単語埋め込みを用いた正則化による言語モデルの追加事前学習

西田光甫　　西田京介　　吉田仙
日本電信電話株式会社　NTT メディアインテリジェンス研究所
{ kosuke.nishida.ap, kyosuke.nishida.rx, sen.yoshida.tu }@hco.ntt.co.jp

1　はじめに

膨大なコーパスと計算機を利用して学習する事前学習済み言語モデルが，自然言語処理タスクで高い性能を発揮している [1]．これらのモデルは，広範な話題を含むコーパスで言語モデルの事前学習をすることで，言わば「テキストの読み方の理解」を獲得する．

一方で，機械学習の一部の問題にはドメイン依存性が存在することが知られている．事前学習済み言語モデルもドメイン依存性を持ち，一般的なドメインで事前学習を行なったモデルは，目的ドメインでの膨大な事前学習を追加することでさらに高い性能を発揮する [2]．また，Task-Adaptive PreTraining (TAPT) と呼ばれる手法では，目的タスクの教師データに含まれるテキストを用いた事前学習を追加するだけで精度が向上することを示した [3]．目的タスクのテキストを用いた事前学習は，通常の事前学習に比べて遥かに短い時間で実行可能である．これらの手法は，事前学習を追加することでドメイン適応が可能であることを示している．

本研究では，単語埋め込みを利用したドメイン適応に注目する．背景には，事前学習済み言語モデルは既にテキストの読み方を理解しているため，ドメインにおける単語の意味を理解することでドメイン適応が可能なのではないか，という発想がある．

我々は **単語埋め込みを用いた正則化による言語モデルの追加事前学習** を含む新たな Fine-Tuning 手法を提案する．提案手法ではまず，単語埋め込み学習手法である fastText [4] を用いて目的タスクのテキストから目的ドメインの単語埋め込みを獲得する．次に，言語モデルの追加事前学習に並行して目的ドメインの単語埋め込みに言語モデルの単語埋め込みを近づける学習を行う．最後に通常の Fine-Tuning を行う．提案手法は，TAPT 同様に少ないデータ数での事前学習を追加する手法であるため，短い計算時間で学習できる．また，単語埋め込みは事前学習済み言語モデルに比べて浅いモデルから学習できるため，少リソースでの学習に適していると考えられる．fastText を利用することで，言語モデルが採用しているサブワードレベルの語彙に対しても単語埋め込みを得ることができる．本稿では，医療ドメインと Wikipedia ドメインの機械読解タスク（BioASQ [5] および SQuAD 1.1 [6]）において，提案手法が通常の Fine-Tuning 及び TAPT を利用した Fine-Tuning を上回る精度を達成することを示す．

2　準備

2.1　事前学習済み言語モデル

本研究では事前学習済み言語モデルの単語埋め込みに注目する．語彙を V_{LM} とし，モデルに入力されるトークン系列を $X \in V_{\mathrm{LM}}^{l}$ とする．ただし l はトークン系列長である．事前学習済み言語モデルの埋め込み層は，パラメータとして単語埋め込み行列 $E \in \mathbb{R}^{V_{\mathrm{LM}} \times d_{\mathrm{LM}}}$ を持つ．ただし，d_{LM} は埋め込み次元である．このとき，i 番目の入力トークン x_i の単語埋め込みは E_{x_i} である．

事前学習済み言語モデルの語彙 V_{LM} は，例えば BERT では，WordPiece [7] によって作成された 30,000 語である．WordPiece ではトークンの区切りとしてサブワードを採用しており，例えば ‘tokenizing’ は ‘token’ と ‘##izing’ に分割される．WordPiece では全サブワードが 1 文字の初期状態から尤度が高くなるようにサブワードを連結していき，規定の語彙数になるまで語彙を増やす．

2.2　fastText

fastText は，サブワードの情報を用いて単語埋め込みを計算する手法である [4]. fastText の skipgram [8] では，単語 x_i から周辺語 x_j　$(j \in C_i)$ を予測するタスクを解くことで単語埋め込みを学習する（C_i は任意に窓幅を指定可能な周辺インデックスの集合）．具体的には，位置 i の単語について，学習コーパス

における全ての周辺語を正例とし，語彙 V_{FT} からランダムにサンプリングした負例 $\mathcal{N}_i$ を用いて，下記の目的関数の最大化を行う:

$$\sum_i \left\{ \sum_{j \in C_i} \log(1 + e^{-s(x_i, x_j)}) + \sum_{x \in \mathcal{N}_i} \log(1 + e^{s(x_i, x)}) \right\}.$$

すなわち，正例に対しては x_i, x_j のスコア $s(x_i, x_j)$ が大きく，負例に対してはスコアが低くなるような学習を行う．

fastText はスコア関数 s をモデリングするために単語 $v \in V_{\mathrm{FT}}$ に対して，v の部分文字列（サブワード）の集合 S_v を考慮する特徴を持つ．入力側の単語 x_i と出力側の単語 x_j のスコアは

$$s(x_i, x_j) = \sum_{w \in S_{x_i}} W_{\mathrm{in},w}^{\top} W_{\mathrm{out},x_j}$$

と表される．ここで，$W_{\mathrm{in}} \in \mathbb{R}^{N \times d_{\mathrm{FT}}}$ は入力側の単語埋め込み，$W_{\mathrm{out}} \in \mathbb{R}^{V_{\mathrm{FT}} \times d_{\mathrm{FT}}}$ は出力側の単語埋め込みである．また，d_{FT} は埋め込み次元，N はサブワードの語彙数に関わる任意の大きな数である．なお fastText の実装では，考慮する部分文字列の数には制限を設けず，指定した長さ（デフォルト値は 3 文字以上 6 文字以内）の部分文字列を全て考慮する．部分文字列 w は N 以下（デフォルト値は 2,000,000）のインデックスにハッシュ化されることで埋め込み $W_{\mathrm{in},w}$ を得る．

以上の学習手法により，fastText はサブワードを考慮した単語埋め込み W_{in} を得る．推論時は，$\sum_{w \in S_v} W_{\mathrm{in},w}$ を単語 v の埋め込みとする．fastText はサブワードの情報を利用した学習によって，他の手法よりも少ないデータ数で単語類似度を学習することが報告されている [4].

3 提案手法

提案手法は，(1) fastText の追加学習，(2) 事前学習済み言語モデルの追加事前学習，(3) Fine-Tuning の 3 段階の手順を行う．

fastText の追加学習　公開されている fastText[1] の埋め込み W_{in} を初期値として，目的のタスクの訓練データをコーパスとした fastText の学習を行い，新たな fastText の埋め込みを得る．得られた fastText の埋め込みから，事前学習済み言語モデルの語彙 V_{LM} の埋め込み $F \in \mathbb{R}^{V_{\mathrm{LM}} \times d_{\mathrm{FT}}}$ を推論する．他の単語埋め込み手法と異なり fastText はサブワードの埋め込みを保持しているため，サブワード分割を採用している事前学習済み言語モデルの語彙の埋め込み行列を得ることができる[2].

1） https://fasttext.cc/

表 1　各データセットの統計値．訓練・評価データは下流タスクのデータとしてのサンプル数である．訓練テキストは追加事前学習のデータとしての単語数である．

	訓練データ	評価データ	訓練テキスト
SQuAD1.1	87599	10570	2.62M
BioASQ5	4950	150	1.38M

事前学習済み言語モデルの追加事前学習　モデルに入力されるトークン系列を $X \in V_{\mathrm{LM}}^{l}$ とする．追加事前学習時は，言語モデルの損失関数 $L_{\mathrm{LM}}(X)$ に L^2 ノルム正則化を加えた学習を行う．つまり，損失関数は正則化の対象トークンの集合を $R(X)$ として

$$L_{\mathrm{LM}}(X) + \frac{1}{|R(X)|} \sum_{x_i \in R(X)} \| f(E_{x_i}) - F_{x_i} \|_2^2$$

である．正則化の対象 $R(X)$ は，ランダムに選んだ全体の 50%のトークンから，ストップワードと fastText で考慮する最短部分文字列長を下回る長さのサブワードを除いたトークンである．

埋め込みの写像 f は d_{LM} 次元ベクトルを d_{FT} 次元ベクトルに写す関数

$$f(z) = \mathrm{LN}(W_f z + b_f)$$

である．LN は Layer Normalization [9] である．$W_f \in \mathbb{R}^{d_{\mathrm{FT}} \times d_{\mathrm{LM}}}, b_f \in \mathbb{R}^{d_{\mathrm{FT}}}$ は学習可能なパラメータである．

目的関数は，第一項によって言語モデルの忘却を抑えつつ第二項によって単語埋め込みのドメイン適応を行うように設計されている．以上の学習を，目的のタスクの訓練データをコーパスとして行う．

Fine-Tuning　Fine-Tuning では，追加事前学習で得られたモデルを初期値として通常の Fine-Tuning を行う．

4 評価

4.1 データセット

提案手法を 2 種の機械読解データセットで評価した．各データセットの統計値を表 1 に示す．実験設定を付録に示す．

SQuAD　SQuAD1.1 は Wikipedia の 1 段落を参照して質問に回答するタスクである [6]．予め 1 つの質問には 1 つの参照テキストが紐づけられている．

2） サブワードの長さが fastText で考慮する最短部分文字列長を下回る場合は，サブワードの埋め込みを得ることができない．提案手法では追加事前学習時にそのようなサブワードを正則化の対象外とする．

入力は質問と参照テキストを連結したトークン系列である．回答は参照テキストに文字列として含まれているため，区間を抽出することで回答とする．出力は回答の始点と終点のインデックスであり，言語モデルに 2 次元への線形変換層を追加して Fine-Tuning することで回答区間を訓練・予測する．評価指標は完全一致（EM）と部分一致（F1）である．

BioASQ　BioASQ5 は医療ドメインの質問応答データセットである [5]．先行研究 [2] に従い，ファクトイド型の質問を SQuAD と同様の形式に前処理して評価した．評価指標の Strict Accuracy (SACC), Lenient Accuracy (LACC), Mean Reciprocal Rank (MRR) はそれぞれトップ 1 の予測の正解率，トップ 100 の予測に正解が含まれる率，正解の順位の逆数の平均である．訓練をランダムなシードで 5 回行い，それらの結果の平均を用いた．本データセットは訓練データが少ないため，SQuAD を使った Fine-Tuning の後に BioASQ を使った Fine-Tuning を行う 2 段階の Fine-Tuning を行った [10].

4.2　比較手法

本稿では，事前学習済み言語モデルとして，BERT-base-cased [1]，BioBERT [2]，RoBERTa-base [11] を用いた．BERT-base-cased モデルは English Wikipedia（25 億語）と BookCorpus [12]（8 億語）をコーパスとして事前学習している．BioBERT は BERT-base-cased モデルを初期値として，更に PubMed の概要と PMC の論文から成る計 180 億語のコーパスで事前学習したモデルである．RoBERTa は BERT-base-cased モデルよりも広範な話題を含む 160GB のテキストから事前学習をしている．

これらの事前学習済み言語モデルに対して，(1) そのまま Fine-Tuning をする手法，(2)TAPT，つまり正則化項なしの追加事前学習の後 Fine-Tuning を行う手法，(3) 提案手法の 3 つの学習手法を比較した．

4.3　実験結果

提案手法は医療ドメインへの適応において有効か？　BioASQ タスクの結果を表 2 に示す．

BERT-base-cased モデルは事前学習中に医療ドメインのテキストでの学習を行わない．そのため，医療ドメインへのドメイン適応における性能を BERT-base-cased モデルを用いて評価できる．

BERT-base-cased モデルでは，提案手法によって SACC/LACC/MRR で 3.80/0.29/2.39 ポイントの精度向上が確認できた．これは，TAPT による精度向上 0.86/0.16/0.55 ポイントを大幅に上回る値である．追加事前学習がドメイン適応において効果的であることとともに，単語埋め込みの正則化によってドメインへの適応が更に進むことが考えられる．

表 2　BioASQ の開発データによる評価結果．数字に付与した記号は，提案手法と TAPT の 5 回の結果について paired t-test を実施した結果を示す（*: $p < .05$）

	SACC	LACC	MRR
BERT-base-cased	37.67	53.44	43.42
+TAPT	38.53	53.60	43.97
+Proposed	**41.47***	**53.73**	**45.81***
BioBERT	44.00	**59.20**	49.91
+TAPT	**44.13**	58.93	**50.05**
+Proposed	44.00	58.13	49.56

また，各手法で学習されたモデルの単語埋め込みを t-SNE で描画した結果を付録に示す．提案手法は訓練テキスト数としては TAPT と同じであるにも関わらず，膨大な医療ドメインのコーパスで学習した BioBERT に近い単語埋め込みを学習していることが示唆されている．

提案手法は目的ドメインで事前学習した言語モデルにおいて有効か？　BioBERT モデルは潤沢な医療ドメインのテキストで事前学習をした言語モデルである．BioBERT からの追加事前学習では，TAPT で一部指標が僅かに向上するものの，全体として性能が大きく向上することはなかった．これは，目的タスクのテキストを使う追加事前学習によって学習できる知識を，BioBERT が既に保持しているためと考えられる．

[3] は，TAPT が目的ドメインに適応した言語モデルの性能を更に向上することを指摘している．この目的ドメインに適応した言語モデルとは，RoBERTa を初期値として，目的ドメインで収集したコーパスで追加事前学習を行なったモデルである．しかし，BioBERT では，彼らが用いたよりも大きな 180 億語のコーパスで事前学習を行なっている．ドメイン適応済みの言語モデルに対する目的タスクのテキストによる追加事前学習の効果の有無は，ドメイン適応時のコーパス量，目的タスクのテキスト数，モデルサイズなどの条件に依存していると考えられる．

提案手法は一般的なドメインのタスクに対しても有効か？　SQuAD タスクの評価結果を表 3 に示す．SQuAD では，BERT-base-cased モデルからの追加事

表 3　SQuAD の開発データによる評価結果.

	EM	F1
BERT-base-cased	**79.12**	**87.55**
+TAPT	79.07	87.48
+Proposed	78.49	87.07
RoBERTa-base	82.76	90.40
+TAPT	83.01	90.45
+Proposed	**83.55**	**90.86**

前学習を行う場合，TAPT と提案手法でともに精度の向上がなかった．これは，BioBERT からの追加事前学習が医療ドメインで効果がなかったことと同じ理由で説明できる．

RoBERTa モデルからの追加事前学習を行う場合，提案手法によって EM/F1 で 0.79/0.46 ポイントの精度向上が確認できた．これは，TAPT による精度向上 0.25/0.05 ポイントを上回る値であり，全比較手法の中で最高の精度である．

SQuAD は Wikipedia 記事に関する質問応答をするタスクである．BERT-base-cased は事前学習コーパスの多くが Wikipedia であるため，追加事前学習の効果がなかった．しかし，RoBERTa は広範な話題をカバーするために更に多くのコーパスから事前学習を行っている．そのため，Wikipedia が事前学習中に利用されているにも関わらず，追加の事前学習によって Wikipedia 単独に適応することが精度向上に繋がったと考えられる．

事前学習のデータ数を増やすことで事前学習済み言語モデルの性能は向上する傾向にあることが知られている [13, 14]．本実験は，多くのデータで事前学習した言語モデルに対して提案手法を適用することで，事前学習中に含まれるドメインのタスクであっても更に性能が向上することを示している．

提案手法は異なる言語モデルやサブワード分割法に対しても有効か？　BERT-base-cased モデルは WordPiece によってサブワード分割を行なっているが，RoBERTa は単語分割を行わずにバイトレベルでのサブワード分割を行う．提案手法は表 2 では BERT-base-cased モデル，表 3 では RoBERTa モデルを用いてそれぞれ精度向上を確認しており，言語モデルやサブワード分割法に依存せずに効果があると考えられる．主要なサブワード分割法には他に SentencePiece [15] があり，さらなる検討が望ましい．

追加事前学習で言語モデルと単語埋め込みはどのように学習が進むか？　BioASQ タスクにおいて提案手法が BERT-base-cased モデルの追加事前学習を行なった際の学習曲線を図 1 に示す．学習は単語埋め込みに関する第二項が先行する形で進むことが観察できる．また，言語モデルに関する第一項は推論時の減少が途中で止まり，以降は訓練データのみでの減少となる．一方，単語埋め込みに関しては訓練・推論時の乖離が起きずに学習が進む．

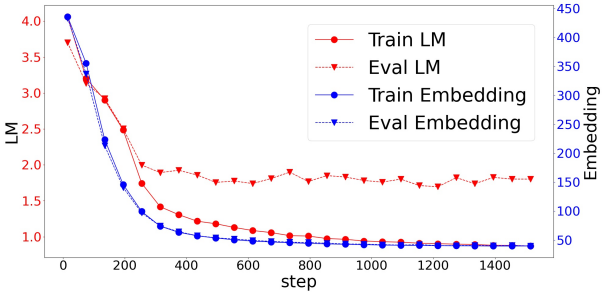

図 1　学習曲線．左の軸は損失関数第一項，右の軸は損失関数第二項の値を示す．

5　おわりに

本研究では目的タスクの教師データをコーパスとしたドメイン適応に取り組んだ．本研究の貢献を以下に示す．

本研究の独自性　fastText を用いて目的ドメインにおける単語埋め込みを獲得し，単語埋め込みに関する L^2 ノルム正則化を加えた事前学習を Fine-Tuning 前に追加する新たな Fine-Tuning 手順を提案した．提案手法は，言語モデルの忘却を抑えつつ言語モデルの単語埋め込みを目的ドメインにおける単語埋め込みに近づけることで，目的ドメインにおける単語の意味を獲得する効果を持つ．

本研究の重要性　医療ドメインへのドメイン適応において，通常の Fine-Tuning 及び言語モデルの事前学習のみを追加する Fine-Tuning 手法に比べて提案手法が高い精度を示すことを確認した．また，Wikipedia ドメインにおいても広範な話題を含むコーパスで事前学習した RoBERTa モデルの性能を提案手法が向上させることを確認した．

近年，より多くのデータから学習したより多くのパラメータを持つ事前学習済み言語モデルの公開が続いている [16]．本研究で提案した追加事前学習を含む新たな Fine-Tuning 手法は，そのような多くのデータで学習した言語モデルに対して，目的ドメインの知識を与える重要な手法であると考えられる．

参考文献

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL-HLT*, pp. 4171–4186, 2019.

[2] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, Vol. 36, No. 4, pp. 1234–1240, 2020.

[3] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *ACL*, pp. 8342–8360, July 2020.

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, Vol. 5, pp. 135–146, 2017.

[5] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artieres, Axel Ngonga, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, Vol. 16, p. 138, 2015.

[6] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *EMNLP*, pp. 2383–2392, 2016.

[7] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *NIPS*, Vol. 26, pp. 3111–3119, 2013.

[9] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[10] Georg Wiese, Dirk Weissenborn, and Mariana Neves. Neural domain adaptation for biomedical question answering. In *CoNLL 2017*, pp. 281–289, 2017.

[11] 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. *arXiv preprint arXiv:1907.11692*, 2019.

[12] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, pp. 19–27, 2015.

[13] Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. Cloze-driven pretraining of self-attention networks. In *EMNLP-IJCNLP*, pp. 5360–5369, 2019.

[14] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*, 2019.

[15] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *EMNLP: System Demonstrations*, pp. 66–71, 2018.

[16] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NIPS*, pp. 8024–8035. 2019.

[19] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *EMNLP: System Demonstrations*, pp. 38–45, 2020.

[20] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 2009.

[21] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python, 2020.

A 付録

実験設定 実験は NVIDIA GeForce GTX 1080Ti (11GB) 1 枚で行った．最適化手法は Adam [17] を用いた．実験で用いたハイパーパラメータを表 4 と表 5 に示す．実装は PyTorch [18] 及び Transformers [19] を，ストップワードは NLTK [20] を，単語ベースのトークナイザは spaCy [21] を用いた．

表 4 言語モデルの訓練に用いたハイパーパラメータ．スラッシュで区切られているのものは SQuAD/BioASQ の 1 段階目/BioASQ の 2 段階目である．

	追加事前学習	Fine-Tuning
batch size	256	128
epochs	100	5 / 2 / 10
max sequence length	512	384
max query length	–	64
learning rate	5e-5	
warmup proportion	0.06	
weight decay	0.01	

表 5 fastText の訓練に用いたハイパーパラメータ．記載の変数以外はデフォルト値を用いた．

	SQuAD	BioASQ
min count	5	2
epochs	5	10
dim	300	

追加事前学習によって単語埋め込みは変化しているか？ 各モデルの単語埋め込みを t-SNE によって描画した結果を示す．図 2 に BERT-base-cased モデル，図 3 に BioASQ の追加事前学習を TAPT で行なったモデル，図 4 に BioASQ の追加事前学習を提案手法で行なったモデル，図 5 に BioBERT モデルの単語埋め込み層を示す．描画の際は，BioASQ で頻出のサブワードから，ストップワード・長さが 3 以下のサブワードを除き，上位 1000 語で t-SNE モデルを作成した後に上位 30 語を描画した．

図中では，‘cell’, ‘cells’, ‘gene’, ‘genes’, ‘protein’ の 5 単語を赤文字で記している．これらの単語は医療ドメインに特有の物質名であるが，BERT-base-cased モデルと TAPT の学習結果では大きく離れた箇所に位置している．一方，提案手法と BioBERT モデルでは近い箇所に集まっている．提案手法は訓練データ数としては TAPT と同じであるにも関わらず，膨大な医療ドメインのコーパスで学習した BioBERT に近い単語埋め込みを学習していることが示唆されている．

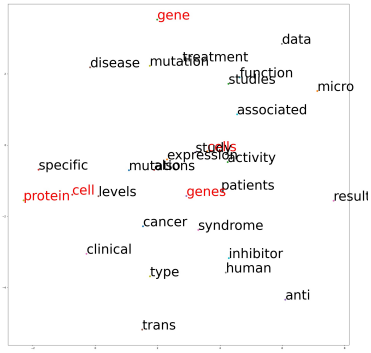

図 2 BERT-base-cased の埋め込み層

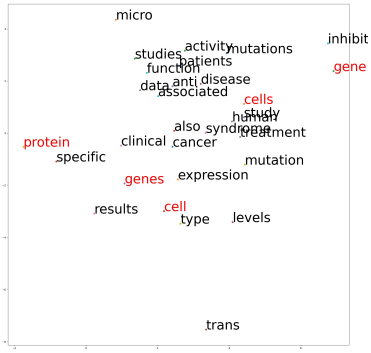

図 3 TAPT の学習結果の埋め込み層

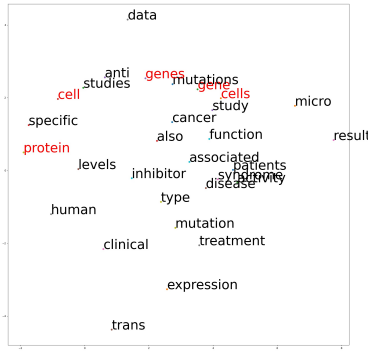

図 4 提案手法の学習結果の埋め込み層

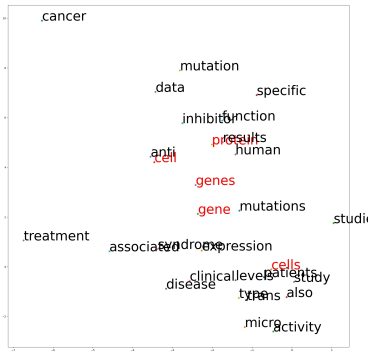

図 5 BioBERT の埋め込み層