Instability in Downstream Task Performance During LLM Pretraining

Yuto Nishida[1,3] **Masaru Isonuma**[3,2] **Yusuke Oda**[1,3]
[1]Nara Institute of Science and Technology [2]Tohoku University
[3]Research and Development Center for Large Language Models,
National Institute of Informatics
{nishida.yuto.nu8, yusuke.oda}@naist.ac.jp
isonuma@nii.ac.jp

Abstract

When training large language models (LLMs), it is common practice to track downstream task performance throughout the training process and select the checkpoint with the highest validation score. However, downstream metrics often exhibit substantial fluctuations, making it difficult to identify the checkpoint that truly represents the best-performing model. In this study, we empirically analyze the stability of downstream task performance in an LLM trained on diverse web-scale corpora. We find that task scores frequently fluctuate throughout training, both at the aggregate and example levels. To address this instability, we investigate two post-hoc checkpoint integration methods: checkpoint averaging and ensemble, motivated by the hypothesis that aggregating neighboring checkpoints can reduce performance volatility. We demonstrate both empirically and theoretically that these methods improve downstream performance stability without requiring any changes to the training procedure.

1 Introduction

Large language models (LLMs) are typically developed through pretraining on large-scale corpora, during which the performance on downstream tasks evolves dynamically. Understanding these training dynamics is essential for diagnosing issues during model development and ensuring the reliability, reproducibility, and effectiveness of the resulting models on downstream tasks. In particular, the stability of evaluation metrics plays a key role in model development: when evaluation metrics fluctuate erratically, they undermine the reliability of the evaluation and hamper fair comparison across checkpoints, models, or experimental settings. Such instability may arise not only in evaluation signals based on next-token prediction, such as training loss, but also in downstream task metrics that are central to model evaluation.

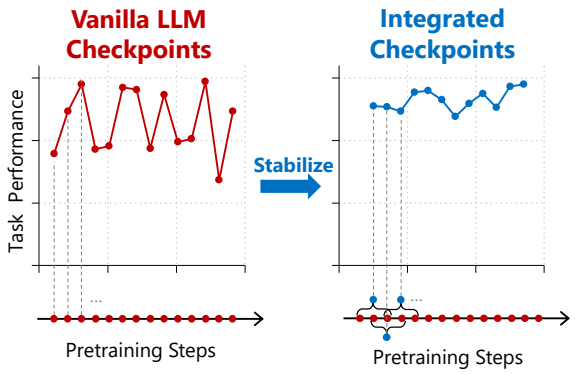


Figure 1: Overview of stabilizing downstream task performance during pretraining. We observe instability in task performance during LLM pretraining, and experimentally show that theoretically motivated checkpoint integration methods improve evaluation stability.

Previous research on training stability has largely focused on signals derived from the next-token prediction objective. Studies of loss dynamics have offered insights into how architectural design, optimization strategies, and initialization schemes can encourage smooth convergence (Chowdhery et al., 2023; Rybakov et al., 2024; Takase et al., 2024). More recently, Chang et al. (2024) have investigated token-level prediction probabilities and shown that their stability varies with token frequency and contextual diversity.

Practical evaluation of LLMs typically relies on downstream tasks such as generation, question answering, or classification, while prior studies shed light on dynamics related to the next-token prediction objective. For reliable evaluation, it is essential to understand how stable downstream performance is during pretraining and to explore ways to mitigate instability when it arises. However, little is known about how downstream task performance behaves throughout the pretraining process.

To this end, we empirically analyze the stability of LLMs of different sizes pretrained on a diverse

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 22883–22895
November 4-9, 2025

web-scale corpus. By tracking the performance of multiple downstream tasks across a series of training checkpoints, we observe that scores generally improve in the long run as training progresses, while in the short term they repeatedly rise and fall in an inconsistent manner, which we refer to as *fluctuations*. These fluctuations are observed not only in aggregate task-level scores but also at the level of individual examples, indicating that instability is inherent in downstream performance during LLM pretraining. We further examine the effect of model size on this instability and find that scaling up models does not necessarily mitigate fluctuations. This inherent instability, in turn, hinders robust and reliable evaluation.

To address this challenge, we explore methods for mitigating such fluctuations in downstream performance. Motivated by the hypothesis that aggregating multiple checkpoints can smooth out performance volatility during training, we investigate two checkpoint integration methods: checkpoint averaging and ensemble. These methods, while simple and theoretically motivated, can reduce short-term instability without altering the training process. Our experimental results show that these methods can mitigate performance instability and improve evaluation robustness during training.

2 Observation of Task Instability

2.1 Experimental Settings

We conduct our analysis using a family of Transformer-based language models trained on web-scale corpora containing a mixture of Japanese, English, and source code. The training data, totaling approximately 2.1 trillion tokens after weighted sampling, is drawn from the LLM-jp Corpus v3.[1] We analyze seven pre-trained models with varying the number of parameters: 150M, 440M, 980M, 1.8B, 3.7B, 7.2B, and 13B. During pretraining, checkpoints were regularly obtained at intervals ranging from several hundred to approximately a thousand steps for each model, according to predefined schedules detailed in Appendix A. In our analysis, we use all checkpoints except the initial one (step 0), where parameters are randomly initialized.[2] Our analysis focuses on the pretraining phase, and does not include instruction-tuned or otherwise adapted models.

For evaluation, we use the llm-jp-eval v1.4.1[3] to perform 4-shot inference on a suite of Japanese downstream tasks. We evaluate nine task categories: EL (entity linking), FA (fundamental analysis), HE (human examination), MC (multiple choice question answering), MR (mathematical reasoning), MT (machine translation), NLI (natural language inference), QA (question answering), and RC (reading comprehension). Each category typically includes multiple datasets; for example, the MC category includes three datasets. Details of the tasks, datasets, and evaluation metrics used in each category are provided in Appendix A.

2.2 Instability Across Task Categories

We begin by examining downstream performance across task categories over the course of pretraining. Figure 2 shows score trajectories for the 13B model across task categories. In all categories except for machine translation (MT; shown in Figure 2(f)), we observe persistent short-term score fluctuations throughout training.[4] Similar fluctuations are also observed in Figure 3, which presents the trajectory of average scores across all categories. These fluctuations are not limited to a particular phase of training, such as early instability or late saturation, but instead persist across many checkpoints. Similar trends are observed across models of other sizes, indicating that instability in task performance is a general phenomenon not limited to specific scales.

2.3 Example-Level Score Dynamics

To understand the source of task-level fluctuations, we investigate example-level prediction dynamics. Figure 4 shows score trajectories for ten examples drawn from JCommonsenseQA (Kurihara et al., 2022) in the MC category, Jamp (Sugimoto et al., 2023) in the NLI category, and JMMLU (Yin et al., 2024) in the HE category using the 13B model.[5] Since these tasks use exact match as the evaluation metric, the score for each example is either 0 (incorrect) or 1 (correct). Across many examples, we observe frequent alternations between correct and incorrect predictions, even over extended training periods. While some examples converge to stable predictions, many continue to exhibit prediction

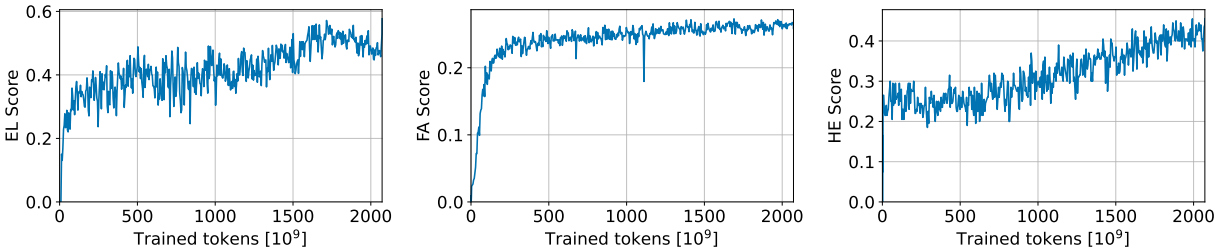
[1] https://gitlab.llm-jp.nii.ac.jp/datasets/llm-jp-corpus-v3

[2] At the first step, the learning rate is set to zero, so parameters at step 0 and step 1 are identical; the only difference is that the optimizer state is updated.

[3] https://github.com/llm-jp/llm-jp-eval

[4] In the following sections, we focus on the eight categories other than MT, where instability was qualitatively observed.

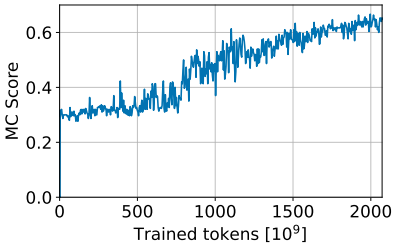
[5] We uniformly sampled indices, and then plotted the score trajectories of the examples corresponding to those indices.



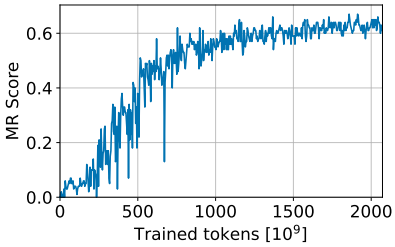
(a) EL (Entity Linking)

(b) FA (Fundamental Analysis)

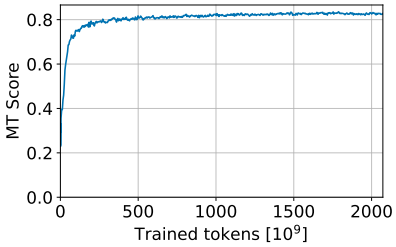
(c) HE (Human Examination)



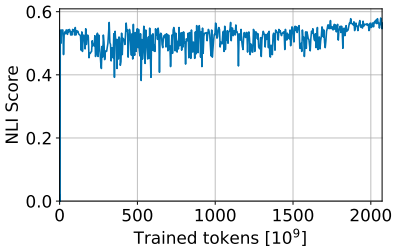
(d) MC (Multiple Choice QA)



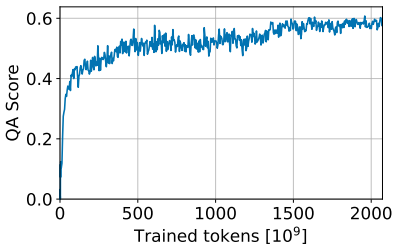
(e) MR (Mathematical Reasoning)



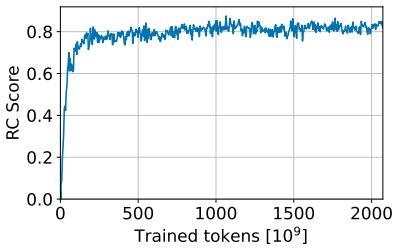
(f) MT (Machine Translation)



(g) NLI (Natural Language Inference)



(h) QA (Question Answering)



(i) RC (Reading Comprehension)

Figure 2: Score trajectories of the 13B model across downstream task categories. In most categories, we observe gradual long-term improvements over the course of training, accompanied by short-term score fluctuations that persist throughout.

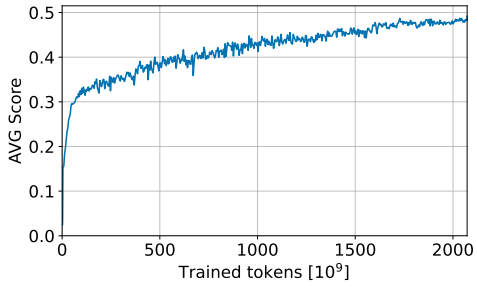


Figure 3: Average score trajectory of the 13B model across all downstream task categories.

instability well beyond early training.

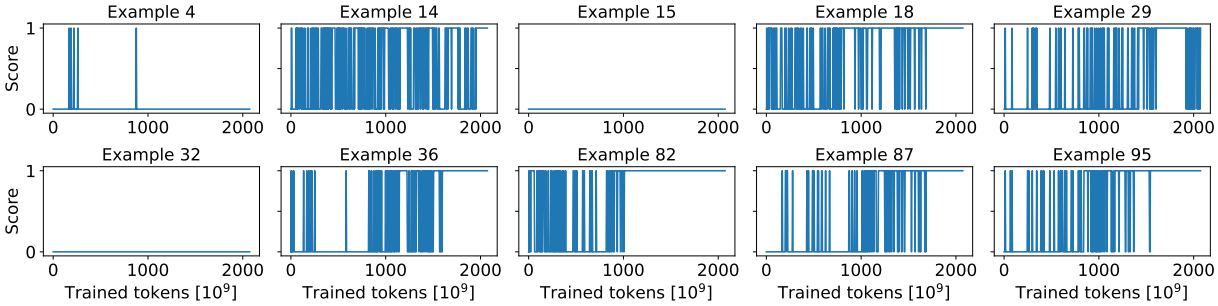
These example-level dynamics suggest that the task-level score fluctuations observed earlier may be attributed to the accumulation of such local prediction instabilities across many examples.

2.4 Scoring-Based Analysis of Pretraining Instability

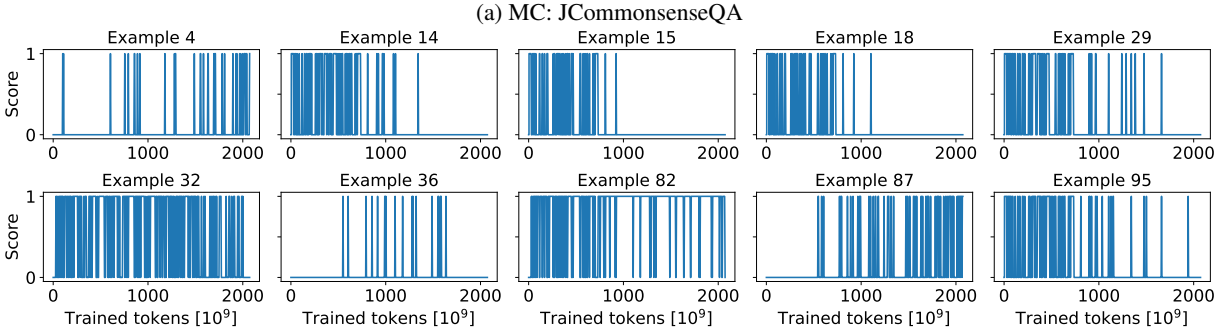
To quantify instability of downstream task performance across checkpoints, we begin by measuring fluctuations in reference-based evaluation scores. Let $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m\}$ denote the sequence of m checkpoints obtained during pretraining. Let $x_{\boldsymbol{\theta}_t}$ be the output generated using parameters $\boldsymbol{\theta}_t$ for a given input, and let $f(x_{\boldsymbol{\theta}_t})$ denote the reference-based evaluation score for that output (e.g., exact match or charF1). We denote $L(\boldsymbol{\theta}_t) = \mathbb{E}_{x_{\boldsymbol{\theta}_t} \sim p(x;\boldsymbol{\theta}_t)}[f(x_{\boldsymbol{\theta}_t})]$ as the expected value of the evaluation score given an output sampled from the model $\boldsymbol{\theta}_t$. We use total variation over these scores to measure how predictions change throughout training. Specifically, we compute the mean total variation (MTV) as follows:

$$\mathrm{MTV}(\boldsymbol{\Theta}) = \frac{1}{m-1} \sum_{t=1}^{m-1} \left| L(\boldsymbol{\theta}_{t+1}) - L(\boldsymbol{\theta}_t) \right|. \tag{1}$$

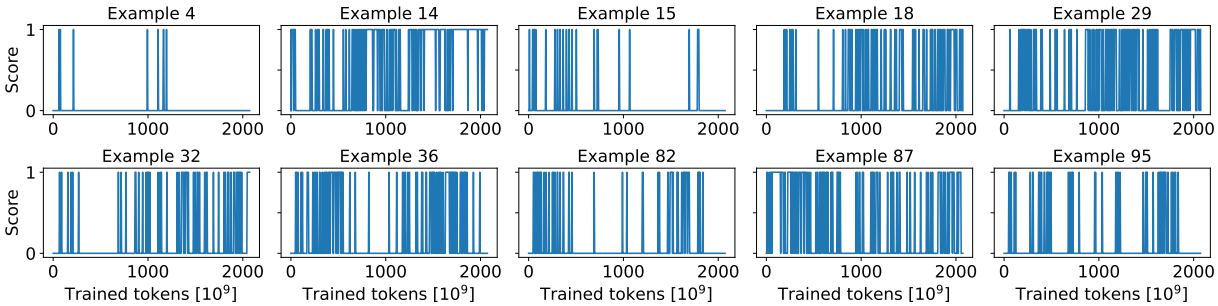
This value captures the average magnitude of score transitions and is normalized by the number of transitions. This normalization ensures fair comparison across models with different numbers of saved checkpoints. In our experiments, we com-



(a) MC: JCommonsenseQA



(b) NLI: Jamp



(c) HE: JMMLU

Figure 4: Example-level score trajectories of the 13B model during training. Across examples from MC, NLI, and HE categories, we observe frequent alternations between correct and incorrect predictions, indicating persistent prediction instability over time.

pute MTV using only the last 20% of checkpoints, in order to focus on fluctuations that remain in the final stages of training, where model performance is typically evaluated and selected in practical settings. Figure 5(a) shows the average MTV over the final training checkpoints for three representative task categories: MC, NLI, and HE. The results reveal no consistent relationship between model size and MTV, suggesting that scaling up model size does not reliably reduce instability in downstream performance.

Although MTV is useful for capturing reference-based score variability, it depends on specific evaluation metrics and does not directly assess how the model's raw outputs change over time. To address this, we introduce a complementary metric based on output similarity. For each example, we compute the dissimilarity between consecutive outputs $x_{\boldsymbol{\theta}_i}$ and $x_{\boldsymbol{\theta}_{i+1}}$ using a similarity function.[6] We define the instability score (IS) as:

$$\mathrm{IS}(\boldsymbol{\Theta}) = \frac{1}{m-1} \sum_{i=1}^{m-1} \left(1 - \mathrm{sim}(x_{\boldsymbol{\theta}_i}, x_{\boldsymbol{\theta}_{i+1}})\right). \tag{2}$$

In contrast to MTV, which reflects correctness with respect to references, this metric directly captures variability in the model's own output behavior, offering a complementary perspective on instability. We compute the IS using the same final 20% of checkpoints as used for MTV, in order to evaluate the remaining output fluctuations at the end of train-

[6] In this paper, we implemented $\mathrm{sim}(\cdot, \cdot)$ as a task-specific score function provided by the llm-jp-eval framework, such as exact match or charF1.

ing. Figure 5(b) presents the average IS by model size. As with the MTV results, IS was not reduced by scaling up model size.

3 Mitigation of Instability Through Post-Processing

In the previous section, we observed that downstream task performance often fluctuates during the pretraining of large language models (LLMs), both at the task and example levels. These fluctuations hinder reliable evaluation and model selection, especially when performance at the final training steps is used to benchmark models. To address this challenge, we investigate post-processing methods that stabilize performance by integrating information from multiple adjacent checkpoints. In the following, we describe two techniques, checkpoint averaging and checkpoint ensemble, provide a theoretical motivation for their use, and empirically evaluate their effectiveness in mitigating instability at both the task and example levels.

3.1 Post-Processing Methods

Checkpoint averaging is a technique in which model weights from multiple adjacent checkpoints are averaged to produce a single inference-time model. At each training step t, we compute the average of the model parameters saved in the most recent n checkpoints: $\bar{\boldsymbol{\theta}}_t = \frac{1}{m}\sum_{i=t-m+1}^{t}\boldsymbol{\theta}_i$. Prior studies have shown that this approach can improve model stability and generalization (Gao et al., 2022). In our experiments, we use window sizes m of 2, 3, 5, 10, and 20.

Checkpoint ensemble is an alternative that retains the original models and aggregates their inference outputs (Chen et al., 2017; Liu et al., 2018). Specifically, we use a majority vote ensemble, where the final prediction is determined by the most frequently predicted label among the constituent checkpoints. Each checkpoint contributes equally. This method is only applicable to multiple-choice tasks (MC, NLI, HE), where the output space consists of a small number of predefined options. We use the same window sizes ($m = 2, 3, 5, 10, 20$) as in averaging. Unlike averaging, checkpoint ensemble does not modify model parameters and is free from artifacts introduced by weight interpolation. However, it incurs a higher inference cost, as it requires separate forward passes for each checkpoint.

3.2 Theoretical Motivation

We now provide a theoretical justification for why checkpoint averaging and ensemble would enhance the stability of downstream performance during training. Let $\boldsymbol{\theta}_i$ denote the model parameters at training step i, and define the average over the most recent n checkpoints as $\bar{\boldsymbol{\theta}}_t = \frac{1}{n}\sum_{i=t-n+1}^{t}\boldsymbol{\theta}_i$. We define $\bar{\boldsymbol{\Theta}} = \{\bar{\boldsymbol{\theta}}_1, \ldots, \bar{\boldsymbol{\theta}}_m\}$ and show $\mathrm{MTV}(\bar{\boldsymbol{\Theta}}) \leq \mathrm{MTV}(\boldsymbol{\Theta})$.

Suppose each $\boldsymbol{\theta}_i$ is close to $\bar{\boldsymbol{\theta}}_t$, we can assume a first-order approximation of the evaluation score[7]:

$$L(\bar{\boldsymbol{\theta}}_t) \approx L(\boldsymbol{\theta}_i) + (\bar{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_i)^\top \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}_i), \quad (3)$$

$$L(\boldsymbol{\theta}_i) \approx L(\bar{\boldsymbol{\theta}}_t) + (\boldsymbol{\theta}_i - \bar{\boldsymbol{\theta}}_t)^\top \nabla_{\boldsymbol{\theta}} L(\bar{\boldsymbol{\theta}}_t). \quad (4)$$

These assumptions imply $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}_i) \approx \nabla_{\boldsymbol{\theta}} L(\bar{\boldsymbol{\theta}}_t)$. By summing Eq. (3) over $i \in \{t-n+1, \ldots, t\}$, we obtain the following equation:

$$\begin{aligned} & nL(\bar{\boldsymbol{\theta}}_t) \\ \approx & \sum_i L(\boldsymbol{\theta}_i) + \left(n\bar{\boldsymbol{\theta}}_t - \sum_i \boldsymbol{\theta}_i\right)^\top \nabla_{\boldsymbol{\theta}} L(\bar{\boldsymbol{\theta}}_t) \\ = & \sum_i L(\boldsymbol{\theta}_i), \end{aligned} \quad (5)$$

since $n\bar{\boldsymbol{\theta}}_t = \sum_i \boldsymbol{\theta}_i$. Thus, the evaluation score at $\bar{\boldsymbol{\theta}}_t$ approximates the average score across the previous checkpoints. The expected value of the evaluation score under majority voting also corresponds to the average score across all models used in the ensemble.

Let $L_t = L(\boldsymbol{\theta}_t)$ and $\bar{L}_t = L(\bar{\boldsymbol{\theta}}_t) \approx \frac{1}{n}\sum_i L_i$. Note that

$$\bar{L}_{t+1} - \bar{L}_t \approx \frac{1}{n}\sum_{i=t-n+1}^{t}\big(L_{i+1} - L_i\big). \quad (6)$$

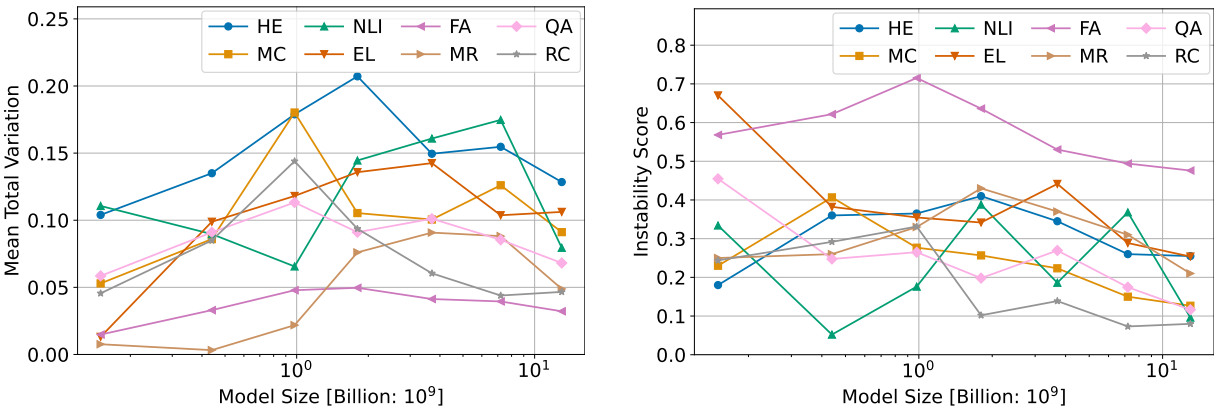
By the triangle inequality,

$$\big|\bar{L}_{t+1} - \bar{L}_t\big| \leq \frac{1}{n}\sum_{i=t-n+1}^{t}\big|L_{i+1} - L_i\big|. \quad (7)$$

[7] Particularly in the final stage of pretraining, which is important in practice, changes in model parameters between adjacent checkpoints are sufficiently small, making this assumption reasonable.

Stabilization method	Win. size	Task category								
		Overall	EL	FA	HE	MC	MR	NLI	QA	RC
NA		$.477_{\pm.006}$	$.403_{\pm.025}$	$.621_{\pm.022}$	$.547_{\pm.021}$	$.507_{\pm.029}$	$.261_{\pm.005}$	$.625_{\pm.018}$	$.579_{\pm.012}$	$.820_{\pm.017}$
Averaging	2	$.479_{\pm.005}$	$.410_{\pm.022}$	$.625_{\pm.018}$	$.549_{\pm.020}$	$.513_{\pm.026}$	$.263_{\pm.005}$	$.628_{\pm.019}$	$.582_{\pm.011}$	$.820_{\pm.017}$
	3	$.480_{\pm.005}$	$.412_{\pm.022}$	$.628_{\pm.017}$	$.551_{\pm.021}$	$.512_{\pm.025}$	$.263_{\pm.004}$	$.628_{\pm.018}$	$.584_{\pm.011}$	$.819_{\pm.016}$
	5	$.481_{\pm.005}$	$.415_{\pm.022}$	$.631_{\pm.016}$	$.551_{\pm.019}$	$.512_{\pm.023}$	$.263_{\pm.004}$	$.628_{\pm.016}$	$.586_{\pm.010}$	$.821_{\pm.014}$
	10	$.483_{\pm.004}$	$.423_{\pm.018}$	$.632_{\pm.016}$	$.553_{\pm.016}$	$.518_{\pm.025}$	$.264_{\pm.004}$	$.628_{\pm.016}$	$.592_{\pm.011}$	$.819_{\pm.014}$
	20	$.484_{\pm.003}$	$.424_{\pm.013}$	$.633_{\pm.013}$	$.551_{\pm.015}$	$.521_{\pm.020}$	$.264_{\pm.003}$	$.630_{\pm.010}$	$.595_{\pm.009}$	$.820_{\pm.010}$
Ensemble	2	-	$.403_{\pm.021}$	$.620_{\pm.018}$	$.547_{\pm.019}$	-	-	-	-	-
	3	-	$.408_{\pm.021}$	$.626_{\pm.018}$	$.549_{\pm.017}$	-	-	-	-	-
	5	-	$.410_{\pm.020}$	$.628_{\pm.015}$	$.549_{\pm.018}$	-	-	-	-	-
	10	-	$.414_{\pm.017}$	$.631_{\pm.013}$	$.548_{\pm.016}$	-	-	-	-	-
	20	-	$.410_{\pm.012}$	$.631_{\pm.015}$	$.547_{\pm.015}$	-	-	-	-	-

Table 1: Effect of stabilization methods on downstream task performance during the training of the 13B model. "Overall" indicates the aggregated score across all task categories. Scores are reported as mean ± standard deviation, computed over the final 20% of training checkpoints. Both checkpoint averaging and ensemble reduce score variance and improve mean performance across many task categories.



(a) Mean total variation (MTV)

(b) Instability score (IS)

Figure 5: Relationship between model size and two measures of downstream performance instability. MTV is computed from reference-based evaluation scores, while IS captures output dissimilarity between consecutive checkpoints. Neither metric shows a consistent decreasing trend with increasing model size, suggesting that scaling does not reliably mitigate instability.

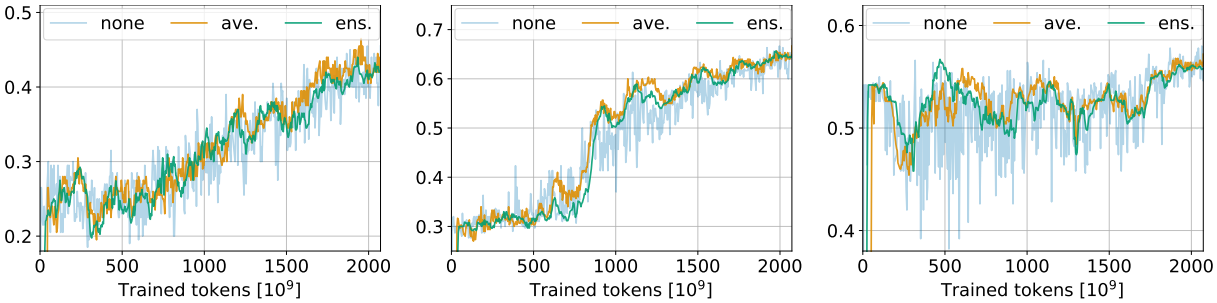
Summing over all t gives

$$\begin{aligned}
\mathrm{MTV}(\bar{\boldsymbol{\Theta}}) &= \frac{1}{m-1}\sum_{t=1}^{m-1}\left|\bar{L}_{t+1}-\bar{L}_t\right| \\
&\le \frac{1}{m-1}\sum_{t=1}^{m-1}\frac{1}{n}\sum_{i=t-n+1}^{t}\left|L_{i+1}-L_i\right| \\
&= \frac{1}{n}\sum_{t=1}^{n}\frac{1}{m-1}\sum_{i=1}^{m-1}\left|L_{i+1}-L_i\right| \\
&= \frac{1}{n}\sum_{t=1}^{n}\mathrm{MTV}(\boldsymbol{\Theta}) \\
&= \mathrm{MTV}(\boldsymbol{\Theta}). \qquad (8)
\end{aligned}$$

3.3 Effect on Task-Level Scores

We first evaluate the impact of stabilization methods on downstream task performance at the category level. Table 1 summarizes the mean and standard deviation of scores for the 13B model across all task categories, measured over the final 20% of training checkpoints. This final-phase evaluation focuses on the later training stages, which are typically used for model selection and benchmark reporting.

Applying checkpoint averaging leads to improvements in both mean scores and stability (i.e., lower standard deviation) in most task categories, including the overall score. These effects become more pronounced with larger averaging window sizes. A similar trend is observed with the checkpoint



(a) HE (Human Examination) (b) MC (Multiple Choice QA) (c) NLI (Natural Language Inference)

Figure 6: Progression of downstream task scores for the 13B model when applying stabilization methods. "none" represents no stabilization method, "ave." represents checkpoint averaging, and "ens." represents checkpoint ensemble scores. The window size for stabilization methods was set at 20.

ensemble, which also reduces score variance and enhances mean accuracy. Both results support our hypothesis that checkpoint integration suppresses short-term fluctuations and improves reliability in evaluation. When comparing the two methods, checkpoint averaging tends to yield greater improvements in mean performance, particularly for larger windows. These trends are consistent across other model sizes as well.

While Table 1 highlights the aggregate improvements in performance and stability, Figure 6 provides a qualitative view of how the stabilization methods affect score trajectories throughout training. The figure visualizes the 13B model's performance in three representative categories (HE, MC, and NLI) under different stabilization settings. With a window size of 20, both checkpoint averaging and ensemble methods visibly suppress score fluctuations across training steps, indicating enhanced stability throughout pretraining.

3.4 Effect on Example-Level Scores

We now examine whether the stabilization effects also hold at the example level. Figure 7 illustrates score trajectories for five representative examples from JCommonsenseQA (MC) before and after applying stabilization methods. The examples were selected based on having the highest MTV under the no-stabilization setting. As shown, both checkpoint averaging and ensemble yield qualitatively smoother predictions, indicating reduced instability across training.

To quantitatively evaluate this effect, we quantify example-level score instability following the same procedure described in § 2.4. Figure 8 shows how the mean total variation (MTV) changes with different window sizes in the MC, NLI, and HE categories.[8] A window size of 1 corresponds to the unstabilized baseline. Both checkpoint averaging and ensemble methods consistently reduce the average MTV, confirming their effectiveness in mitigating example-level instability. We also observe that larger window sizes yield greater reductions in instability. These findings align with our task-level results and highlight the stabilizing effect of checkpoint integration at the granularity of individual performance.

We additionally report results based on the instability score in Appendix B.2. Similar to MTV, both stabilization methods consistently reduced example-level instability across multiple tasks. These results indicate that the effect is not limited to reference-based evaluation metrics but also holds for reference-free assessments of output consistency.

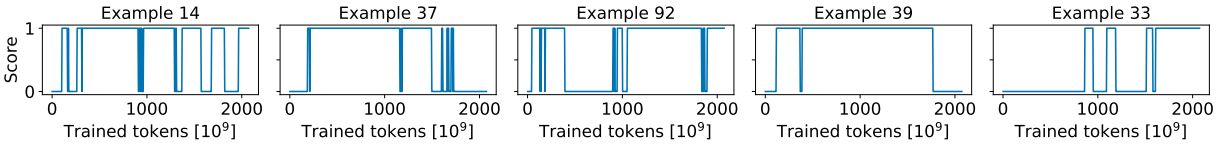
4 Related Work

Stability in Language Model Training. Stability in language model training has primarily been investigated through the behavior of training loss. Challenges such as loss divergence and large fluctuations, as well as inconsistency due to sensitivity to random initialization, have been widely recognized. To address these issues, various architectural and optimization strategies have been proposed to improve convergence and reduce variance across training runs (Chowdhery et al., 2023; Rybakov et al., 2024; Takase et al., 2024). In addition, studies have examined token-level prediction probabilities during training and found that high-frequency tokens tend to be learned more reliably (Chang et al., 2024). However, these studies have mainly

[8] Results for the remaining task categories are presented in Appendix B.1.

(a) No stabilization method applied.

(b) Checkpoint averaging with window size 20.



(c) Checkpoint ensemble with window size 20.

Figure 7: Example-level score trajectories for JCommonsenseQA in the 13B model with and without stabilization methods. The examples shown were selected based on having the highest MTV under the no-stabilization setting.

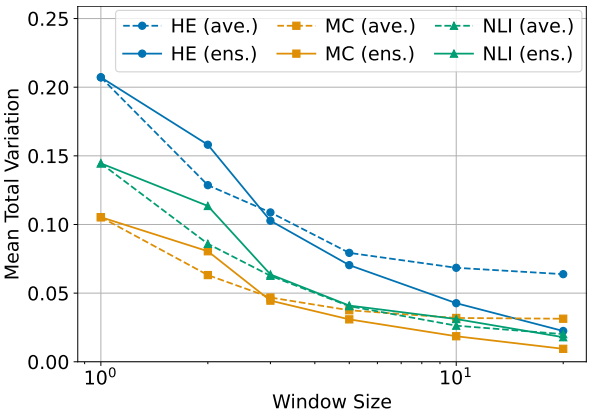


Figure 8: Mean total variation (MTV) for the 13B model in each category. Checkpoint averaging is abbreviated as "ave.", and checkpoint ensemble as "ens."

focused on internal metrics and do not directly address fluctuations in downstream task performance throughout pretraining.

Progression of Downstream Task Performance in LLMs. Several recent studies have tracked how language models acquire capabilities during pretraining by analyzing downstream task performance over time (Biderman et al., 2023; Xia et al., 2023; Yang et al., 2024; Gadre et al., 2024). For example, Biderman et al. (2023) examined the learning curves of models of different sizes in the Pythia suite, while Yang et al. (2024) analyzed example-level learning trajectories to uncover when and how individual examples are learned. Gadre et al. (2024) showed that larger models tend to exhibit smoother progress on evaluation tasks. These works focus on identifying learning phases or scaling behaviors. In contrast, our study concentrates on the short-term instability of downstream task scores, such as frequent reversals in performance, which can occur even in the later stages of pretraining.

Inference Using Multiple Checkpoints. Using multiple checkpoints from a single training run has been proposed as a way to improve robustness and stability. Two common techniques are checkpoint averaging, which averages model weights, and checkpoint ensemble, which aggregates predictions from multiple checkpoints (Junczys-Dowmunt et al., 2016; Chen et al., 2017; Liu et al., 2018). These methods are primarily used to improve the final performance of natural language processing models. Notably, Gao et al. (2022) reported that applying checkpoint averaging to neural machine translation models also led to improved stability in translation quality. However, the effectiveness of such checkpoint-based integration methods for checkpoints during the pretraining of large language models remains largely unexplored. Our work addresses this gap by empirically demonstrating that simple checkpoint integration strategies can effectively mitigate downstream performance instability during LLM pretraining, without requiring any changes to the training procedure.

5 Conclusion

This study analyzed the instability of downstream task performance during the pretraining of large language models, focusing on performance fluctuations across checkpoints. We found that such instability occurs widely across tasks and model sizes. These findings suggest that downstream performance can fluctuate significantly even in the later stages of training, which can undermine the reliability and reproducibility of evaluation results.

To address this, we investigated post-processing approaches that integrate multiple checkpoints. Both checkpoint averaging and ensemble methods effectively reduced performance fluctuations at both the task and example levels, without modifying the training procedure. While checkpoint averaging contributed more to improving average scores, checkpoint ensemble was more effective in stabilizing example-level predictions. Given its lower inference cost, checkpoint averaging is particularly well-suited for practical deployment.

Limitations

Our work demonstrates that instability in downstream task performance is prevalent during LLM pretraining, and that simple checkpoint integration methods can effectively mitigate this issue at both the task and example levels, without requiring changes to the training procedure. However, to further improve robustness and generality, there remain several limitations to be addressed. First, the integration methods explored in this work do not incorporate model confidence or prediction uncertainty, which could offer additional gains in stabilization. Second, our analysis focuses on models trained on a single multilingual corpus and does not extend to instruction-tuned or alignment-based models.

Expanding the scope of checkpoint integration to include adaptive strategies, such as confidence-weighted averaging, and evaluating models across diverse pretraining regimes would be valuable next steps. Additionally, analyzing how output instability relates to prediction probability dynamics and internal representations could offer deeper insights into the nature of learning fluctuations. Understanding whether the observed instability persists or transforms in downstream-tuned models also remains an important open question.

Acknowledgments

This work was supported by the "R&D Hub Aimed at Ensuring Transparency and Reliability of Generative AI Models" project of the Ministry of Education, Culture, Sports, Science and Technology.

References

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Tyler A. Chang, Zhuowen Tu, and Benjamin K. Bergen. 2024. Characterizing learning curves during language model pre-training: Learning, forgetting, and stability. *Transactions of the Association for Computational Linguistics*, 12:1346–1362.

Hugh Chen, Scott Lundberg, and Su-In Lee. 2017. Checkpoint ensembles: Ensemble methods from a single training process. *arXiv preprint arXiv:1710.03282*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sashank Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and 48 others. 2023. Palm: scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(1).

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Jenia Jitsev, Luca Soldaini, Alexandros G. Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, and 6 others. 2024. Language models scale reliably with over-training and on downstream tasks. *arXiv preprint arXiv:2403.08540*.

Yingbo Gao, Christian Herold, Zijian Yang, and Hermann Ney. 2022. Revisiting checkpoint averaging for neural machine translation. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 188–196, Online only. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Kaito Horio, Eiki Murata, Hao Wang, Tatsuya Ide, Daisuke Kawahara, Takato Yamazaki, Kenta Shinzato, Akifumi Nakamachi, Shengzhe Li, and Toshinori Sato. 2023. Verification of chain-of-thought prompting in japanese. *Proceedings of the Annual Conference of JSAI*, JSAI2023:3T1GS602–3T1GS602. In Japanese.

Ai Ishii, Naoya Inoue, Hisami Suzuki, and Satoshi Sekine. 2024. JEMHopQA: Dataset for Japanese explainable multi-hop question answering. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9515–9525, Torino, Italia. ELRA and ICCL.

Marcin Junczys-Dowmunt, Tomasz Dwojak, and Hieu Hoang. 2016. Is neural machine translation ready for deployment? a case study on 30 translation directions. In *Proceedings of the 13th International Conference on Spoken Language Translation*, Seattle, Washington D.C. International Workshop on Spoken Language Translation.

Ai Kawazoe, Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki. 2015. A methodology for constructing inference problem sets based on formal semantic studies. *Proceedings of the Annual Conference of JSAI*, JSAI2015:1K31–1K31.

Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. 2022. JGLUE: Japanese General Language Understanding Evaluation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2957–2966, Marseille, France. European Language Resources Association.

Yuchen Liu, Long Zhou, Yining Wang, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2018. A comparable study on model averaging, ensembling and reranking in nmt. In *Natural Language Processing and Chinese Computing*, pages 299–308.

Kazumasa Omura, Daisuke Kawahara, and Sadao Kurohashi. 2020. A Method for Building a Commonsense Inference Dataset based on Basic Events. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2450–2460, Online. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A Neural Framework for MT Evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Oleg Rybakov, Mike Chrzanowski, Peter Dykas, Jinze Xue, and Ben Lanir. 2024. Methods of improving llm training stability. *arXiv preprint arXiv:2410.16682*.

Tomoki Sugimoto, Yasumasa Onoe, and Hitomi Yanaka. 2023. Jamp: Controlled Japanese Temporal Inference Dataset for Evaluating Generalization Capacity of Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 57–68, Toronto, Canada. Association for Computational Linguistics.

Sho Takase, Shun Kiyono, Sosuke Kobayashi, and Jun Suzuki. 2024. Spike no more: Stabilizing the pre-training of large language models. *arXiv preprint arXiv:2312.16903*.

Ye Kyaw Thu, Win Pa Pa, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Introducing the Asian Language Treebank (ALT). In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1574–1578, Portorož, Slovenia. European Language Resources Association (ELRA).

Mengzhou Xia, Mikel Artetxe, Chunting Zhou, Xi Victoria Lin, Ramakanth Pasunuru, Danqi Chen, Luke Zettlemoyer, and Veselin Stoyanov. 2023. Training trajectories of language models across scales. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13711–13738, Toronto, Canada. Association for Computational Linguistics.

Hitomi Yanaka and Koji Mineshima. 2021. Assessing the Generalization Capacity of Pre-trained Language Models through Japanese Adversarial Natural Language Inference. In *Proceedings of the 2021 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP (BlackboxNLP2021)*.

Hitomi Yanaka and Koji Mineshima. 2022. Compositional Evaluation on Japanese Textual Entailment and Similarity. *Transactions of the Association for Computational Linguistics*, 10:1266–1284.

Chen Yang, Junzhuo Li, Xinyao Niu, Xinrun Du, Songyang Gao, Haoran Zhang, Zhaoliang Chen, Xingwei Qu, Ruibin Yuan, Yizhi Li, Jiaheng Liu, Stephen W. Huang, Shawn Yue, and Ge Zhang. 2024. The fine line: Navigating large language model pre-training with down-streaming capability analysis. *arXiv preprint arXiv:2404.01204*.

Ziqi Yin, Hao Wang, Kaito Horio, Daisuke Kawahara, and Satoshi Sekine. 2024. Should we respect LLMs? a cross-lingual study on the influence of prompt politeness on LLM performance. In *Proceedings of the Second Workshop on Social Influence in Conversations (SICon 2024)*, pages 9–35, Miami, Florida, USA. Association for Computational Linguistics.

A Detailed Settings

Table 2 summarizes the number of parameters, batch sizes, maximum sequence lengths, and the training steps at which checkpoints were obtained for each model. The checkpoints were saved at regular intervals ranging from several hundred to approximately a thousand steps.

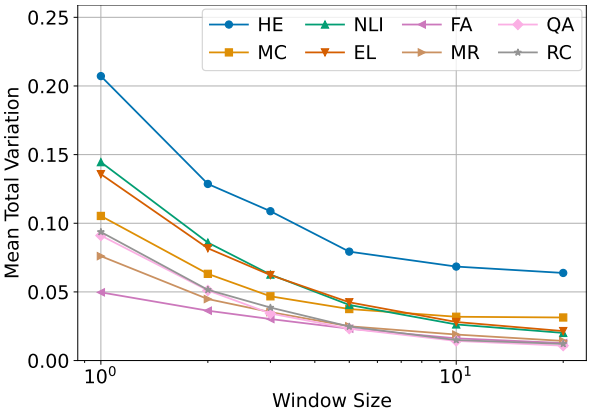


Figure 9: Mean total variation for the 13B model across all task categories using checkpoint averaging. This figure complements the main text by providing results for the remaining categories and enabling direct comparison with HE, MC, and NLI.

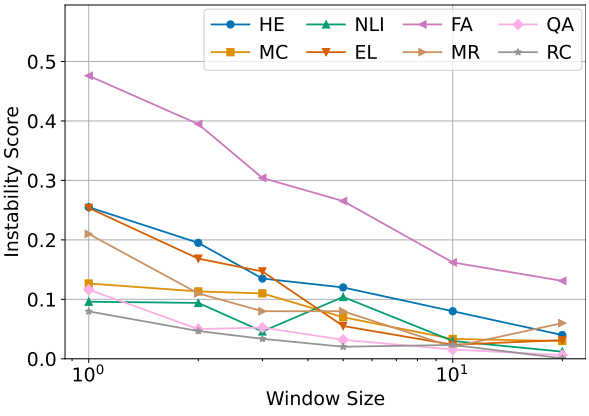


Figure 10: Instability score for the 13B model across different task categories using checkpoint averaging. Larger window sizes consistently reduce output instability.

Table 3 lists all downstream task categories, datasets, and the evaluation metrics used. Each task category typically comprises multiple datasets. For example, the MC (multiple choice question answering) category includes JCommonsenseQA, JCommonsenseMorality, and KUCI.

B Additional Results

B.1 Mean Total Variation for All Task Categories

To supplement the main results shown in Figure 8, we report the mean total variation (MTV) across different window sizes for all task categories using the 13B model. As described in § 2.4, MTV quantifies the average fluctuation in example-level scores across checkpoints, with lower values indicating greater temporal stability in predictions.

While Figure 8 focused on three representative categories (HE, MC, and NLI), Figure 9 presents a complementary overview that includes the remaining six categories alongside the previously shown ones for comparison. Since the results of checkpoint ensemble have already been presented for all categories, this figure exclusively reports the results of checkpoint averaging.

The trends observed here are consistent with the findings in the main text: increasing the averaging window size reduces MTV across categories, indicating the stabilizing effect of checkpoint averaging during LLM pretraining.

B.2 Instability Score Across Tasks

In addition to the MTV results presented in the main text, we report complementary results using the instability score (IS) introduced in § 2.4. Unlike MTV, which measures reference-based score fluctuations, IS directly evaluates the variability of model outputs by computing dissimilarity between consecutive predictions.

Figures 10 and 11 show the IS values for the 13B model across multiple task categories using checkpoint averaging and checkpoint ensemble, respectively. Both figures report the average IS over the final 20% of checkpoints, which captures instability during the final stage of pretraining, an essential phase for model selection in practical applications.

Figure 10 illustrates the effect of checkpoint averaging across different window sizes. We observe a consistent reduction in IS as the window size increases, confirming that averaging helps smooth output transitions across checkpoints.

Figure 11 shows the results for checkpoint ensemble. This figure includes the corresponding IS values for checkpoint averaging as a baseline for comparison. We find that ensemble methods also reduce instability and yield comparable trends to averaging, further validating the effectiveness of checkpoint integration strategies.

Together, these results confirm that both checkpoint averaging and ensemble reduce output-level fluctuations as measured by IS, and that the effect extends beyond reference-based metrics to general output consistency.

Model size	Checkpointed steps (total)	Batch size	Max tokens
150M	0, 1, ⋯, 9, 10, ⋯, 90, 100, ⋯, 900, 1000, ⋯, 988000, 988240 (1,017 total)	512	4,096
440M	0, 1, ⋯, 9, 10, ⋯, 90, 100, ⋯, 900, 1000, ⋯, 988000, 988240 (1,017 total)	512	4,096
980M	0, 1, ⋯, 9, 10, ⋯, 90, 100, ⋯, 900, 1000, ⋯, 988000, 988240 (1,017 total)	512	4,096
1.8B	0, 1, ⋯, 9, 10, ⋯, 90, 100, ⋯, 900, 1000, ⋯, 988000, 988240 (1,017 total)	512	4,096
3.7B	0, 1, ⋯, 9, 10, ⋯, 90, 100, ⋯, 900, 1000, ⋯, 494000, 494120 (523 total)	1,024	4,096
7.3B	0, 1, ⋯, 9, 10, ⋯, 90, 100, ⋯, 900, 1000, ⋯, 494000, 494120 (523 total)	1,024	4,096
13B	0, 1, ⋯, 9, 10, ⋯, 90, 100, ⋯, 900, 1000, ⋯, 494000, 494120 (523 total)	1,024	4,096

Table 2: Training configurations and saved checkpoints for each model size.

Task category	Dataset	Sub task	Evaluation metric
NLI (natural language inference)	Jamp	-	Exact match
	JaNLI	-	Exact match
	JNLI	-	Exact match
	JSeM	-	Exact match
	JSICK	-	Exact match
QA (question answering)	JEMHopQA	-	Char. F1
	NIILC	-	Char. F1
RC (reading comprehension)	JSQuAD	-	Char. F1
MC (multiple choice question answering)	JCommonsenseMorality	-	Exact match
	JCommonsenseQA	-	Exact match
	KUCI	-	Exact match
EL (entity linking)	chABSA	-	Set F1
FA (fundamental analysis)	Wikipedia Annotated Corpus	NER	Set F1
		PAS	Set F1
		Coreference	Set F1
		Dependency	Set F1
		Reading	Char. F1
MR (mathematical reasoning)	MAWPS	-	Exact match
MT (machine translation)	ALT	Ja→En	COMET (Rei et al., 2020)
		En→Ja	COMET
	WikiCorpus	Ja→En	COMET
		En→Ja	COMET
HE (human examination)	MMLU	-	Exact match
	JMMLU	-	Exact match

Table 3: Task categories, datasets, subtasks, and evaluation metrics used in the llm-jp-eval framework. Each category may include multiple datasets, each evaluated with a task-specific metric.

C Used Data and Software

C.1 Data

Jamp created by Sugimoto et al. (2023). License: CC BY-SA 4.0. https://github.com/tomo-ut/temporalNLI_dataset

JaNLI created by Yanaka and Mineshima (2021). License: CC BY-SA 4.0. https://github.com/verypluming/JaNLI

JNLI created by Kurihara et al. (2022). License: CC BY-SA 4.0. https://github.com/yahoojapan/JGLUE

JSeM created by Kawazoe et al. (2015). License: Available for research use. https://github.com/DaisukeBekki/JSeM

JSICK created by Yanaka and Mineshima (2022). License: CC BY 4.0. https://github.com/verypluming/JSICK

JEMHopQA created by Ishii et al. (2024). License: CC BY-SA 4.0. https://github.com/aiishii/JEMHopQA

NIILC created by Sekine et al. License: CC BY-SA 4.0. https://mynlp.is.s.u-tokyo.ac.jp/niilc-qa/

JSQuAD created by Kurihara et al. (2022). License: CC BY-SA 4.0. https://github.com/yahoojapan/JGLUE

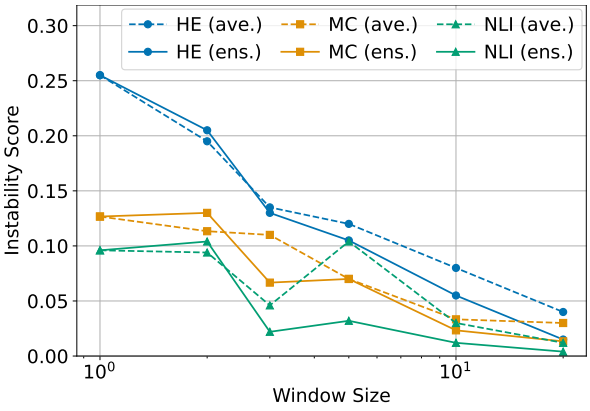


Figure 11: Instability score for the 13B model using checkpoint ensemble. For comparison, corresponding results for checkpoint averaging are also shown.

JCommonsenseMorality created by Takeshita et al. License: MIT. https://github.com/Language-Media-Lab/commonsense-moral-ja

JCommonsenseQA created by Kurihara et al. (2022). License: CC BY-SA 4.0. https://github.com/yahoojapan/JGLUE

KUCI created by Omura et al. (2020). License: CC BY-SA 4.0. https://github.com/ku-nlp/KUCI

chABSA created by Kubo et al. License: CC BY 4.0. https://github.com/chakki-works/chABSA-dataset

Wikipedia Annotated Corpus created by Kawahara et al. License: CC BY-SA 4.0. https://github.com/ku-nlp/WikipediaAnnotatedCorpus

MAWPS created by Horio et al. (2023). License: Apache-2.0 license. https://github.com/nlp-waseda/chain-of-thought-ja-dataset

Asian Language Treebank (ALT) Parallel Corpus created by Thu et al. (2016). License: CC BY-NC-SA 4.0. https://www2.nict.go.jp/astrec-att/member/mutiyama/ALT/

WikiCorpus created by NICT. License: CC BY-SA 3.0. https://alaginrc.nict.go.jp/WikiCorpus/

MMLU created by Hendrycks et al. (2021). License: MIT. https://github.com/hendrycks/test

JMMLU created by Kawazoe et al. License: CC BY-SA 4.0. https://github.com/nlp-waseda/JMMLU

LLM-jp Corpus v3 created by LLM-jp. License: Available for research use. https://gitlab.llm-jp.nii.ac.jp/datasets/llm-jp-corpus-v3/

C.2 Software

llm-jp-eval created by Han et al. License: Apache-2.0 license. https://github.com/llm-jp/llm-jp-eval