# 事前学習済みの分散表現は表層的な知識を獲得しているか

平岡 達也 [1,2]　岡崎 直観 [2]
[1] 富士通株式会社 [2] 東京工業大学
hiraoka.tatsuya@fujitsu.com
okazaki@c.titech.ac.jp

## 概要

本研究では，事前学習済みのサブワード・単語の分散表現が表層的な知識（文字数や部分文字列）を獲得しているかを 3 つの観点から検証する．英語を主に学習した 9 つの分散表現について実験を行い，これらが文字数や部分文字列に関する情報はある程度捉えているが，どの文字が何番目に出現するかといった順序の情報は獲得できていないことを示す．

## 1　はじめに

自然言語処理の様々なタスクにおいて，事前学習済みの大規模言語モデル（LLM）を活用した手法が研究されている．LLM を用いた手法は，テキストや単語などのまとまった単位での生成タスクで高い性能を発揮することが知られている [1, 2, 3, 4].

しかしながら，LLM は入力テキストに含まれる単語や文字を数えたり，部分文字列を認識したり，回文を作成したりといった，テキストの表層的な情報を理解する必要があるタスクを苦手としているようである．実際に，性能が良いとされている ChatGPT (GPT-3.5) に対して単語の文字数や $N$ 番目の文字を問うた場合，英語であっても長い単語については正しい回答が得られないことが多く，日本語では短い単語でも正しい回答が得られない (表 1)．これは，文字数を指定した要約や抽出型の質問応答などの自然言語処理のタスクを解く際の課題になり得る．

この問題点は，一般的な自然言語処理の手法がサブワード単位での分散表現を入力としていることに起因すると考えられる．一方で，LLM によって学習されたサブワード単位の分散表現や，それらをエンコードして得られるテキストのベクトル表現が，どのような表層的な情報を獲得できているかについての検証は少なく [5]，知識の蓄積に乏しい．

そこで本稿では，3 つの観点から LLM が学習した分散表現が表層的な情報を獲得できているかを検証する．英語での検証より，サブワード・単語の分散表現は表層的な長さと部分文字列に関する知識を部分的に獲得しているが，構成文字の出現位置についての情報は獲得していないことが分かった．また，LLM が表層的な知識を苦手とする原因として，出力部分でのボトルネックの存在が示唆された．

**表 1**　単語の表層的な知識を Zero-shot で問うた場合の GPT-3.5 の出力例 (2024 年 1 月 1 日時点).

| | |
|---|---|
| 入 | How many letters is “anthropologist”? |
| 出 | The word “anthropologist” has 13 letters. |
| 入 | What is the seventh letter of “anthropologist”? |
| 出 | The seventh letter of “anthropologist” is “o.” |
| 入 | 「人類学者」は何文字の単語ですか？ |
| 出 | 「人類学者」は、日本語で 5 文字の単語です。 |
| 入 | 「人類学者」の 3 文字目は何ですか？ |
| 出 | 「人類学者」の 3 文字目は「者」です。 |

## 2　検証の概要と対象

### 2.1　検証の概要

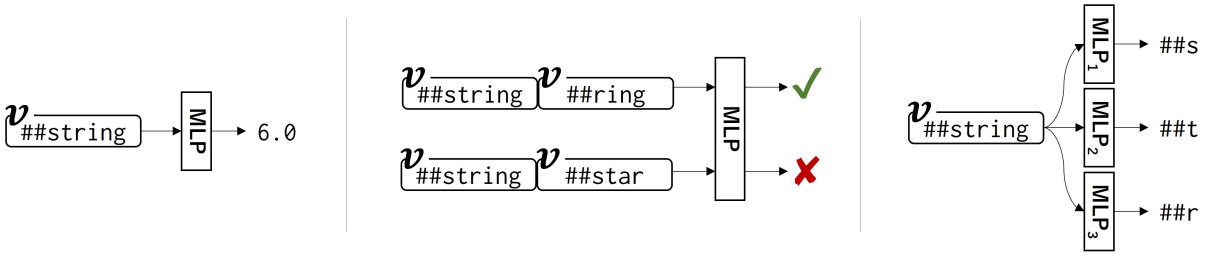
本研究では，以下の 3 つの観点から分散表現の表層的な知識について検証を行う．

(A)文字数の情報を獲得しているか (§3)
(B)部分文字列の情報を獲得しているか (§4)
(C)構成文字に関する情報を獲得しているか (§5)

各検証方法の概要をそれぞれ図 1 にまとめた．

各検証では，サブワード $s \in S$ に対応する分散表現 $\mathbf{v}_s$ や単語 $w \in W$ に対応する分散表現 $\mathbf{v}_w$ を入力として，検証に応じて異なる多層パーセプトロン (MLP) $f$ を学習する．学習した MLP による分散表現の変換の結果（$y = f(\mathbf{v}_s)$ あるいは $y = f(\mathbf{v}_w)$）を観察分析することで，検証を行う．

入力となるサブワード・単語のリストについて k-分割交差検証 ($k = 10$) の平均値を報告する．すなわち，サブワード単位では事前学習済み言語モデルの




(A) 文字数の予測　(B) 部分文字列の予測　(C) 構成文字の予測

**図 1**　サブワード・単語分散表現が持つ表層に関する知識の調査に用いる手法の概要.

語彙 $S$，単語単位では別途作成する単語リスト $W$ について，9 割で MLP を学習し，1 割で評価を行う工程を 10 回繰り返した結果の平均を報告する.

## 2.2　検証対象の事前学習済みモデル

本研究では，英語を主な学習データとする 9 つのモデルの表層的な知識を検証する．各モデルの事前学習には異なるコーパスが用いられており，語彙の規模や内容も異なるため，モデル間での数値の比較はできないことに留意されたい.

従来の分散表現として **Word2Vec** [6]，**FastText** [7]，**GloVe** [8] を用いた [1]．事前学習済みのマスク言語モデル [9] として **BERT-base-cased/uncased**，単方向の言語モデルとして **T5-base** [10] と **LLaMA2-7B**，**LLaMA2-7B-chat** [11]，文字単位の情報を入力とする言語モデルとして **CANINE** [12] を用いた.

## 2.3　入力の単位

### 2.3.1　サブワード単位

サブワード単位での検証では，単純にモデルが持つ語彙 $S$ に含まれる各サブワード $s$ に対応する分散表現 $\mathbf{v}_s$ を入力として用いる.

### 2.3.2　単語単位

単語単位での検証では，単語のリスト $W$ に含まれる各単語 $w$ に対応する分散表現 $\mathbf{v}_w$ を入力として用いる．単語のリスト $W$ には Wiktionary が提供する，Project Gutenberg での頻出 100,000 単語[2] を用いた.

単語に対応する分散表現 $\mathbf{v}_w$ は，学習済みの言語モデルでエンコードすることで計算する．具体的には，各学習済みモデルに同梱されているトークナイザーを用いて，単語を $M$ 個のサブワードの系列 $w = s_1, ..., s_M$ へと変換し，[CLS] や </s> などの特殊トークンとあわせて学習済みの言語モデルへと入力する．言語モデルの出力のうち，特定のトークンに対応する出力を単語の分散表現 $\mathbf{v}_w$ として利用する[3]．出力を使用する入力トークンはモデルによって異なるため，付録 B に詳しくまとめた.

# 3　文字数に関する知識

## 3.1　検証方法

学習済みの分散表現が表層的な知識を獲得しているのであれば，分散表現から文字数を復元できるはずである．例えば “word” という単語に対応する分散表現から，文字数が 4 であるという情報を抜き出せるはずである．そこで，単語の分散表現から文字数を予測する回帰モデルを新たに学習する.

$$l_w = f_{\text{length}}(\mathbf{v}_w). \tag{1}$$

ここで $f_{\text{length}}(\cdot)$ は多層パーセプトロン (MLP)[4] で，平均二乗誤差（MSE）の最小化による学習を行った.

サブワードの分散表現について検証する場合は，$\mathbf{v}_w$ の代わりに $\mathbf{v}_s$ を入力とする．サブワードには，単語の先頭や途中を示す特殊な接頭辞が付与されてる場合がある [5]．本研究では表層的な単語の長さを対象としているため，これらの接頭辞は文字数には含まない．すなわち，“##string” というサブワードの文字数は 6 として取り扱う.

## 3.2　結果

表 2 の「文字数」の列に記載した実験結果では，回帰モデルの予測数値と正解との差を MSE で比較した評価（回帰）と，予測数値を四捨五入した値を

1)　モデルの詳細を付録 A にまとめた.
2)　2005 年 8 月 15 日版 [13] を使用．一部に単語の重複があるため，実際の単語リストの規模は 83,404 単語である.
3)　Transformer[14] で文字から単語の表現を計算する場合に CLS トークンを用いる既存研究 [15] に倣った.
4)　本研究における全ての MLP は 3 層であり，活性化関数には ReLU 関数を用いた．また，学習時には入力となるサブワード・単語の分散表現のパラメータを固定した.
5)　SentencePiece [16] の U+2581，BERTTokenizer の “##” など.

**表 2**　実験の結果．平均二乗誤差（MSE），重み付き F1 値（F1%），正解率（ACC）を評価指標として用いた．

| | サブワード単位 | | | | | 単語単位 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 文字数 | | 部分 | 構成文字 | | 文字数 | | 部分 | 構成文字 | |
| | 回帰 | 分類 | 文字列 | 前向き | 後向き | 回帰 | 分類 | 文字列 | 前向き | 後向き |
| | (MSE) | (F1%) | (F1%) | (ACC) | (ACC) | (MSE) | (F1%) | (F1%) | (ACC) | (ACC) |
| Word2Vec | 9.48 | 26.85 | 99.68 | 21.48 | 18.93 | - | - | - | - | - |
| FastText | 2.18 | 30.23 | 99.62 | 37.67 | 25.91 | - | - | - | - | - |
| GloVe | 4.43 | 19.06 | 98.25 | 19.38 | 14.36 | - | - | - | - | - |
| BERT-base-uncased | 1.52 | 42.68 | 99.00 | 38.80 | 22.52 | 1.16 | 41.37 | 98.58 | 38.77 | 32.81 |
| BERT-base-cased | 1.48 | 41.69 | 99.63 | 39.54 | 26.35 | 3.79 | 39.14 | 98.59 | 37.57 | 34.30 |
| T5-base | 1.61 | 27.90 | 99.83 | 28.89 | 13.05 | 0.75 | 52.30 | 97.56 | 48.10 | 42.37 |
| LLaMA2-7B | 0.35 | 75.37 | 99.93 | 51.39 | 41.93 | 0.35 | 52.45 | 98.21 | 55.15 | 47.37 |
| LLaMA2-7B-chat | 0.54 | 60.99 | 99.93 | 51.68 | 40.60 | 1.00 | 46.27 | 98.13 | 54.85 | 47.87 |
| CANINE | - | - | - | - | - | 0.38 | 88.81 | 99.91 | 83.53 | 77.27 |

文字数の分類問題として重み付き F1 値で比較した評価（分類）を掲載した．分類として評価した場合の F1 値は全体的に低めだが，回帰として評価した場合の MSE は比較的小さく，ある程度の正確さで文字数を予測できているとみられる．特に言語モデルを用いて学習した分散表現においては，MSE が小さくなっている．付録の図 3,4 に掲載した詳細な予測の結果を見ても，10 文字程度までの入力であれば多くの場合において文字数を再現できている．単体のサブワードだけではなく，複数のサブワードから構成される単語単位での実験においても文字数を予測できており，言語モデルベースの手法は，モデル全体に文字数の知識がある程度の範囲において獲得されていると考えられる．

# 4　部分文字列に関する知識

## 4.1　検証方法

学習済みの分散表現が表層的な知識を獲得しているのであれば，単語分散表現から構成要素である部分文字列を復元できるはずである．例えば，“word” という単語に “ord” が含まれ，“old” は含まれないことを正しく識別できると期待される．そこで，単語 $w$ に対応する分散表現 $\mathbf{v}_w$ について，文字列 $s \in S$ が $w$ の部分文字列かを判別する分類器を作成する．分類器を用いて，文字列が単語の部分文字列である確率 $p(t|w)$ を次のように計算する．

$$p(s|w) = \sigma(f_{\text{substring}}(\mathbf{v}_w \oplus \mathbf{v}_s)). \tag{2}$$

ここで $\sigma$ はシグモイド関数，$f_{\text{substring}}(\cdot)$ は MLP である．また，$\oplus$ はベクトルの連結を表す．$s$ が $w$ の部分文字列となる場合を正例とし，同数の負例を語彙 $S$ からサンプリングすることで交差エントロピー誤差最小化による学習を行った．サブワード単位での実験では，$\mathbf{v}_w$ の代わりに $\mathbf{v}_s$ を用いた．評価時はテスト分割の入力に対して，語彙 $S$ に含まれるすべてのサブワードが部分文字列になるかを分類する．

## 4.2　結果

表 2 の「部分文字列」の列に記載した実験結果より，事前学習済みの分散表現からほぼ正確に部分文字列についての分類ができることが分かった．ここから，事前学習済みの言語モデルの分散表現は部分文字列に関する知識を持つと考えられる．この結果は，既存研究による報告 [5] とも合致する．

# 5　構成文字に関する知識

## 5.1　検証方法

学習済みの分散表現が表層的な知識を獲得しているのであれば，分散表現から構成文字を復元できるはずである．例えば “word” が，“w + ##o + ##r + ##d” から構成されると識別できるはずである．そこで，分散表現 $\mathrm{v}_w$ から単語 $w$ の $N$ 番目の文字を予測する分類器を作成する．単語の $N$ 番目の文字が $c$ である確率 $p(c|w, N)$ は次のように計算する．

$$\mathbf{h}_w^{\text{char}} = f_N(\mathbf{v}_w), \tag{3}$$

$$p(c|w, N) = \frac{\exp(\mathbf{h}_w^{\text{char}\top}\mathbf{v}_c)}{\sum_{t \in S_c} \exp(\mathbf{h}_w^{\text{char}\top}\mathbf{v}_t)}. \tag{4}$$

ここで $S_c$ は，事前学習済みのモデルの語彙のうち文字数が 1 のトークンの集合である．$f_N(\cdot)$ は

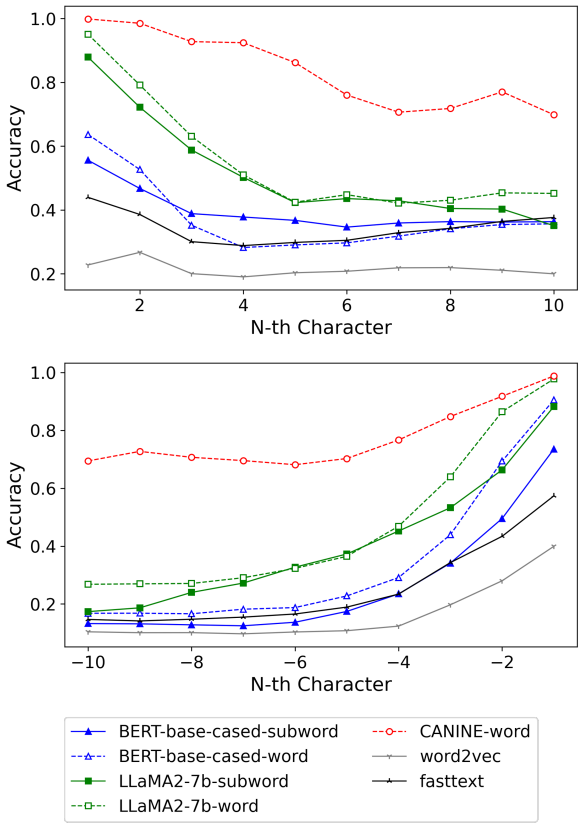



**図 2**　前向き方向（上）と後ろ向き方向（下）それぞれの実験設定における $N$ 文字目の予測精度の平均．$N = 2$ は末尾から後ろ向きに数えて 2 文字目の予測精度を示す．

MLP であり，予測する文字の表層上の位置（$N$ 文字目）ごとに用意する．例えば $f_{N=1}(\cdot)$ は，入力の分散表現に対応する表層のうち，1 文字目に出現する文字を予測するための MLP である．本研究では $N = \{1, ..., 10\}$，すなわち 1 文字目から 10 文字目までの範囲で検証を行った．また，単語の先頭と末尾からそれぞれ前向き・後ろ向きに数えて $N$ 文字目を予測する 2 種類の実験を行った．分類器は交差エントロピー誤差の最小化を目的関数として学習を行った．サブワード単位での実験では，$\mathbf{v}_w$ の代わりに $\mathbf{v}_s$ を入力として用いた．

### 5.2　結果

表 2 の「構成文字」の列に示した実験結果より，事前学習済みの分散表現からは構成文字を半分程度しか予測できないことが分かった．前向きからの予測精度の方が後ろ向きの予測精度に比べて高いことから，サブワードや単語の先頭に近い文字ほど予測しやすいこと分かる．いくつかのモデルについて $N$ 文字目ごとの予測精度を並べてみると（図 2），後ろ向きに予測する場合の方が $N$ が増えたときの性能の落ち込みが大きい．この結果から事前学習済みの分散表現は，表層を構成する文字がどの位置にあるかについての情報を獲得していないことが示唆される．特に表層の中ほどを構成する文字については，ほとんど知識を獲得できていないと考えられる．

**表 3**　生成による文字数予測の結果．

| モデル | 予測性能 (F1%) |
|---|---|
| BERT-base-uncased | 9.97 |
| BERT-base-cased | 2.96 |
| LLaMA2-7B | 3.54 |
| LLaMA2-7B-chat | 21.08 |

## 6　生成による知識の検証

§3,4,5 でのモデルの内部的な知識の検証に加えて，事前学習済みの言語モデルが単語の文字数を適切に生成できるかを評価する．“The number of characters in {WORD} is” に続く 1 から 20 の数字のうち，最も生成確率が高いものをモデルの予測として扱う．Wiktionary の頻出上位 1,000 単語を対象とし，BERT と LLaMA2 の出力について評価を行った．

表 3 の結果より，今回取り上げた事前学習済みの言語モデルでは単語の文字数を適切に生成できないことが分かった．LLaMA2-7B-chat の出力結果が最も性能が高いが，§3 での内部的な知識の検証結果には大きく劣る．この結果は，モデルが内部にもつ分散表現には単語の文字数に関する知識がある程度獲得されているにもかかわらず，出力時に内部の知識を適切に取り扱えないことを示唆する．すなわち，LLM が表層的な情報を適切に扱えない問題は，出力側の機構の計算能力 [17] などの情報処理能力の低さにも原因があると考えられる．

## 7　おわりに

事前学習済みの言語モデルの分散表現には，表層的な長さや部分文字列に関する知識が部分的に獲得されているようである．一方で，どの文字がどの位置に出現するかといった順序の情報は十分に獲得されておらず，表層的な情報の取り扱いの不得手に繋がっていると考えられる．また，獲得した知識を取り出して使用する出力側の情報処理能力にも課題が残されていることが示唆された．引き続き多言語での表層知識に関する検証を重ね，LLM の基礎能力を評価するベンチマークなどにも展開していきたい．

## 謝辞

本研究成果は，国立研究開発法人情報通信研究機構（NICT）の委託研究（22501）により得られたものです．また本研究の一部は，JST，ACT-X，JPMJAX21AM の支援を受けて実施しました．

## 参考文献

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. **Advances in neural information processing systems**, Vol. 33, pp. 1877–1901, 2020.

[2] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. **arXiv preprint arXiv:2107.03374**, 2021.

[3] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. **arXiv preprint arXiv:2307.03109**, 2023.

[4] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. **ACM Computing Surveys**, Vol. 56, No. 2, pp. 1–40, 2023.

[5] Ayush Kaushal and Kyle Mahowald. What do tokens know about their characters and how do they know it? In **Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 2487–2507, 2022.

[6] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

[7] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. **arXiv preprint arXiv:1607.04606**, 2016.

[8] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In **Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)**, pp. 1532–1543, 2014.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, **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, June 2019. Association for Computational Linguistics.

[10] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. **Journal of Machine Learning Research**, Vol. 21, No. 140, pp. 1–67, 2020.

[11] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. **arXiv preprint arXiv:2307.09288**, 2023.

[12] Jonathan H Clark, Dan Garrette, Iulia Turc, and John Wieting. Canine: Pre-training an efficient tokenization-free encoder for language representation. **Transactions of the Association for Computational Linguistics**, Vol. 10, pp. 73–91, 2022.

[13] Wiktionary:frequency lists/english/project gutenberg. https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists/English/Project_Gutenberg.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. **Advances in neural information processing systems**, Vol. 30, pp. 5998–6008, 2017.

[15] Li Sun, Florian Luisier, Kayhan Batmanghelich, Dinei Florencio, and Cha Zhang. From characters to words: Hierarchical pre-trained language model for open-vocabulary language understanding. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, **Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 3605–3620, Toronto, Canada, July 2023. Association for Computational Linguistics.

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

[17] GP Spithourakis and S Riedel. Numeracy for language models: Evaluating and improving their ability to predict numbers. In **ACL 2018-56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)**, Vol. 56, pp. 2104–2115. Association for Computational Linguistics, 2018.

[18] Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. **NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic**, Vol. 3, No. 2, 2011.

[19] Steven Bird. Nltk: the natural language toolkit. In **Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions**, pp. 69–72, 2006.

**表 4**　各事前学習済み言語モデルに同梱されたトークナイザーで単語 “Wugs” を分割した時のサブワード列と，出力を単語分散表現 $\mathbf{v}_w$ として使用する入力（下線）．

| モデル | サブワード列 |
|---|---|
| BERT-base-uncased | [CLS] wu ##gs [SEP] |
| BERT-base-cased | [CLS] Wu ##gs [SEP] |
| T5-base | ▁Wu g s </s> |
| LLaMA2-7B | <s> ▁W ugs |
| LLaMA2-7B-chat | <s> ▁W ugs |
| CANINE | [BOS] W u g s [EOS] |

## A　Word2Vec, FastText, GloVe の詳細

分析で用いる Word2Vec と FastText のモデルは，Gensim [18] が提供するパッケージを用いて独自に学習を行った．どちらも，英語の Wikipedia から作成した 20GB のテキストを学習データとして用い，Skip Gram による学習を行った．学習データは，NLTK [19] の英語用トークナイザーによって句読点等の分割を行ってから利用した．分散表現のパラメータ数は 768，学習エポック数は 10，学習対象にする単語の最小出現回数は 10 とした．また，学習に用いる窓幅は 5 とした．GloVe のモデルには，glove-wiki-gigaword-300 を使用した．

## B　モデルごとの入力例

単語単位の検証においては，学習済みの言語モデルで単語を構成するサブワード列をエンコードした結果を分散表現として用いる．この時，特定の入力トークンに対応する出力を，単語全体の分散表現として利用する．事前学習済みの言語モデルの構造や，対応している特殊トークンに応じて，どの入力トークンに対応する出力を用いるかは異なる．表 4 に，モデルごとに対応する入力トークンをまとめた．表において，下線で示したトークンに対応する出力を，単語の分散表現として各検証で学習する MLP に入力する．BERT と CANINE の各モデルについては，先頭に挿入される特殊トークンに対応する出力を，単語の分散表現として用いた．T5 では，文の末尾に追加される特殊トークンに対応する出力を単語の分散表現として用いた．LLaMA2 の各モデルでは，入力の末尾のトークンに対応する出力を単語の分散表現として用いた[6]．

6）　LLaMA2 は単方向の言語モデルであり，入力内容によらず 1 トークン目に対応する出力結果は一定であるため，先頭の特殊トークンを用いることはできない．

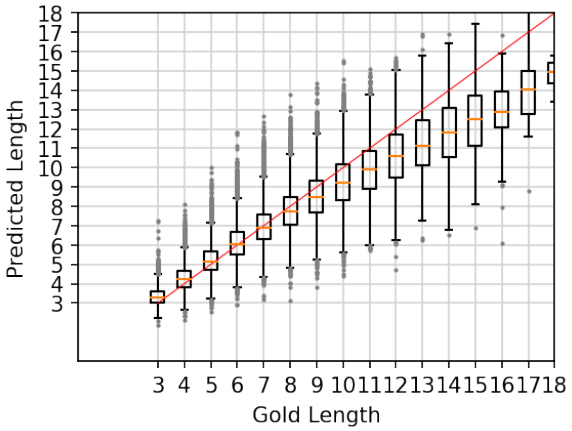



**図 3**　BERT-base-cased を用いた単語単位の文字数における，回帰モデルの予測結果と正解との比較．

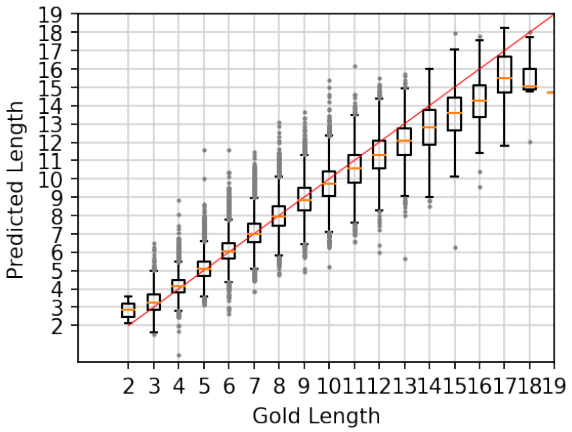



**図 4**　LLaMA2-7B-chat を用いた単語単位の文字数における，回帰モデルの予測結果と正解との比較．

## C　文字数の詳細な予測結果

図 3 と図 4 に，それぞれ BERT-base-cased と LLaMA2-7B-chat を用いた単語単位での文字数予測（§3，表 2）と正解の文字数との比較結果を示した．横軸は正解の文字数，縦軸は回帰モデルが予測した文字数であり，赤い直線は正解を示す．結果より，どちらのモデルを用いた場合であっても，3 から 10 文字程度の長さの単語であれば，それなりに正解に近い文字数を予測できていると言える．特に LLaMA2 のモデルは，比較的長い単語であっても予測が大きく外れることが無く，表層的な長さの知識をある程度の範囲で獲得できていると言える．