オープンドメイン質問応答における解答可能性判別の役割

鈴木 正敏 [1,2]　松田 耕史 [2,1]　大内 啓樹 [2,1]　鈴木 潤 [1,2]　乾 健太郎 [1,2]
[1] 東北大学　[2] 理化学研究所
{m.suzuki,matsuda,jun.suzuki,inui}@ecei.tohoku.ac.jp,　hiroki.ouchi@riken.jp

1 はじめに

自然言語による質問の入力に対して適切な解答を出力する質問応答システムは，自然言語処理技術の最も重要な応用課題の 1 つである [1]．特に，対象とする知識の範疇を限定しないオープンドメイン質問応答は，「どんな質問にも答えられるコンピュータ」を実現する挑戦的な課題として，学術界と産業界の両方で積極的な研究が行われている [2]．

既存のオープンドメイン質問応答システムの多くは，大量の文書に対する情報検索と，検索された文書から解答を抽出する処理から成るパイプライン型の手法を採用している [3, 4, 5]．近年では，機械読解タスクの隆盛を背景に，深層学習を基盤とする読解モデルを用いて解答抽出を行う質問応答システムも提案されている [6, 7, 8, 9, 10, 11]．

情報検索を基盤としたパイプライン型の質問応答システムでは，検索された文書が質問の内容と無関係であった場合，文書からの正しい解答の抽出が困難になる．この問題に対処するための方法として，検索された文書や解答候補のリランキングを行う手法 [7, 8] などが提案されている．

しかし本来，与えられた文書が質問に解答するための情報として適切かどうかの判別には，文書の読解と同等の言語理解能力が必要となるはずである．したがって，読解のための文書の選別も，別個のモデルを用意するのではなく，読解モデルに行わせるのが合理的ではないかと考えられる．

そこで本研究では，解答抽出を担う読解モデルに文書の読解による解答可能性の判別を行わせることで，解答性能の高いオープンドメイン質問応答システムを実現する方法を研究する．具体的には，解答可能性が明示的に付与されたデータセット [12] を用いて読解モデルを訓練し，質問との関連度が低い文書を「読み捨てる」ことで，読解で得られる解答候補の質の向上を図る．実験の結果，解答可能性判別を行うよう訓練された読解モデルを用いることで，最終的な質問応答の精度が大幅に向上した．さらに，解答可能性の教師信号として，文書中の正解文字列の有無ではなく，文書の読解によって質問に解答できるかどうかの人手による判断を用いることで，より高い質問応答の性能が得られた[1)]．

表 1　解答可能性付き読解データセットに含まれる事例．

質問: 地元では「がめ煮」とよぶ、鶏肉や野菜を一口大に切って、だしや醤油などで煮込んだ福岡県の郷土料理は何でしょう?
正解: 筑前煮

文書: **筑前煮**など、ほかの煮物にも鰹節を出汁のひとつとして使うものは多いが、土佐煮では鰹節の濃厚な味を生かしたり、具材の一部として使う特徴がある。土佐酢などと同様、鰹節を用いることから鰹で有名な土佐（現高知県）の名を用いる。...
解答可能性スコア: 0

2 関連研究

オープンドメイン質問応答の初期のシステムは，質問文の解析，情報検索，解答候補の抽出などから成るパイプライン処理を，人手で設計された規則や特徴量に基づいて行うものが主流であった [3, 4, 5]．近年では，深層学習技術の進展と同時に，SQuAD [13] に代表される機械読解タスクの研究が広く取り組まれるようになったことを背景に，従来のパイプライン型質問応答システムの一部を深層学習を基盤としたモデルで置き換え，より解答性能の高いシステムの実現を目指す研究が進められている．

Chen ら [6] は，従来の情報検索のモジュールと深層学習による読解モデルを組み合わせたオープンドメイン質問応答システムを初めて提案した．このシステムでは，TF-IDF を特徴量とした情報検索モジュールにより取得された文書（Wikipedia 記事の段落）に対して，事前に SQuAD 等のデータセットで訓練された読解モデルを適用し最も予測確率の高い解答候補を出力することで，質問応答を実現してい

1) 実験に用いたコードは https://github.com/cl-tohoku/open-book-qa で公開している．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

る．以降，読解モデルに訓練済み言語モデルを利用する方法 [10] や，検索も深層学習で行う方法 [9, 11] など，多くの手法が相次いで提案されている．

パイプライン型の質問応答システムでは，検索により取得される文書に必ずしも質問の正解が書かれているとは限らないため，そのような文書が入力されたときに読解モデルは誤った解答を出力してしまうという課題がある．この問題を回避する手法として，検索された文書に質問の正解が書かれているかどうかを判別する新たなモジュールの導入 [7] や，読解モデルが出力する解答候補をリランキングするモジュールの導入 [8] が提案されている．

これらの既存研究に対して本研究は，文書の選別には読解の能力が必要であるという着想のもと，解答可能性判別を読解モデルに行わせることでより良い質問応答システムを実現することを追究する．

3 解答可能性付き読解データセット

解答可能性付き読解データセット [12] は，日本語のクイズ問題を元に作成された，機械読解タスクのデータセットである．ウェブから収集されたクイズ問題である質問に対して，正解の文字列を含む Wikipedia 記事の段落が読解対象の文書として 1 問あたり最大 5 件付与されている．さらに，全ての質問-正解-文書の組に対して，クラウドソーシングにより付与された「解答可能性スコア」が与えられている．解答可能性スコアは，1 事例あたり 5 名の作業者に質問-正解-文書の組を提示し，質問の正解の根拠が文書中に書かれているかを Yes/No の 2 択で答えてもらい，Yes と答えた人数をそのまま 5 点満点のスコアとしたものである．

表 1 に，データセットに含まれる事例の例を示す．この事例の文書には，正解の「筑前煮」という文字列は書かれているが，解答の根拠となる「がめ煮」に関する情報は書かれていないため，解答可能性スコアは 0 が付与されている．

なお，本データセットと同時期に提案された同様のデータセットに SQuAD 2.0 [14] があるが，SQuAD 2.0 では**「文書に正解が書かれているか」**を解答可能性としているのに対し，本データセットでは**「文書に正解の根拠が書かれているか」**を解答可能性としており，正解の文字列そのものは全ての文書に書かれている．すなわち，本データセットは，読解による解答可能性を判別させるための，より難しいデータセットになっている．

表 2 データセットの統計量．

		質問-正解-文書の組数		
	質問数	解答可能	解答不可能	合計
訓練	9,691	21,091	22,519	43,610
開発	1,500	3,524	3,339	6,863
評価	1,400	3,079	3,099	6,178

本研究では既存研究 [12] と同様に，質問の元となったクイズ問題の出題年ごとにデータセットを訓練・開発・評価データに分割する．また，解答可能性スコアが 2 以上のものを解答可能な事例とし，1 以下のものを解答不可能な事例として扱う．表 2 にデータセットの各種統計量を示す．

4 提案手法

3 節のデータセットで訓練した読解モデルを用いてオープンドメイン質問応答システムを構築する．

4.1 読解モデル

読解モデルとして，日本語の Wikipedia 記事で事前訓練された BERT [15] を利用する．モデルの構造は BERT-base とし，語彙の大きさは 32,768 とする．

事前訓練された BERT を読解タスクに適応させるため，3 節で述べた解答可能性付き読解データセットを用いてモデルを追加訓練する．追加訓練には，BERT の SQuAD 2.0 に対する既存手法 [15] を応用する．すなわち，質問 q と文書 d を連結させた入力 `[CLS]` q `[SEP]` d `[SEP]` に対して，解答可能な事例に対しては，文書中の正解の開始位置 i と終了位置 j $(i \leq j)$ を予測し，解答不可能な事例に対しては，たとえ正解の文字列が偶然含まれている場合でもそれを正解と見なさずに，開始位置と終了位置として文頭（`[CLS]` の位置）を予測するように訓練する[2)]．

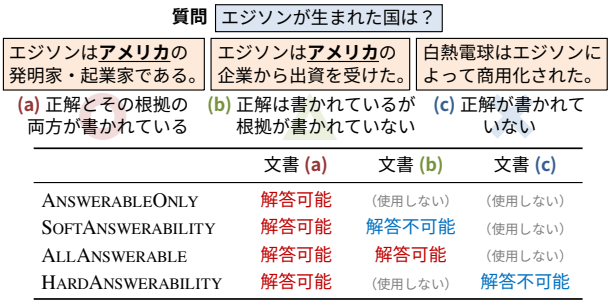
4.2 質問応答システム

4.1 節で訓練した読解モデルを用いて，オープンドメイン質問応答システムを構築する．本研究で構築する質問応答システムは，検索，読解，解答統合の 3 つのモジュールから成る．以下で，それぞれのモジュールについて説明する．

検索　システムに入力される質問に対して，全文検索エンジンを用いて予めインデックスされた文書集合から関連度の高い k 件の文書を取得する．全文検索のスコアリング関数には Okapi BM25 を用い，

2） BERT の事前訓練および追加訓練の詳細は付録 A を参照．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).



	文書 (a)	文書 (b)	文書 (c)
AnswerableOnly	解答可能	(使用しない)	(使用しない)
SoftAnswerability	解答可能	解答不可能	(使用しない)
AllAnswerable	解答可能	解答可能	(使用しない)
HardAnswerability	解答可能	(使用しない)	解答不可能

図 1　訓練時における解答可能性判別の例.

表 3　開発・評価データに対するシステムの解答性能.

		開発		評価	
	k	EM	F1	EM	F1
AnswerableOnly	12	42.7	52.9	39.1	50.8
SoftAnswerability	960	**55.1**	**65.7**	**52.0**	**64.7**
AllAnswerable	19	42.1	52.2	38.6	50.4
HardAnswerability	203	51.1	62.4	49.8	61.7

文書集合には 2020 年 8 月 30 日時点の Wikipedia 全記事を段落に分割したものを用いた[3]．

読解　システムに入力された質問と検索モジュールで取得された文書から成る k 件の質問-文書ペアのそれぞれに対して，4.1 節で訓練した読解モデルを用いて解答候補を出力する．ただし，読解モデルが解答不可能と予測した事例は棄却する．

解答統合　読解モジュールにより得られた最大 k 件の解答候補から，多数決によりシステムの最終的な解答を 1 つ決定する．ただし，最も出現数の多い解答候補が複数ある場合は，元の質問-文書ペアに対する検索モジュールの関連度スコアの最上位がより上位にある解答候補を選択する．また，読解モジュールにより得られた解答候補が 0 件である（すなわち k 件全てが解答不可能と予測された）場合は，最終的な解答も「解答不可能」とする[4]．

5　実験

5.1　実験設定

3 節のデータセットを用いて，以下の 4 つの条件で読解モデルを訓練する．

- AnswerableOnly: 訓練データの解答可能な事例のみを用いて読解モデルを訓練する．
- SoftAnswerability **（提案手法）**: 訓練データの解答可能・解答不可能な事例の両方を用いて解答可能性判別を伴う読解モデルを訓練する．
- AllAnswerable: 訓練データの全ての事例を解答可能と見なし用いて読解モデルを訓練する．すなわち，読解で解答不可能な文書に対しても文書中の正解文字列の位置を予測させる．
- HardAnswerability: 訓練データの解答不可能な文書を，正解が書かれていない別の文書に置き換えたデータセットを新たに作成し，解答可能性判別を伴う読解モデルを訓練する．これは SQuAD 2.0 における「解答不可能＝正解が書かれていない」という条件を再現したものである．解答不可能な文書の付与には 4.2 節の全文検索エンジンを用い，質問との関連度が高く，かつ正解を含まない文書を，元の解答不可能な文書と同じ数だけ付与した．

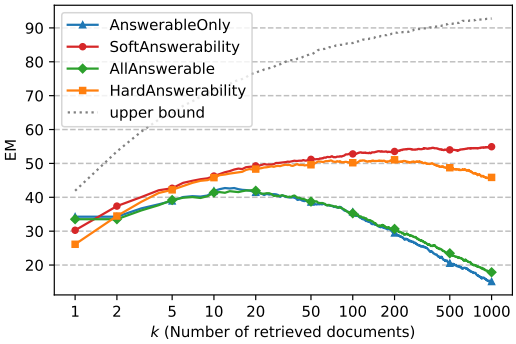


図 2　開発データでの文書数 k と解答性能 (EM) の関係.

各条件の訓練時の解答可能性判別の例を図 1 に示す．それぞれの条件で訓練された読解モデルを用いて，4.2 節で述べた質問応答システムを構築する．

システムの評価には，3 節のデータセットの開発・評価データの質問と正解を用いる．質問応答性能の定量的な評価指標として，予測された解答と正解の完全一致の割合である EM と，部分一致率の平均である F1 を測定する[5]．検索モジュールで取得する文書数は $k \in [1, 1000]$ とし，上記の 4 つそれぞれの条件で，開発データ上で EM が最大となる k を用いて評価データ上での評価を行う．

5.2　実験結果

表 3 に実験結果を示す．提案手法の SoftAnswerability では，解答可能性判別を行わない設定である AnswerableOnly および AllAnswerable と比較して EM がおよそ 13 ポイント改善し，解答

3)　すなわち，本研究における「文書」の単位は段落である．
4)　本研究の実験設定では不正解としてカウントされる．
5)　F1 は本来は単語の部分一致率を計算するが，本研究では日本語を対象にするため文字レベルの部分一致率を計算する．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

表 4 開発データの質問に対するシステムの解答例.
質問: 現在日本に生息する 2 種類のキツネは、キタキツネと何でしょう? **正解:** ホンドギツネ

文書（検索上位 3 件）	システムの解答		
	AnswerableOnly	AllAnswerable	SoftAnswerability
山腹にはキタキツネ、エゾクロテン、エゾリス、エゾシマリス、エゾモモンガ、エゾユキウサギなどの哺乳類が生息しており、130 種類以上の野鳥がいることも確認されている。	エゾクロテン	エゾクロテン	（解答不可能）
2008 年 11 月開園。園名の通り、フクロウなどの猛禽類とキタキツネなどのキツネ類を中心に展示する動物園で、キタキツネへのエサやりや抱っこ、フクロウやタカの腕乗せやフライトショーをメインとしていた。...	フクロウ	フクロウ	（解答不可能）
日本に生息する**ホンドギツネ**とキタキツネを比較すると、ホンドギツネの方が毛色がより暗褐色で体長がやや小さい。...	ホンドギツネ	ホンドギツネ	ホンドギツネ

可能性判別の有効性を示した．また提案手法は，SQuAD 2.0 のように正解文字列の有無を判別する HardAnswerability と比較しても，EM が 2 から 4 ポイント高い結果となった．この結果は，人間の判断を元に付与された解答可能性の情報が，正解の記述有無を元に機械的に付与された情報よりも，データ作成のコストはかかる一方，より効果的に質問応答に応用できる可能性を示している．

図 2 に，開発データにおける，検索モジュールで取得する文書数 k とシステムの解答性能 (EM) の関係を示す．図 2 の upper bound は，正解を含む文書が k 件中 1 つでもある質問の割合，すなわち EM の上界を示している[6]．解答可能性判別を行わない AnswerableOnly と AllAnswerable では，$k = 10$ から 20 の範囲で EM が最大になり，それ以上 k を大きくしていくと EM は低下した．これとは対照的に，提案手法の SoftAnswerability では，文書数 k を大きくしていくと，性能はより向上し，今回の実験の範囲では $k = 960$ で EM が最大となった[7]．

文書検索を基盤とした質問応答システムでは一般に，文書数 k を大きくすると質問の正解を含んだ文書が検索でヒットする確率は高くなるが，関連度が低い文書がより多く取得される可能性も高くなる．解答可能性判別を行わない読解モデルは，任意の入力に対して何かしら解答を抜き出そうとするため，k が大きくなり解答不可能な文書の入力が増えるに従い，誤った解答候補の出力が増え，解答統合時のノイズになったと考えられる．一方，提案手法では，k が大きくなり解答不可能な文書が増えた場合でも，解答不可能と判断された文書は棄却することができ，かつ検索結果の下位にある解答可能な文書に対しては読解を行えるため，結果として得られる解答候補の集合の質が向上し，多数決による解答統合に効果的であったと考えられる．

6) 正解が書かれていても読解で解答できるとは限らないため，実際の EM の上限はこれ以下となる．
7) 文書数 k をより大きくすることで更なる解答性能の向上が期待できるが，k に比例する実行時間とのトレードオフが生じる．例えば $k = 1,000$ では，開発データの 1,500 問に対する予測に約 6 時間を要した（NVIDIA V100 を 1 コア使用）．

5.3 システムの解答例

表 4 に，開発データの質問に対するシステムの解答例を示す．表 4 に示した検索上位 3 件の文書のうち，質問の正解である「ホンドギツネ」を含むものは 3 番目の文書だけである．解答可能性判別を行わない AnswerableOnly と AllAnswerable のシステムでは，正解が解答不可能な文書に対しても解答抽出を行い，その結果誤った解答候補を出力している．一方，解答可能性判別を行う SoftAnswerability のシステムでは，解答不可能な文書に対しては正しく「解答不可能」と出力し，正解を解答可能な文書に対してのみ解答抽出を行っていることがわかる．

6 おわりに

本研究では，解答可能性の判別を行う読解モデルを応用してオープンドメイン質問応答システムを構築した．実験により，人手で付与された読解による解答可能性の情報が文書の選別に有効であることを示す結果が得られた．

今後の課題として，低コストで作成された解答不可能な訓練事例をより効果的に活用する方法や，DPR [11] のような最先端の情報検索手法と組み合わせる研究に取り組む．

謝辞 本研究は JSPS 科研費 JP19H04425, JP19J13238 の助成を受けたものである．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

参考文献

[1] R. F. Simmons. Answering English questions by computer: a survey. *Communications of the ACM*, Vol. 8, No. 1, pp. 53–70, 1965.

[2] D. A. Ferrucci. Introduction to “This is Watson”. *IBM Journal of Research and Development*, Vol. 56, No. 3.4, pp. 1:1–1:15, 2012.

[3] Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum, and Vasile Rus. The Structure and Performance of an Open-Domain Question Answering System. In *ACL*, pp. 563–570, 2000.

[4] Eric Brill, Susan Dumais, and Michele Banko. An Analysis of the AskMSR Question-Answering System. In *EMNLP*, pp. 257–264, 2002.

[5] John Prager. Open-Domain Question–Answering. *Foundations and Trends® in Information Retrieval*, Vol. 1, No. 2, pp. 91–231, 2007. Publisher: Now Publishers, Inc.

[6] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to Answer Open-Domain Questions. In *ACL*, Vol. 1, pp. 1870–1879, 2017.

[7] Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. Denoising Distantly Supervised Open-Domain Question Answering. In *ACL*, Vol. 1, pp. 1736–1745, 2018.

[8] Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. Evidence Aggregation for Answer Re-Ranking in Open-Domain Question Answering. In *ICLR*, 2018.

[9] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *ACL*, pp. 6086–6096, 2019.

[10] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-End Open-Domain Question Answering with BERTserini. In *NAACL*, Vol. Demonstrations, pp. 72–77, 2019.

[11] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wentau Yih. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP*, pp. 6769–6781, 2020.

[12] 鈴木 正敏, 松田 耕史, 岡崎 直観, 乾 健太郎. 読解による解答可能性を付与した質問応答データセットの構築. 言語処理学会 第 24 回年次大会, pp. 702–705, 2018.

[13] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *EMNLP*, pp. 2383–2392, 2016.

[14] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know What You Don’t Know: Unanswerable Questions for SQuAD. In *ACL*, Vol. 2, pp. 784–789, 2018.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, Vol. 1, pp. 4171–4186, 2019.

[16] 伝 康晴, 小木曽 智信, 小椋 秀樹, 山田 篤, 峯松 信明, 内元 清貴, 小磯 花絵. コーパス日本語学のための言語資源：形態素解析用電子化辞書の開発とその応用. 日本語科学, Vol. 22, pp. 101–123, 2007.

[17] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019.

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

A　読解モデルの訓練の詳細

BERT の事前訓練には，日本語版 Wikipedia の 2020 年 8 月 30 日付けの Cirrusearch ダンプデータ[8)]から抽出した記事本文データを利用した．テキストの前処理として，単語分割を Unidic 2.1.2 [16] を用いて行い，さらに BERT の WordPiece によるサブワード分割を行った．語彙の大きさは 32,768 とした．1 訓練事例あたりの最大トークン数は 512 とし，ミニバッチのサイズは 256 とした．また，masked language model における whole word masking を適用した．最適化アルゴリズムには AdamW [17] を用い，学習率は 1e-4 とした．訓練のステップ数は 1,000,000 とし，最初の 10,000 ステップにおける学習率の warmup と以降の linear decay を適用した．BERT の事前訓練には，TensorFlow Research Cloud プログラム[9)]で提供された Cloud TPU v3-8 を使用し，訓練の完了にはおよそ 5 日間を要した．

BERT の読解タスク向けの追加訓練では，1 訓練事例あたりの最大トークン数を 512 とし，ミニバッチのサイズを 8，gradient accumuration のステップ数を 4 とした．最適化アルゴリズムには AdamW を用い，学習率は 5e-5 とした．訓練のエポック数は 3 とし，開発データで読解性能 (EM) が最大となったエポックのモデルを質問応答システムの実験に用いた．BERT の追加訓練には，NVIDIA V100 GPU を 1 コア使用し，訓練の完了には約 1 時間を要した．

8） https://dumps.wikimedia.org/other/cirrussearch/
9） https://www.tensorflow.org/tfrc

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).