日本語投機的デコーディングの検討

林崎由　能勢隆　伊藤彰則
東北大学
hayashizaki.yu.t5@dc.tohoku.ac.jp
{takashi.nose.b7,akinori.ito.a2}@tohoku.ac.jp

概要

投機的デコーディングは，言語モデルによるテキスト生成において，出力の精度を低下させることなく推論を高速化する手法として注目を集めている．この手法ではドラフトモデルと呼ばれるより小規模な言語モデルを利用することで生成の一部並列化による推論速度の向上を実現している．一方，これまでの検討は英語が中心であり，日本語などの言語固有の性能やドメイン依存のメカニズムについては明らかになっていない．本研究では，日本語を対象として，モデルサイズの異なる複数の事前学習済み言語モデルに基づきドラフトモデルを構築し，日本語要約タスクにおける投機的デコーディングの効果検証およびトークン単位での詳細な分析を行う．

1　はじめに

大規模言語モデル (LLM) は，多種多様なタスクを高精度で解くことができる汎用性から，自然言語処理の根幹を担う技術として注目を集めている [1, 2]．その一方，これらのモデルは一般的にパラメータ数が非常に多く，1 回の推論につき 1 つしかトークンを生成できないため，テキスト生成速度が遅いことが問題となっている．言語モデルによるテキスト生成を高速化する手法も多く提案されているが [3, 4]，いずれも出力精度の低下が避けられない．この性質により，マルチモーダル対話 [5, 6] や自動運転 [7] などのリアルタイム性と出力品質が同時に要求される用途での応用が妨げられている．

出力の精度を低下させることなくテキスト生成を高速化する手法として，投機的デコーディング (Speculative decoding)[8, 9] が注目されている．この手法では，推論を高速化したい対象の言語モデル (ターゲットモデル) に対し，ドラフトモデルと呼ばれるより小規模な言語モデルにより高速に複数の候補トークンを逐次生成した後，ターゲットモデルに

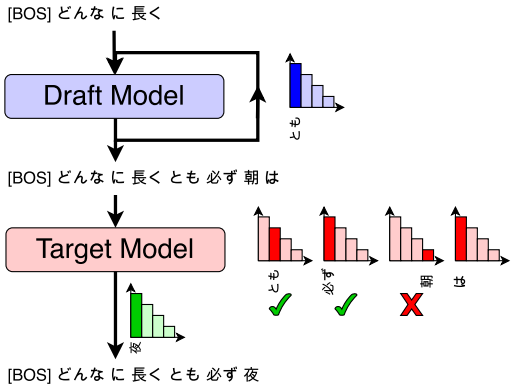

図 1: 投機的サンプリングの概要図

より並列に出力を検証する投機的サンプリングと呼ばれる処理を繰り返す (図 1)．候補トークンが受理された場合，ターゲットモデルの 1 回の推論につき，同時に複数のトークンを生成することができるため，高速化を実現することができる．ドラフトモデルを追加で運用する必要があるが，この手法で生成されたトークンの分布は単一のターゲットモデルで生成した場合と一致することが理論的に保証されている [8, 9] ため，有望な手法であると言える．

一方，投機的デコーディングにより得られる効果はドメインに依存することが知られている．先行研究では，英語での要約 [8, 9]，英独翻訳 [8]，コード生成 [9] でテキスト生成速度が変化することを示しているが，ドメイン依存の詳細なメカニズムについては明らかになっていない．また，言語固有の特徴 [10, 11] が影響を与える可能性があるにも関わらず，英語以外での検証も行われていない．

そこで本研究では，日本語の事前学習済み言語モデルを構築し，要約生成タスク XLSum[12] により投機的デコーディングの効果検証を行う．また，ターゲットモデルにより検証されるトークンに着目し，そのメカニズムを分析する．なお，実験で使用したコードとモデルは https://github.com/u-hyszk/japanese-speculative-decoding で公開している．

2 投機的デコーディング

投機的デコーディング [8, 9] は，言語モデルでのテキスト生成において出力の精度を低下させることなく推論を高速化する手法である．この手法では，推論の対象となるターゲットモデルに対し，より小規模なドラフトモデルを用いて予め γ 個の候補トークンを逐次生成した後，ターゲットモデルにより並列に出力を検証する投機的サンプリング (アルゴリズム 1) による部分的な並列推論を繰り返し行うことでトークンを生成する[1)]．各投機的サンプリングにおいて，ターゲットモデルでの推論は 1 回しか行われないが[2)]，アルゴリズム 1 の 10 行目においてトークンが受理された回数に応じて最大で $\gamma+1$ 個のトークンを生成できるため，高速化が期待される．また，この処理で生成されたトークンの分布は単一のターゲットモデルで生成した場合と一致することが理論的に保証されている [8, 9].

投機的デコーディングでは，(1) ドラフトモデルの推論速度が速く，(2) ドラフトモデルとターゲットモデルの出力分布が近いほど，高速化が期待される．(2) について，より具体的には，トークンが受理される確率であるトークン受理率 β と確率分布 $p(y)$，$q(y)$ 間の全変動距離 $D_{\mathrm{TVD}}(p||q)$ との間に次式の関係があることが示されている [8, 13].

$$\beta = 1 - D_{\mathrm{TVD}}(p||q) \tag{1}$$

ただし，全変動距離 $D_{\mathrm{TVD}}(p||q)$ は以下のように定義される．

$$D_{\mathrm{TVD}}(p||q) = \sum_{y} \left| \frac{p(y) - q(y)}{2} \right| \tag{2}$$

このトークン受理率 β は，一般的には平均トークン受理率 α(検証された総トークン数に占める受理されたトークンの割合) で報告され，ドメインに依存して変化することが知られている．例えば，Chen ら [9] は，コード生成タスク Human Eval[14] では α が高く，英語要約タスク XSum[15] では α が低くなる傾向にあることを報告している．

3 日本語言語モデルでの検討

本研究では，モデルサイズが異なる複数の日本語事前学習済み言語モデルを構築し，要約生成タスク XLSum[12] により，投機的デコーディングの効果検証を行う．本節では，日本語での検証を行うためのドラフトモデルの事前学習 (3.1 項)，および追加学習 (ファインチューニング)(3.2 項) について述べる．

Algorithm 1 投機的サンプリング

Input: 入力トークン列 $\mathbf{X} = \{x_1 ... x_n\}$, ドラフトモデル M_p, ターゲットモデル M_q, 候補トークン数 γ
Output: 生成トークン列 $\mathbf{Y} = \{y_1 ... y_{\gamma+1}\}$

1: **for** $i = 1 : \gamma$ **do**
2: $\quad p_i(y) \leftarrow M_p(\mathbf{X}, \hat{y}_{<i})$
3: $\quad \hat{y}_i \sim p_i(y)$
4: **end for**
5: $\{q_1(y), ..., q_\gamma(y), q_{\gamma+1}(y)\}$
6: $\leftarrow \{M_q(X), ..., M_q(X, \hat{y}_{<\gamma}), M_q(X, \hat{y}_{<\gamma+1})\}$
7: **for** $i = 1 : \gamma$ **do**
8: $\quad r \sim U[0, 1]$
9: $\quad$ **if** $r < \min(1, \frac{q_i(y)}{p_i(y)})$ **then**
10: $\quad\quad y_i \leftarrow \hat{y}_i$
11: $\quad$ **else**
12: $\quad\quad y_i \sim \frac{\max(0, q_i(y) - p_i(y))}{\sum_y \max(0, q_i(y) - p_i(y))}$
13: $\quad\quad$ **break**
14: $\quad$ **end if**
15: **end for**
16: **if** すべてのトークンが受理される **then**
17: $\quad y_{\gamma+1} \sim q_{\gamma+1}$
18: **end if**
19: **return** $\mathbf{Y}$

3.1 日本語ドラフトモデルの事前学習

本研究では，日本語での投機的デコーディングの効果検証を行うために，モデルサイズが異なる複数の日本語言語モデルを事前学習から構築し，ドラフトモデルとして使用することとした．事前学習から行った理由としては以下の 3 つが挙げられる:

1. 投機的デコーディングでは，ターゲットモデルとドラフトモデルの出力分布同士の演算を行うため，両方のモデルで共通の語彙を使用する必要がある．
2. 公開されている日本語の事前学習済み言語モデルのうち，上記の条件 1 を満たしているものは少ない．条件を満たすものについても，異なるモデルサイズが少ないため，分析できる範囲が限られる[3)4)]．
3. 投機的サンプリングにおける受理率の向上と出力の正確な分析のために，学習に用いるデータセットが同じであることが望ましい．

1) $p(y)$ および $q(y)$ はそれぞれドラフトモデル M_p，ターゲットモデル M_q の出力分布である．また，$y \sim p(y)$ は確率分布 $p(y)$ からトークン y がサンプリングされたことを表す．U は一様分布である．
2) アルゴリズム 1 の 5~6 行目で 1 回のみ並列で推論を行う．
3) https://rinna.co.jp/news/2023/05/20220531.html
4) https://www.cyberagent.co.jp/news/detail/id=28817

構築した事前学習モデルのモデルサイズとアーキテクチャは付録 A.1 の通りである．重要な点として，全てのモデルで OpenCALM 1b[5] と同じトークナイザを用いることで，共通の語彙を使用できるようにした．また，学習データセットも全てのモデルで共通であり，日本語 Wikipedia[5] および日本語 CC-100[16] から 1 億 240 万文を取得して使用した．詳細については付録 A.1 に示す．

3.2 日本語評価タスクでの追加学習

本研究では，日本語における投機的デコーディングの評価に多言語要約データセットの XLSum[12] を採用し，前項の事前学習済みモデルに対して XLSum を用いた追加学習を行った．追加学習を行った理由としては，モデルの出力をタスクの内容と一致させることで公平な評価を行うためである[6]．XLSum はニュースメディア BBC の記事とその要約から構成されたデータセットである．このうち日本語の学習用データ 7110 文により追加学習を行った (付録 A.2)．テストデータ 889 文を使用して，自動評価指標 Rouge[17] により評価した結果を付録 A.2 に示す．また，OpenCALM medium/small[5] を同様に追加学習した結果も併記する．表 3 の結果から，前項で作成した事前学習済み言語モデルは，先行研究のモデルと同程度の要約性能を有していると考えられる．

4 実験

本節では，3.2 節で作成した要約モデルを使用して，日本語における投機的デコーディングの効果を検証する．まず，実験条件について述べ (4.1 項)，生成高速化実験の結果を示し (4.2 項)，最後に受理されたトークンについて分析を行う (4.3 項)．

4.1 実験条件

使用データ　3.2 節で述べた XLSum のテストデータ 889 文のタイトルと本文を入力として使用した．

使用モデル　3.2 節で作成した要約モデルを使用した．ターゲットモデルには 409M パラメータのモデルを使用し，ドラフトモデルにはそれ以外のモデルを使用した．なお，すべての実験は RTX 3090 1 枚を搭載した単一の計算機により行った．

表 1: 日本語 XLSum による投機的デコーディングの結果．ただし，ターゲットモデルのパラメータ数は 409M である．

T	# of params	γ (best)	α	c	Speed up
0.0	6M	3	0.405	0.026	x1.33
	13M	7	0.498	0.028	x1.45
	29M	7	0.636	0.062	x1.58
	47M	1	0.617	0.082	**x2.02**
	72M	1	0.625	0.119	x1.93
	115M	1	0.667	0.198	x1.21
	165M	1	0.668	0.322	x1.16
	247M	1	0.677	0.745	x0.90
1.0	6M	7	0.412	0.071	x1.18
	13M	5	0.446	0.051	x1.34
	29M	1	0.515	0.099	x1.26
	47M	1	0.549	0.109	**x1.55**
	72M	1	0.580	0.186	x1.40
	115M	1	0.608	0.305	x1.13
	165M	1	0.607	0.433	x1.10
	247M	1	0.650	0.699	x0.92

評価指標　先行研究 [8, 9] と同様に，次式で定義されるスピードアップを評価指標とした．なお，t_{target} はターゲットモデルの 1 トークンあたりの生成時間，t_{spec} は投機的デコーディングによる 1 トークンあたりの生成時間である．

$$Speedup = \frac{t_{\text{target}}}{t_{\text{spec}}} \tag{3}$$

また，ドラフトモデルとターゲットモデルの出力の近さを表す平均トークン受理率 α と，ドラフトモデルの推論速度を確認する指標として，ターゲットモデルに対するドラフトモデルの 1 トークンあたりの生成時間の比 c も同様に報告する．なお，全ての評価指標において 3 回の測定の平均値を報告する．

ハイパーパラメータ　出力の多様性を制御する Temperature(T) は $\{0.0, 1.0\}$ で評価した．候補トークン数 γ は $\{1, 3, 5, 7\}$ の 4 種類で評価した．

4.2 日本語要約タスクでの生成高速化実験

実験の結果を表 1 に示す．なお，表 1 では最もスピードアップが高かった場合の候補トークン数 γ の結果のみを表記し，候補トークン数を変えた場合の結果は付録 A.3 に示す．Temperature が 0.0，1.0 の両方の場合において，ドラフトモデルが 47M の場合に最もスピードアップが大きくなり，それぞれ x2.02, x1.55 のスピードアップが得られた．英語要約タスクにおいて 2 倍から 3 倍程度のスピードアップが得

5) https://dumps.wikimedia.org/jawiki/20231201/

6) 例えば，タスクの内容に反して，同じトークンを繰り返し生成し続けるターゲットモデルを評価に使用した場合，ドラフトモデルが容易に予想しやすくなるため，不当に高速になる問題が生じることを確認している．

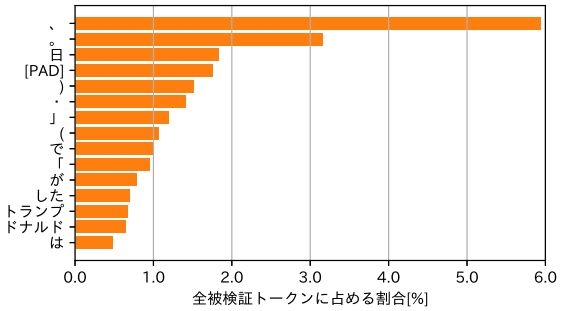

図 2: 受理された回数が多いトークンの上位 15 個

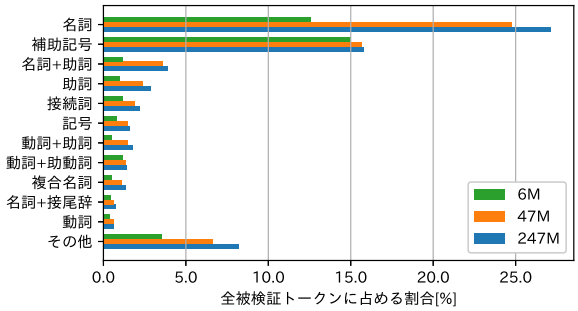

図 3: 受理されたトークンの品詞分類

られた先行研究 [8, 9] と比較した場合，若干速度向上の幅は小さいが，これについては言語やモデルサイズの違いが一因であると考えられる．

一方，ドラフトモデルのモデルサイズを変更して α や c が変化した場合，スピードアップが著しく低下する傾向も見られた．この傾向は先行研究 [8, 9] と同様であるものの，モデルサイズを細かく調整した本実験の結果から，得られるスピードアップはモデルサイズに対して鋭敏である可能性が示唆された．すなわち，投機的デコーディングによる効果を最大限得るためには，ターゲットモデルに合わせてドラフトモデルのサイズを慎重に選定することが重要であると考えられる．

4.3 受理トークンの傾向分析

本項では，投機的デコーディングにより受理されたトークンの傾向を分析する．まず，表 1 において，最もスピードアップが大きくなった，ドラフトモデルのパラメータ数が 47M，Temperature $T = 0.0$，候補トークン数 γ=1 の場合について，受理されたトークンの上位 15 個を図 2 に示す[7]．この結果から，受理されたトークンの上位には，句読点や括弧などの補助記号，および助詞が多く含まれていることが分かる．また，XLSum 特有のニュースに関連する名詞も上位に含まれている．

次に，受理されたトークンについて，MeCab[8] による形態素解析を行った結果を図 3 に示す[9]．モデルサイズによる比較のため，パラメータ数 6M の場合と 247M の場合の結果も併記する．この結果から，出力全体に対しても受理されるトークンの大部分は名詞や補助記号，および助詞が占めていることが分かる．また，モデルサイズが大きくなるほど，名詞のトークン受理率が高くなる傾向が見られる．

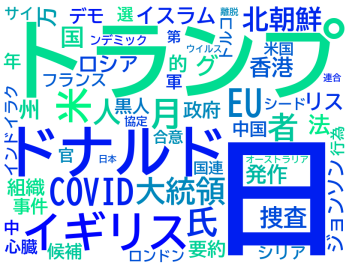

図 4: 受理された回数が多い「名詞」トークンのワードクラウド

最後に，「名詞」に属するトークンのうち，受理された回数が多いものをワードクラウドにより可視化した結果を図 4 に示す．この結果から，ニュース特有の「トランプ」「COVID」などのトークンが上位を占めていることが分かる．

以上の結果をまとめると，(1) ターゲットモデルにより受理されるトークンは名詞や補助記号，および助詞が大部分を占めており，(2) このうち名詞にはドメイン特有のトークンが多く含まれていた．特に (2) については，投機的デコーディングのドメイン依存のメカニズムとして，名詞トークンの受理率が大きく関連している可能性が示唆された．

5 まとめ

本研究では，日本語の事前学習済み言語モデルを構築し，要約生成タスク XLSum により投機的デコーディングによる推論速度向上の効果検証を行った．分析の結果，日本語でも適切なモデルサイズを選択することで投機的デコーディングが有効であることが確認された．また，ターゲットモデルにより受理されるトークンは名詞や句読点などの補助記号，および助詞が大部分を占め，特に名詞がドメイン依存を引き起こしている可能性が示唆された．

7) 被検証トークンとは，アルゴリズム 1 の 9 行目において検証されたトークンのことを指す．
8) https://taku910.github.io/mecab/
9) 複数の名詞から構成されるトークンの場合は「複合名詞」と表記した．また，複数の品詞から構成されるトークンは，「名詞+助詞」などと表記した．

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, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, **Advances in Neural Information Processing Systems**, Vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020.

[2] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. In **ArXiv**, 2023.

[3] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In **ArXiv**, 2015.

[4] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. **J. Mach. Learn. Res.**, Vol. 18, No. 1, p. 6869–6898, jan 2017.

[5] Takato Yamazaki, Tomoya Mizumoto, Katsumasa Yoshikawa, Masaya Ohagi, Toshiki Kawamoto, and Toshinori Sato. An open-domain avatar chatbot by exploiting a large language model. In Svetlana Stoyanchev, Shafiq Joty, David Schlangen, Ondrej Dusek, Casey Kennington, and Malihe Alikhani, editors, **Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue**, pp. 428–432, Prague, Czechia, September 2023. Association for Computational Linguistics.

[6] Ryota Yahagi, Yuya Chiba, Takashi Nose, and Akinori Ito. Multimodal dialogue response timing estimation using dialogue context encoder. In Svetlana Stoyanchev, Stefan Ultes, and Haizhou Li, editors, **Conversational AI for Natural Human-Centric Interaction**, pp. 133–141, Singapore, 2022. Springer Nature Singapore.

[7] Kotaro Tanahashi, Yuichi Inoue, Yu Yamaguchi, Hidetatsu Yaginuma, Daiki Shiotsuka, Hiroyuki Shimatani, Kohei Iwamasa, Yoshiaki Inoue, Takafumi Yamaguchi, Koki Igari, Tsukasa Horinouchi, Kento Tokuhiro, Yugo Tokuchi, and Shunsuke Aoki. Evaluation of large language models for decision making in autonomous driving. In **ArXiv**, 2023.

[8] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, **Proceedings of the 40th International Conference on Machine Learning**, Vol. 202 of **Proceedings of Machine Learning Research**, pp. 19274–19286. PMLR, 23–29 Jul 2023.

[9] Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. In **ArXiv**, 2023.

[10] 貴弘岩畑. 英語と日本語の構文選択における差異について. 人文研究 = Studies in humanities, No. 175, pp. 95–122, 2011.

[11] 光弘大村. 文化を映し出すことば：日英比較から文化を言語学する. 人文論集, Vol. 66, No. 2, pp. 81–95, 01 2016.

[12] Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, **Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021**, pp. 4693–4703, Online, August 2021. Association for Computational Linguistics.

[13] Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. Distillspec: Improving speculative decoding via knowledge distillation. In **ArXiv**, 2023.

[14] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. In **ArXiv**, 2021.

[15] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing**, pp. 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[16] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised Cross-lingual Representation Learning at Scale. In **ArXiv**, No. arXiv:1911.02116. arXiv, April 2020.

[17] Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In **Text Summarization Branches Out**, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, **Advances in Neural Information Processing Systems**, Vol. 30. Curran Associates, Inc., 2017.

[19] Alex Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Jason Phang, Shivanshu Purohit, Hailey Schoelkopf, Dashiell Stander, Tri Songz, Curt Tigges, Benjamin Thérien, Phil Wang, and Samuel Weinbach. GPT-NeoX: Large Scale Autoregressive Language Modeling in PyTorch, 9 2023. https://github.com/EleutherAI/gpt-neox.

[20] Kenta Shinzato. HojiChar: The text processing pipeline, September 2023. https://github.com/HojiChar/HojiChar.

A 付録

A.1 事前学習の詳細

事前学習では，表 2 の 9 種類の Transformer Decoder モデル [18] を構築した．事前学習ではライブラリ GPT-NeoX[19] を使用した．日本語 CC-100 データセットについては，学習データ量を絞るため，1 文あたり 30 文字以上の文のみを使用し，両方のデータセットで NFKC 正規化を行った．これらの一連の前処理には，ライブラリ hojichar[20] を使用した．テストデータ 50000 文により，perplexity を評価した結果を表 2 に示す．パラメータ数が増えるにつれて，perplexity が低下していることが確認できる．

表 2: モデルのパラメータ数とアーキテクチャ

# of params	Layers	Dim	Heads	Dev ppl.
6M	1	64	1	349.5
13M	1	128	1	253.1
29M	3	256	4	120.5
47M	4	384	6	95.2
72M	6	512	8	78.6
115M	10	640	10	65.5
165M	12	768	12	57.9
247M	16	896	16	52.5
409M	24	1024	16	45.9

A.2 追加学習モデルの要約性能

XLSum は日本語の学習用データ 7110 文により 10 エポックの追加学習を行い，損失が最も小さくなったモデルを選択して実験に使用した．テストデータ 889 文による評価結果では (表 3)，同パラメータ数・トークナイザの OpenCALM medium/small[5] と比較しても，同程度の要約性能となっていることが確認できる．

表 3: 日本語 XLSum による追加学習

model	# of params	Rouge-1	Rouge-2	Rouge-L
Ours	6M	8.88	2.49	7.76
	13M	13.90	3.82	11.80
	29M	25.46	9.25	20.94
	47M	32.37	12.59	26.00
	72M	34.96	14.18	27.81
	115M	37.06	14.33	29.33
	165M	38.70	16.46	30.77
	247M	39.31	16.63	30.83
	409M	40.36	17.06	31.53
Open-CALM[5]	165M	38.61	16.04	30.34
	409M	42.96	18.58	33.12

A.3 候補トークン数を変えた場合の結果

Temperature が 0.0，1.0 の両方の場合において，ドラフトモデルが 47M の場合に最もスピードアップが大きくなり，それぞれ x2.02，x1.55 のスピードアップが得られた．一方，ドラフトモデルのモデルサイズを変更して α や c が変化した場合，スピードアップが著しく低下する傾向も見られた．例えば，表 4 において，T=0.0，γ=1 のとき，パラメータ数 47M と 29M を比較すると，α と c がそれぞれ 0.027，0.020 しか変わらないにも関わらず，スピードアップは x2.02 から x1.24 に低下していることが分かる．また，実験設定によっては，スピードアップが等倍より低くなる場合もあった．

表 4: 日本語 XLSum による投機的デコーディングの結果．ただし，ターゲットモデルのパラメータ数は 409M である

		T=0.0		T=1.0	
# of params	γ	α	Speed up	α	Speed up
6	1	0.383	x1.33	0.368	x1.18
	3	0.405	x1.61	0.386	x1.25
	5	0.432	x1.30	0.401	x1.15
	7	0.457	x1.20	0.412	x1.26
13	1	0.427	x1.39	0.411	x1.29
	3	0.447	x1.26	0.430	x1.26
	5	0.473	x1.18	0.446	x1.34
	7	0.498	x1.45	0.460	x1.05
29	1	0.590	x1.24	0.515	x1.26
	3	0.600	x1.32	0.528	x1.25
	5	0.620	x1.45	0.542	x1.15
	7	0.636	x1.58	0.556	x1.09
47	1	0.617	**x2.02**	0.549	**x1.55**
	3	0.630	x1.57	0.557	x1.11
	5	0.647	x1.55	0.573	x1.10
	7	0.663	x1.58	0.585	x1.02
72	1	0.645	x1.93	0.576	x1.40
	3	0.653	x1.40	0.583	x1.09
	5	0.671	x1.33	0.596	x1.06
	7	0.685	x1.23	0.611	x0.90
115	1	0.667	x1.21	0.608	x1.13
	3	0.676	x0.90	0.614	x0.97
	5	0.693	x0.93	0.624	x0.85
	7	0.705	x0.82	0.635	x0.75
165	1	0.668	x1.16	0.607	x1.10
	3	0.678	x0.88	0.620	x0.85
	5	0.694	x0.79	0.630	x0.66
	7	0.707	x0.62	0.641	x0.59
247	1	0.677	x0.90	0.624	x0.92
	3	0.684	x0.66	0.632	x0.69
	5	0.701	x0.58	0.639	x0.53
	7	0.712	x0.49	0.650	x0.45