイン適応手法に対する評価を行いたい．

表 4 の列名 D_{temp} と表 6 に D_{temp} 3.4 項と 3.5 項で説明した提案する手法の実験結果である精度と MRR を示す．同じ学習データ，検証データを用いた場合では，学習データにテンプレート+逆翻訳，検証データに検証テンプレートを用いた，パッセージ分割手法を構造依存パッセージ分割拡張とした場合を除いて，パッセージ分割手法として，単純パッセージ分割を用いたものよりも，構造依存パッセージ分割と構造依存パッセージ分割拡張を用いたものがより良い MRR であった．また，繰り返しなしの 2 元配置分散分析を算出された MRR に対して行なった．その結果，前述した 30 種類のバリエーションの F 値 $F(29, 7242) = 6.30$ であり，MRR において，30 種類のバリエーションによる効果が統計的に有意であることが分かった（$p < 0.05$）．さらに，MRR に対して，Tukey HSD 検定を行い，学習データにテンプレート，検証データに検証テンプレート+検証逆翻訳を用いた場合の単純パッセージ分割と構造依存パッセージ拡張との間を除いて，同条件の学習データ，検証データ上でのパッセージ分割手法がそれぞれの手法との間で統計的に有意な差が認められた（$p < 0.05$）．この結果から，逆翻訳による学習データの拡張はテンプレートから作成した質問に対する，質問応答システムの *Reader* のリランキング性能に対して有効であると考えることができる．

5 ま と め

本論文では，プログラミング言語リファレンスに対する質問

応答を提案した．質問応答の形式として，与えられた質問に対して文書集合を検索し，検索によって取得された文書から質問に対する回答を抽出する形式の質問応答に，事前学習言語モデルに基づいた，*Retriever* と *Reader* を用いる方式を採用した．この質問応答手法をプログラミング言語リファレンスへ適用するに当たり，構造依存パッセージ分割と構造依存パッセージ分割拡張の 2 つの文書構造を考慮したパッセージ分割，および，テンプレートを用いた半自動的に生成された質問-適合パッセージ-回答の組によるドメイン適応を提案した．実験では，自身で作成したプログラミング言語リファレンスに対する質問と質問に対する適合パッセージと回答の組から構成される評価セットを用いて，2 つの提案手法を評価した．その結果，提案手法のドメイン適応を行わなかった場合の構造依存パッセージ分割拡張が *Reader* のリランキング性能に対して有効であることがわかった．また，提案手法のドメイン適応に用いたテンプレートから作成した質問-適合パッセージ-回答の組のデータ分布では，自身で作成した評価セットの分布を近似することができず，ドメイン適応が上手くいかなかった．今後の課題として，有効なドメイン適応手法の模索をすることが考えられる．

謝辞　本研究は JSPS 科研費 22H03905 の助成を受けたものです．ここに記して謝意を表します．

文　献

[1] Minyu Chen, Guoqiang Li, Chen Ma, Jingyang Li, and Hongfei Fu. Repo4qa: Answering coding questions via dense retrieval on github repositories. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1580–1592, 2022.

[2] Tao Chen, Mingyang Zhang, Jing Lu, Michael Bendersky, and Marc Najork. Out-of-domain semantics to the rescue! zero-shot hybrid retrieval models. In *European Conference on Information Retrieval*, pages 95–110. Springer, 2022.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*, 2021.

[5] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. Codebert: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, 2020.

[6] Revanth Gangi Reddy, Bhavani Iyer, Md Arafat Sultan, Rong Zhang, Avirup Sil, Vittorio Castelli, Radu Florian, and Salim Roukos. Synthetic target domain supervision for open retrieval qa. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1793–1797, 2021.

[7] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. corr abs/2002.08909 (2020). *arXiv preprint arXiv:2002.08909*, 2020.

[8] Junjie Huang, Duyu Tang, Linjun Shou, Ming Gong, Ke Xu, Daxin Jiang, Ming Zhou, and Nan Duan. Cosqa: 20,000+ web queries for code search and question answering. 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)*, pages 5690–5700, 2021.

[9] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

[10] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020.

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

[12] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[13] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, 2019.

[14] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.

[15] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. 2019.

[16] Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook fair's wmt19 news translation task submission. *arXiv preprint arXiv:1907.06616*, 2019.

[17] Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148, 2021.

[18] Connor Shorten, Taghi M Khoshgoftaar, and Borko Furht. Text data augmentation for deep learning. *Journal of big Data*, 8(1):1–34, 2021.

[19] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 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)*, 2021.

[20] Ellen M Voorhees et al. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82, 1999.

[21] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. Multi-passage bert: A globally normalized bert model for open-domain question answering. 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)*, pages 5878–5882, 2019.

[22] Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*, 2021.