オープンドメイン質問応答における 文集合と位置情報を用いた抽出精度向上に関する検証

初鹿憂　　柴田千尋
法政大学 理工学部 創生科学科
yu.hatsushika.2r@stu.hosei.ac.jp,chiro @ hosei.ac.jp

概要

近年，BERT や Transformer を用いた深層学習モデルが様々な自然言語処理のタスクで高い精度を上げているが，質問応答もその例外には当たらない．特に，最近，鈴木ら [1] によって作成された日本語版質問応答データセットが公開されたことにより，日本語でも質問応答に関する研究を進めることが容易になった. 本研究では，日本語版質問応答データセットを用いて Retriever-Reader とよばれる質問応答のためのモデルに対して， 精度改善に向けた手法を 2 つ提案し，その結果を検討する．Retriever 側は，質問文に対する関連度が高い記事候補を関連度の高い順に抽出する．一方，Reader が正解の解答を出力できる場合は，候補となる記事中に正解が含まれている場合がほとんどであることが，上記データセットを用いた実験の結果わかっている．したがって，Reader が出力する候補に正解が含まれる確率を，なるべく上げることが重要となる．記事は通常多数の文を含むが，そのすべての文が質問文に関連しているわけではない．そのため，同じ数のトークンを Reader に入力する場合，記事よりも文のほうがより多数の候補を入力することができる．本稿では，提案手法の 1 つとして，Retriver に，記事の代わりに文ごとに候補を抽出させることを提案する．

1　はじめに

近年，ニューラルネットワークを用いたモデルがクイズ形式の質問に対して高精度に解答する研究が盛んに行われている [2]. これらの研究が進むことで従来の検索システムに比べて，より自然な言葉での質問応答が可能になると考えられている．

出題範囲に制限のない広範囲な一般常識が必要となるようなクイズに対して，高精度で解答するためには，なるべくたくさんの知識を利用することが必要である．そのため，正解するためには膨大な記事の中から，答えを含む可能性の高いものを抽出するプロセス，及び，抽出された記事のなかから，文脈を考慮して答えとなる短いフレーズを導くためのプロセスが必要となる．

与えられた質問に対して答えを正しく導き出すために用いられる主な手法としては， Retriever-Reader モデルが挙げられる [3]．Retriever とはある記事集合から質問文の情報を用いて関連する記事を抽出することが行われ Reader では Transformer を元にした言語モデルを用いて抽出した関連する記事から答えを導き出すことが行われる．正答数を上げるためには Retriever と Reader 両方の精度を上げる必要がある．Retriever の精度は抽出した N 件の記事の中に答えを含んだ記事が含まれるかで測られ， Reader の精度は答えが出力できるかで測られる．今回の実験では Retriever に Dense Passage Retriever [4] を， Reader に Fusion-in-Decoder [5] を採用する．本研究では 2 つの手法を提案する．1 つ目は Retriever に従来用いていた記事集合を文ごとに分割した文集合を用いる．2 つ目は抽出された記事に対して位置情報を用いる．

2　関連研究

2.1　Retriever-Reader モデル

Retriever-Reader モデルは文を抽出する役割である Retriever と答えを生成する役割である Reader という 2 つの部分から構成されている．Retriever では記事集合と質問文の類似度を計算することによって全ての記事に対して質問文との関連度スコアを付け，スコア上位から順番に抽出する．Reader では抽出した記事集合から, 答えとなるフレーズを Transformer モデルにより生成する. 今回は埋め込み表現に変換し連結させることにより言語モデルの入力として解答

を生成する．

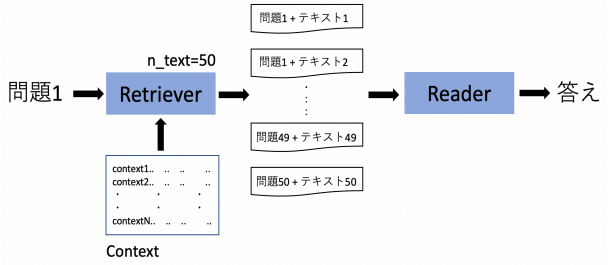

図 1　Retriever-Reader モデル

2.2　DPR (Dense Passage Retriever)

DPR[4] は質問に対して関連する記事を大規模な記事集合から意味的に抽出するモデルである．まず質問文 q と記事集合 p に対して，BERT [6] を用いて密なベクトル表現 $E(q)$，$E(p)$ にエンコードする．次に式 1 のように内積を計算し関連性の高い記事を順番に出力する．

$$\mathrm{sim}(q,\ p) = E_Q(q)^T E_P(p) \tag{1}$$

DPR の学習データセットは，Elasticsearch [1] を用いて作られ，$(q_i, p_i^+, p_{i1}^-, p_{i2}^-, \cdots, p_{iN}^-)$ で定義される．q は質問文，p^+ はポシティブパッセージ，p^- はネガティブパッセージを表す．モデルは式 1 の損失関数が最小になるように 2 つの BERT エンコーダを学習する．ポジティブパッセージとの類似度が高く，ネガティブパッセージとの類似度が低いほど損失関数は小さくなる．

$$\begin{aligned}&\mathrm{L}(q_i, p_i^+, p_{i1}^-, p_{i2}^-, \cdots,\ p_{iN}^-)\\ &= -\log \frac{e^{\mathrm{sim}(q_i, p_i^+)}}{e^{\mathrm{sim}(q_i, p_i^+)} + \sum_{j=1}^{n} e^{\mathrm{sim}(q_i p_{ij}^-)}}\end{aligned} \tag{2}$$

2.3　FiD (Fusion-in-Decoder)

Fusion-in-Decoder[5] は Retriever-Reader モデルの Reader として用いられている．言語モデルをベースに作られており，質問文と記事から答えを生成できる．Retriever 側で抽出された複数の記事をエンコードした後に連結することで複数の記事を入力可能にしていることが特徴である．モデルとしては，一般的な翻訳モデルと同じように，次トークンの予測するデコーダを用いている．

1）　https://www.elastic.co/jp/elasticsearch/

3　提案手法

提案手法としては，以下に述べるように，Retriever と Reader に対してそれぞれ 1 つずつ工夫を行なう．

3.1　文集合

記事集合として wikipedia などのクイズとは直接関連のない一般的な文書セットを用いた場合，記事の中に多数の文が含まれ，さらに質問文の解答には必要のない文も多数含まれている．例えば表 1 の例に示すように，質問「ゴルフボールの表面につけられているくぼみのことをなんと言うでしょう」に対し，正答が「ティンプル」であるような問題に対して，「乱流」というタイトルの記事が抽出されている．記事には，3 文含まれているが，回答には赤く色付けした 2 文目のみが関係していると考えらえる．

本研究では，関連する文をより多く正確に抽出するために，分け方を記事ごとから文ごとにすることを提案する．具体的には Retriever に用いる記事集合を文集合に変更して，事前学習を行なう．文集合で学習をした Retriever を用いて文章抽出を行なう．

表 1　抽出した記事に含まれる文章の例

質問文	ゴルフボールの表面につけられているくぼみのことをなんと言うでしょう？
答え	ディンプル
記事タイトル	乱流
記事	乱流の数値シミュレーションは、気象予報や自動車等の空力設計からノートパソコンの冷却まで工学的には非常に幅広く利用されている。ゴルフボール表面につけたディンプルによる飛距離延伸 (マグヌス効果も参照)、新幹線 500 系電車パンタグラフの突起による騒音低減などにも乱流の効果が応用されている。しかし高い計算機性能を要求するため、スーパーコンピュータなど HPC(高性能計算) の重要な用途の一つになっている。

3.2　位置情報

もう一つの提案は，Reader の入力に関するものである．前述のように，Retriver は関連度の高い順に記事を列挙する．Retriver が関連度が高いと推定したとしても，必ずしも正答となるフレーズがその記事の中に含まれているわけでは当然ないが，実際に，実験結果から，図 2 に示すように，Retriever

で抽出した記事を分析した結果，解答を含んだ記事が検索順位上位に集まる傾向が高いということがわかる．そこで，情報の重要度を記事位置ごと与えることで，精度が向上する可能性があるため，Reader に入力する際に位置情報を加える手法を提案する．方法としては記事の埋め込み表現に位置エンコーディングを加算する．位置エンコーディングは Transformer Encoder [7] の位置エンコーディングと同じものを用いる．

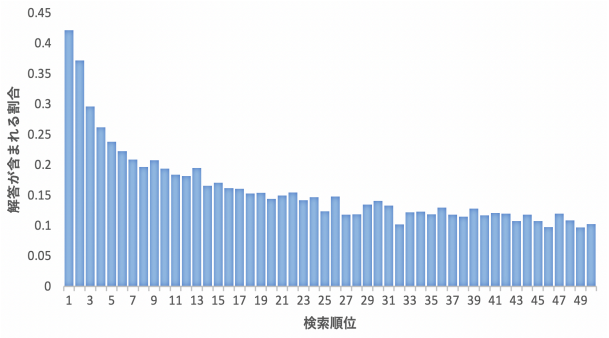

図 2　解答が含まれる記事の分布

4　データセット

4.1　コンテキスト

記事集合や文集合といった DPR で抽出する際の検索元のデータセットをコンテキストと呼ぶ．

4.1.1　記事集合

AI 王クイズ日本一決定戦に公開されていたウィキペディアを用いて作られた記事集合を利用した．記事数としては 4,288,199 件のものを利用する．実際，1 件の記事サイズは大きいため，段落ごとに区切ったデータを記事集合として，記事に段落があった場合同タイトルの別の記事として保存されている．

4.1.2　文集合

今回記事ごとではなく，文ごとのインデックスを作成するために，AI 王クイズ日本一決定戦に公開されていたウィキペディアのデータとコードを用いて集められていた記事集合を句点で区切り，再度ラベルを付け直す．その結果，表 2 に示したようにラベルの総数が約 430 万件から約 2,000 万件に増加した．このため記事または文ベクトルを作成する際に約 5 倍の時間がかかった．

表 2　検索インデックスの総数

	記事ごと	文ごと
インデックス数	4,288,199	**20,728,072**
インデックス作成時間	4 時間 50 分	**21 時間 7 分**

4.2　訓練データ

本研究では AI 王クイズ日本一決定戦で公開されていた質問応答タスクの日本語データセットを用いる．JAQKET [1] をもとにして作られている．訓練データセットとして 22,335 件の質問文と解答が含まれている．また，質問文に対して Elasticsearch を用いることで正解が含まれている記事であるポジティブパッセージを抽出する．さらにネガティブパッセージとして回答が含まれていない記事を付与する． Retriever の訓練データは質問文，p^+ はポシティブパッセージ，p^- はネガティブパッセージを用いて $(q_i, p_i^+, p_{i1}^-, p_{i2}^-, \cdots, p_{iN}^-)$ で定義される．

Reader における訓練データは，質問文を Retriever に入力し検索をかけて得られる記事，記事タイトルと質問文を上位 N 件まで組み合わせて作る．評価データは JAQKET から訓練データに含まれない 1,000 件の問題を利用して，同様にして作られる．

5　実験

5.1　実験設定

表 3 に示したハイパーパラメータを基準としてコンテキストと Reader の入力情報に対して変更を加えて提案手法の有効性を検証する． Retriever に事前学習済みの日本語 BERT[2] を用いた Dense Passage Retriever， Reader に事前学習済みの日本語 T5 [3] を用いた Fusion-in-Decoder を採用する．また，Fusion-in-Decoder に入力するコンテキスト数は 50 と 100 で実験を行う．

5.2　抽出精度の比較

記事ごとのインデックスを使用した基準モデルと文ごとのインデックスを使用したモデルに関して比較実験を行なった．結果は表 4 のようになり，数値は，上位 N 件までの記事または文を抽出した際に，

2）https://huggingface.co/cl-tohoku/bert-base-japanese-whole-word-masking
3）https://huggingface.co/sonoisa/t5-base-japanese

表 3　ハイパーパラメーター

	Retriever	Reader
バッチサイズ	64	1
ベクトルの次元数	768	768
学習率	1e-5	5e-5
ドロップアウト率	0.3	-
weight decay	0.0	0.01
optimizer	adam	adamw
fp16	あり	なし

答えが含まれた記事が抽出された割合を示す. インデックスのサイズは表 2 より文集合は記事集合の 4 から 5 倍であるため，コンテキスト 1 つ当たりのサイズは小さいと推測される. 文ごとで精度が約 0.85 になるのは上位 100 件の時であるのに対して記事ごとでは上位 50 件の時であるため，抽出精度が同等に達するために必要なコンテキストの差は 2 倍である．そのため，少ない量の文で解答を抽出できていることが分かる.

表 4　抽出精度の比較

記事または文の数	記事ごと	文ごと
上位 1 件	0.414	0.328
上位 25 件	0.811	0.733
上位 50 件	0.845	0.794
上位 75 件	0.86	0.827
上位 100 件	0.871	0.847

5.3　解答生成精度の比較

次に記事集合を用いて学習させた FiD と文集合を用いて 60,000 ステップ学習させた FiD に対して評価を行う. 結果は表 5 のようになった. 記事集合でテキスト数 50 の時と，文集合でテキスト数 100 の時を比較すると表 4 より解答を含んだコンテキストの数は同等であったが精度は記事集合の方が高いが，実際に Reader 入力されている文数やトークン数は，文集合のほうが少ないため，入力する文数やトークン数を揃えて，精度を比較をする必要がある.

表 5　FiD の結果

用いた手法	テキスト数	精度
記事集合	50	0.616
文集合	50	0.595
	100	0.606

5.4　位置情報を入力した結果

DPR を用いて抽出した記事 50 件に対して Transformer の位置エンコーディング [7] を参考に位置情報を追加する. 位置情報を加えたモデルと基準モデルに対して 90,000 ステップ学習する. 位置情報を加えて学習された FiD に対して行なった評価を表 6 に示す.

表 6　記事集合と位置情報を用いた結果

用いた手法	テキスト数	精度
記事集合	50	**0.613**
記事集合+位置 Enc	50	0.602

次に DPR を用いて抽出した文 50 件， 100 件に対して位置エンコーディングを加えて 60,000 ステップ学習する. 位置情報を加えて学習された FiD に対して行なった評価を表 7 に示す. 表よりわかるように，少なくとも本稿で述べた位置情報の入力の手法では，精度向上に寄与しないことがわかる.

表 7　文集合と位置情報を用いた結果

用いた手法	テキスト数	精度
文集合	50	**0.594**
	100	0.601
文集合+位置 Enc	50	0.589
	100	**0.606**

6　結論

本稿では，コンテキストとして用いていた記事集合を文集合に変えたることにより， DPR での抽出精度がどのように変化するかを検討した．その結果，少ない文章数で解答を含めることができていると言える．一方で，FiD で記事集合と文集合で解答を含むコンテキストを抽出する精度が同等であった場合は, Reader の精度は記事集合の方が高くなっている事がわかる. これは，記事集合の場合は，入力の文数が少なくなっているためであり，入力の文数やトークン数を揃えた上で比較する必要があるため，フューチャーワークとする．位置情報の入力に関しては, 精度の改善は認められなかったが， 問題それぞれで分布が少しずつ変わっている可能性があるため，今後は広い範囲に対して位置情報を取る改善が必要である.

今後の展望として, コンテキストを増やして記事集合と文集合の比較実験を行うことを考えている. 位置情報に関しては抽出データに対して更に分析を行う必要がある. また Retriever-Reader モデルにおいて FiD の出力精度を向上させるために DPR で抽出する情報の性質を検証する必要がある.

謝辞

今回研究するにあたり，町田秀輔氏には実験の準備やコードのエラーの解消に助言を頂いたこと深く感謝申し上げます。本研究の一部は JSPS 科研費 JP18K11449 の助成を受けたものです.

参考文献

[1] 鈴木正敏, 鈴木潤, 松田耕史, 西田京介, 井之上直也. Jaqket: クイズを題材にした日本語 qa データセットの構築. 言語処理学会 第 26 回年次大会 発表論文集, pp. 237–240, 2020.

[2] Ai 王 クイズ日本一決定戦 , （2022-9 閲覧）. https://sites.google.com/view/project-aio/competition3.

[3] 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, 2021.

[4] 加藤拓真, 宮脇峻平, 西田京介, 鈴木潤. オープンドメイン qa における dpr の有効性検証. 言語処理学会 第 27 回年次大会 発表論文集, pp. 1403–1407, 2021.

[5] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In **Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics**, pp. 874–880, 2021.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert:pre-training of deep bidirectional transformers for language understanding. In **In NAACL-HLT**, 2019.

[7] Denis Rothman, 黒川利明 (訳). Transformer による自然言語処理. 朝倉書店, 2022.