Length Controlled Generation for Black-box LLMs

Anonymous ACL submission

Abstract

Large language models (LLMs) have demonstrated impressive instruction following capabilities, while still struggling to accurately manage the length of the generated text, which is a fundamental requirement in many real-world applications. Existing length control methods involve fine-tuning the parameters of LLMs, which is inefficient and suboptimal for practical use. In this paper, we propose a novel iterative sampling framework for text length control, integrating the Metropolis-Hastings algorithm with an importance sampling acceleration strategy. This framework efficiently and reliably regulates LLMs to generate length-constrained text without modifying the underlying parameters, thereby preserving the original capabilities of LLMs. Experimental results demonstrate that our framework achieves almost 100% success rates of length control on LLAMA3.1 for tasks such as length-controlled abstractive summarization and length-constrained instruction following, with minimal additional computational overhead. This also highlights the significant potential of our method for precise length control across a broader range of applications, without compromising the versatility of LLMs.

1 Introduction

Recent advancement of pre-trained large language models (LLMs) has significantly improved the performance of various natural language processing tasks (Vaswani, 2017; Devlin, 2018; Brown, 2020). LLMs such as GPT-4 (Achiam et al., 2023) and LLAMA (Touvron et al., 2023a,b; Dubey et al., 2024) exhibit exceptional capabilities to follow instructions (Ouyang et al., 2022), allowing them to generate text aligning closely with user intentions. Applications such as dialogue generation (Yi et al., 2024), code completion (Jiang et al., 2024), and reasoning (Plaat et al., 2024) have benefited greatly from these advances, establishing LLMs as the core component in building general AI systems.

Despite the strong generative capability, LLMs still struggle to precisely manage the length of generated text (Wang et al., 2024; Huang et al., 2024; Li et al., 2024), due to inherent architectural limitations such as subword tokenization (Sennrich, 2015; Devlin, 2018) and autoregressive decoding (Sutskever, 2014; Vaswani, 2017; Brown, 2020). This issue is critical because length control is a fundamental requirement in many real-world applications. For example, summarization tasks often require outputs of specific lengths to balance informativeness and conciseness (Fan et al., 2017; Liu et al., 2018, 2022; Jie et al., 2024). In addition, LLM-based chatbots favor longer responses due to the length bias introduced in pairwise preference optimization (Singhal et al., 2023), which undermines the fairness of model evaluation (Dubois et al., 2024a; Yuan et al., 2024) and degrades the user experience in practical conversations.

To address the issue of length control, various methods have been proposed, including fine-tuning based on specifically designed length instructions (Yuan et al., 2024; Wang et al., 2024; Li et al., 2024) and reinforcement learning with length feedback (Stiennon et al., 2020; Jie et al., 2024). However, we argue that it is necessary to design length control methods tailored for black-box LLMs for the following reasons: (1) Fine-tuning LLMs specifically for length control requires extensive computational resources and can degrade their general-purpose utility (Lin et al., 2024). Worse still, not all LLMs are open source. The fine-tuning methods cannot be applied to black-box LLMs. (2) Length control has been actually considered in the instruction tuning phase of LLMs (Wang et al., 2022; Taori et al., 2023). As such, a superior and more efficient solution is to activate the inherent length-following capabilities within LLMs rather than undertaking a costly retraining process.

We propose a novel framework for black-box LLMs that operates length control without the need for parameter training. The length controlled generation can be viewed as sampling from a target distribution, which is influenced simultaneously by the length constraint and language probability. However, it is intractable to directly sample from this distribution, and we utilize an iterative sampling framework called Metropolis-Hastings (Metropolis et al., 1953; Hastings, 1970), which is a classic and prevalent Markov chain Monte Carlo (MCMC) method specially suited for this complex scenario. In detail, our framework initiates from the original output of LLM and iteratively produces candidate outputs conditioned on the previous ones via a proposal distribution. The acceptance or rejection of these candidates is determined by their comparative advantage over previous candidates, which is quantified as an acceptance distribution that involves: the alignment with the target length, the generative probability density of the LLM, and the probability density of the proposal distribution. Furthermore, we leverage importance sampling (Kahn and Marshall, 1953; Owen and Zhou, 2000) in the proposal distribution to accelerate the iteration process, where candidates with lengths closer to the desired target are more likely to be sampled. We treat the LLM as an immutable component, enabling the integration of effective length control mechanisms across the broadest possible spectrum of LLMs.

We assess the effectiveness of our method on several tasks, including the abstract text summarization task with precise length control and the instruction following task with maximum length constraint. Experimental results demonstrate that our black-box approach significantly improves existing LLMs in length control and achieves the state-of-the-art performance without compromising the quality of generated contents. Specifically, in the case of the LLAMA3.1 model (Dubey et al., 2024), our method achieves success rates close to 100% of the length control in only five iterations at most, highlighting its efficiency and practicality. Our contributions are summarized as follows:

1. We propose a novel framework for black-box LLMs, offering more flexible and general length control compared to existing methods.
2. We introduce an innovative integration of the classic Metropolis-Hastings algorithm with modern LLMs, thereby enhancing the efficiency and precision of length control.
3. We achieve remarkable length control performance in advanced LLMs, showcasing the robustness and effectiveness of our framework.

2 Related Work

2.1 Instruction Following

LLMs are endowed with powerful instruction following capabilities in the supervised fine-tuning stage (Ouyang et al., 2022; Zhou et al., 2024). Despite being able to understand human instructions and handle a broad spectrum of tasks, LLMs still leave a large room for improvement in their instruction following capabilities (Liu et al., 2023). In addition to training stronger instruction following capabilities (Rafailov et al., 2024b,a), it is also important to better utilize and activate the power of LLMs (Wei et al., 2022; Yao et al., 2023).

2.2 Length Control

Controlling the output length is a crucial skill in text generation, particularly for tasks where lengths vary significantly. Early length controllable generation methods focus on the abstractive summarization task. For example, some methods discretize lengths into bins with specialized tokens (Fan et al., 2017), introduce length constraint factors to convolutional blocks (Liu et al., 2018), or optimize output quality through minimum risk training (Makino et al., 2019). In addition, length control signals can be incorporated in positional encodings (Takase and Okazaki, 2019), attention units (Yu et al., 2021; Liu et al., 2022), and natural language instructions (Yuan et al., 2024; Wang et al., 2024; Jie et al., 2024; Li et al., 2024). These methods require the length training, which is inefficient when applied to LLMs and has the potential to damage general abilities. Recent instruction-based control schemes (Juseon-Do et al., 2024) can alleviate these issues but still fail to achieve precise control. In contrast, our framework effectively enhances LLM's ability to enable more precise text generation.

3 Methodology

3.1 Overall Framework: Metropolis-Hastings

As illustrated in Figure 1, we introduce how to apply the Metropolis-Hastings framework (Metropolis et al., 1953; Hastings, 1970) to the length control scenario. Given the probability distribution of LLMs $P(y|x)$ and the score of length constraint $f(y)$, our target distribution $\pi(y|x)$ is derived as:

$$\pi(y|x) = \frac{f(y)P(y|x)}{\int f(y)P(y|x)\mathrm{d}y}, \quad (1)$$

where x is the human instruction and y is the response of the target LLM. We cannot directly sam-

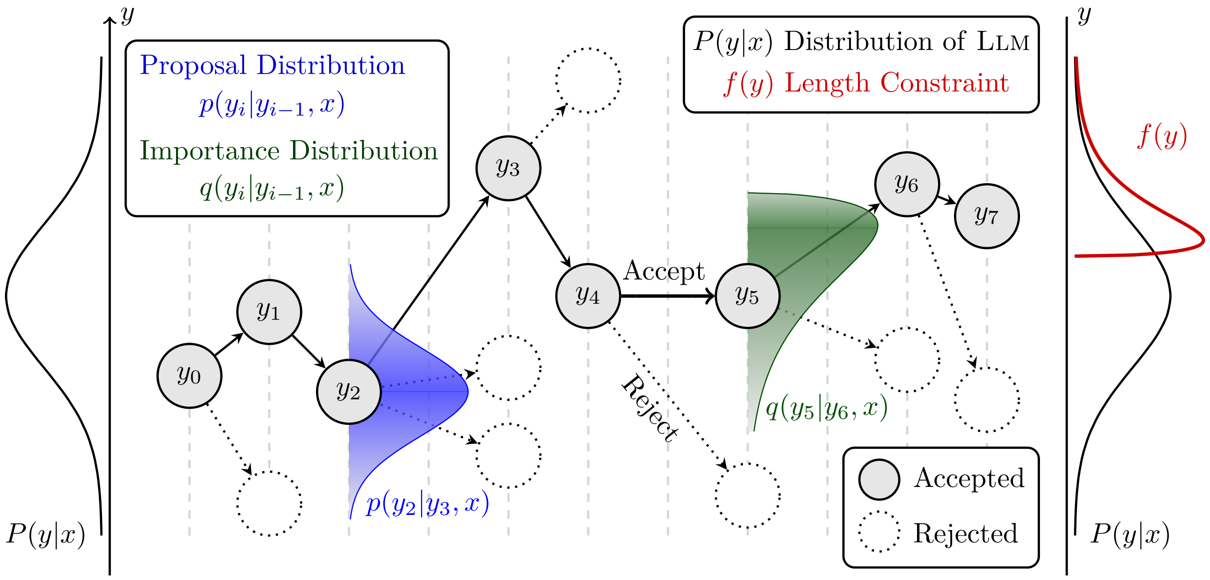

Figure 1: The overall sampling process of our Metropolis-Hastings framework. The iteration starts by sampling an initial state from the distribution of LLM $y_0 \sim P(y|x)$, and ends at y_7, which maximizes the target combination of length constraints and probability densities $\pi(y|x) \propto f(y)P(y|x)$. During each iteration, a new candidate content y_i is generated based on the previous one y_{i-1} via the proposal distribution $p(y_i|y_{i-1},x)$. The generated candidate y_i will be either accepted or rejected considering the degree to which the target objectives are satisfied. We enhance the original proposal distribution by incorporating length constraints, yielding the importance distribution $q(y_i|y_{i-1},x)$, which increases the acceptance rate of candidates and significantly improves the iteration efficiency.

ple y from the target distribution $\pi(y|x)$ because: (1) $f(y)$ is a deterministic function designed to evaluate length constraints, rather than a probability distribution, which is not suitable for sampling; and (2) the integral of the normalization constant $Z = \int f(y)P(y|x)\mathrm{d}y$ is intractable.

The Markov chain Monte Carlo algorithms can handle the problem by starting from an initial state $y_0 \sim P(y|x)$, iteratively generating a collection of states $[y_1, \ldots, y_n]$ with a transition distribution $P(y_i|y_{i-1},x)$, and approaching the target distribution $\pi(y|x) = \lim_{n\to\infty} P(y_0|x) \prod_{i=1}^{n} P(y_i|y_{i-1},x)$. Therefore, y_n can be considered as sampled from the target distribution $\pi(y|x)$ when $n \to \infty$.

The Metropolis-Hastings algorithm designs the transition probability as a combination of two steps:

$$P(y_i|y_{i-1},x) = p(y_i|y_{i-1},x)\mathcal{A}(y_{i-1} \to y_i), \quad (2)$$

where $p(y_i|y_{i-1},x)$ is the proposal distribution that generates a new candidate y_i given the previous one y_{i-1}. The acceptance distribution $\mathcal{A}(y_{i-1} \to y_i)$ provides the probability of accepting the proposed candidate y_i. To ensure convergence, $\pi(y|x)$ must be the unique stationary distribution of the Markov chain. Thus the Metropolis-Hastings algorithm further requires the transition probability $P(y_i|y_{i-1},x)$ to fulfill the detailed balance condition, which is a sufficient condition for the stationary distribution,

$$\pi(y_{i-1}|x)P(y_i|y_{i-1},x) = \pi(y_i|x)P(y_{i-1}|y_i,x). \quad (3)$$

Based on eqs. (2) and (3), the acceptance distribution is derived to satisfy the following constraint:

$$\begin{aligned} \frac{\mathcal{A}(y_{i-1} \to y_i)}{\mathcal{A}(y_i \to y_{i-1})} &= \frac{\pi(y_i|x)p(y_{i-1}|y_i,x)}{\pi(y_{i-1}|x)p(y_i|y_{i-1},x)} \\ &= \frac{f(y_i)P(y_i|x)p(y_{i-1}|y_i,x)}{f(y_{i-1})P(y_{i-1}|x)p(y_i|y_{i-1},x)}, \end{aligned} \quad (4)$$

where the normalization constant Z cancels, making subsequent calculations convenient. In addition, the most popular choice of $\mathcal{A}(y_{i-1} \to y_i)$ in Metropolis-Hastings that satisfies eq. (4) is:

$$\min\left(1, \frac{f(y_i)P(y_i|x)p(y_{i-1}|y_i,x)}{f(y_{i-1})P(y_{i-1}|x)p(y_i|y_{i-1},x)}\right), \quad (5)$$

The sampling process of Metropolis-Hastings is illustrated in algorithm 1. During each iteration loop, a new candidate y_i is generated from the previous one y_{i-1}. Whether to accept or reject the new candidate is determined by the acceptance distribution $\mathcal{A}(y_{i-1} \to y_i)$, where the randomness is achieved with a uniform distribution $u \sim \mathcal{U}(0,1)$.

In the black-box setting where direct access to the internal probability outputs of LLM is not

Algorithm 1 Metropolis-Hastings Algorithm

```
Initialize the start state $y_0 \sim P(y|x)$
for $i = 0$ to $n$ do
    Propose: $y_i \sim p(y_i|y_{i\text{-}1}, x)$
    Calculate: $\mathcal{A}(y_{i\text{-}1} \to y_i)$          // eq. (5)
    Randomize: $u \sim \mathcal{U}(0, 1)$
    if $u > \mathcal{A}(y_{i\text{-}1} \to y_i)$ then
        $y_i = y_{i\text{-}1}$                              // Reject
    end if                                             // else Accept
end for
Return $y_n$
```

available, the following key components of the Metropolis-Hastings algorithm require careful and specialized designs: (1) the length constraint score $f(y)$ (§3.2), which serves as a quantitative metric to assess the degree to which the generated samples adhere to predefined length requirements; (2) the probability distribution of LLM $P(y|x)$ (§3.3), which estimates the likelihood of the LLM generating the specific sample y; and (3) the proposal distribution $p(y_i|y_{i\text{-}1}, x)$ (§3.4), which requires tailored construction to effectively generate candidate samples and efficiently explore the solution space, ensuring adherence to the length constraint while achieving sufficiently high generation quality.

3.2 Length Constraint Score $f(y)$

Given a word counter $\texttt{Len}(\cdot)$, such as the NLTK word tokenization function (Bird et al., 2009), the deviation of the LLM generated contents y from the target length ℓ can be quantitatively measured using the Manhattan distance, which is:

$$\mathscr{D}(y, \ell) = |\,\texttt{Len}(y) - \ell\,|. \tag{6}$$

Considering the target of our sampling process, the combination of $f(y)P(y|x)$, we observe a resemblance to the constrained optimization problem, where $\log f(y)$ can be interpreted as a constraint and $\log P(y|x)$ serves as an objective to be maximized. Our target is similar to a Lagrangian function $f(y)P(y|x) \propto \log P(y|x) + \lambda \log f(y)$ with the Lagrange multiplier $\lambda = 1$. Furthermore, our sampling process can be seen as analogous to exterior optimization, where the proposed candidates y_i are always unable to satisfy the constraints during each iteration prior to the termination of the loop. Therefore, the constraint function $f(y)$ needs to incentivize candidates that approximate the length constraint, and terminate the iterations with a significantly large reward when the length constraint is satisfied. Thus our length constraint score is defined as follows. For an exact target length $\ell > 0$:

$$f(y) = \frac{1}{\mathscr{D}(y, \ell)}. \tag{7}$$

For an interval of target length $[\ell_1, \ell_2]$:

$$f(y) = \begin{cases} \dfrac{1}{\mathscr{D}(y, \ell_1)}, & y < \ell_1 \\ +\infty, & \ell_1 \le y \le \ell_2 \\ \dfrac{1}{\mathscr{D}(y, \ell_2)}, & y > \ell_2 \end{cases}, \tag{8}$$

where $0 < \ell_1 < \ell_2 \le +\infty$ and ℓ_2 can be $+\infty$.

3.3 Probability Distribution of LLM $P(y|x)$

We can obtain the responses generated by black-box LLMs such as GPT-4 (Achiam et al., 2023), i.e., sampling from their distributions $y \sim P(y|x)$,

Prompt Template: $y_0 \sim P(y|x)$

`[USER]:` Answer the following instruction using $\{\ell\}$ words or less: $\{x\}$

`[ASSISTANT]:` Answer: $\{y_0\}$

which is easily accomplished with a simple instruction. However, we are unable to access their internal parameters or underlying probability distributions. Consequently, it is intractable to verify the probability density $P(y|x)$ of specific samples y.

To address this issue, we employ the LLM-as-a-Judge approach (Chiang et al., 2023; Zheng et al., 2023; Dubois et al., 2024b) as a solution. Leveraging the advanced understanding, reasoning, and mathematical capabilities of the model, we require LLMs to score samples generated by themselves, thus implicitly estimating their probability density distributions. Besides, we predefine a series of perspectives to unify the scoring mode for our tasks. For the abstractive summarization task, we measure the *information coverage*, *linguistic fluency*, *conciseness*, *logical coherence*, and *faithfulness* of the generated summaries. For instruction following, we measure the response with *helpfulness*, *relevance*, *accuracy*, *depth*, *creativity*, and *level of detail*. Denoting the score function as $\phi(y|x)$, we get the estimated probability distribution as:

$$P(y|x) \simeq \frac{\phi(y|x)}{\int \phi(y|x)\mathrm{d}y}. \tag{9}$$

Similarly to the eq. (4), we can cancel the calculation of the normalization constant $\int \phi(y|x)\mathrm{d}y$.

Since the eq. (4) requires the calculation of the ratio $P(y_i|x) \div P(y_{i\text{-}1}|x)$, where y_i and $y_{i\text{-}1}$ are both sampled from the target LLM, we can further refine the scoring function by employing a pairwise comparison function $\Phi(y_i, y_{i\text{-}1}|x)$ that

$$\frac{P(y_i|x)}{P(y_{i\text{-}1}|x)} \simeq \Phi(y_i, y_{i\text{-}1}|x) \simeq \frac{\phi(y_i|x)}{\phi(y_{i\text{-}1}|x)}, \qquad (10)$$

where the pairwise score can discern subtle differences between the sample pair $(y_i, y_{i\text{-}1})$ at adjacent iteration steps. In addition, the pairwise function will produce scores with less fluctuation than the absolute one $\phi(y|x)$ (Zheng et al., 2023).

3.4 Proposal Distribution $p(y_i|y_{i\text{-}1}, x)$

The proposal distribution $p(y_i|y_{i\text{-}1}, x)$ plays a pivotal role, as it directly influences the efficiency and quality of the sampling process. Since the probability distribution of LLM, $P(y|x)$, is estimated by itself (§3.3), requiring LLMs to further approximate the probability density of the proposal distribution $p(y_i|y_{i\text{-}1}, x)$ not only introduces additional complexity, but also amplifies the estimation errors. Therefore, we impose a symmetry constraint (Chib and Greenberg, 1995; Haario et al., 2001) on the design of the proposal distribution, which is $q(y_i|y_{i\text{-}1}, x) = q(y_{i\text{-}1}|y_i, x)$. And the acceptance distribution $\mathcal{A}(y_{i\text{-}1} \to y_i)$ (eq. (5)) reduces to

$$\min\left(1, \frac{f(y_i)P(y_i|x)}{f(y_{i\text{-}1})P(y_{i\text{-}1}|x)}\right), \qquad (11)$$

when current result is better than the previous one, we directly accept it. Otherwise, we accept with a probability equal to its ratio, enabling expansion of the sampling space. Therefore, we design time-unbiased instructions to make LLM satisfy the symmetry constraints as much as possible. The detailed prompt template for LLMs is as follows:

Prompt Template: $y_i \sim p(y_i|y_{i\text{-}1}, x)$

`[USER]`: Answer the following instruction using $\{\ell\}$ words or less: $\{x\}$
`[ASSISTANT]`: Answer: $\{y_{i\text{-}1}\}$

`[USER]`: Please generate a new answer based on the previous one:
`[ASSISTANT]`: Answer: $\{y_i\}$

where y_i and $y_{i\text{-}1}$ are equivalent and interchangeable in the semantics of this template.

However, this preliminary Metropolis-Hastings framework, constructed with the current proposal function $p(y_i|y_{i\text{-}1}, x)$, is not efficient due to following reasons. (1) Intuitively, when generating new candidates, the length signal remains the initial one ("*using ℓ words or less*"). Without introducing updated length signals, LLM may remain trapped in its own errors, unable to converge to improved solutions. (2) From a theoretical perspective, the sampling efficiency and quality will be maximized when the proposal function $p(y_i|y_{i\text{-}1}, x)$ aligns closely with the target distribution $\pi(y|x)$ (Gelman et al., 1997). This means that the sampling efficiency decreases as this discrepancy increases.

Therefore, we apply the importance sampling strategy (Kahn and Marshall, 1953; Owen and Zhou, 2000) to improve the proposal distribution. We define an importance distribution $q(y_i|y_{i\text{-}1}, x)$ that complies with length constraints, serving as a replacement for the proposal distribution to facilitate accelerated sampling. Equations (5) and (11) can be further derived when $y_i \sim q(y_i|y_{i\text{-}1}, x)$:

$$\begin{aligned}\mathcal{A}(y_{i\text{-}1} \to y_i) &= \frac{p(y_i|y_{i\text{-}1}, x)}{q(y_i|y_{i\text{-}1}, x)} \min\left(1, \frac{\pi(y_i|x)}{\pi(y_{i\text{-}1}|x)}\right) \\ &\leq \min\left(1, \frac{f(y_i)P(y_i|x)}{f(y_{i\text{-}1})P(y_{i\text{-}1}|x)}\right), \qquad (12)\end{aligned}$$

where $\frac{p(y_i|y_{i\text{-}1},x)}{q(y_i|y_{i\text{-}1},x)} \leq 1$ and eq. (11) becomes an upper bound of $\mathcal{A}(y_{i\text{-}1} \to y_i)$. By simply replacing line 3 in algorithm 1 with $y_i \sim q(y_i|y_{i\text{-}1}|x)$ and calculating the acceptance rate with the upper bound eq. (12), we can significantly accelerate the sampling process. Although calculating this upper bound may lead to higher acceptance rates, potentially compromising generation quality, the remarkable capabilities of LLMs fortunately mitigate this risk to an almost negligible level. In addition, the detailed template for the importance distribution is:

Prompt Template: $y_i \sim q(y_i|y_{i\text{-}1}, x)$

`[USER]`: Answer the following instruction using $\{\ell\}$ words or less: $\{x\}$
`[ASSISTANT]`: Answer: $\{y_{i\text{-}1}\}$

`[USER]`: The generated answer is too (long / short) at $\{$`Len`$(y)\}$ words.
Please (delete / add) $\{\mathscr{D}(y, \ell)\}$ words appropriately based on the previous response:
`[ASSISTANT]`: Answer: $\{y_i\}$

Models	Samplers	Acc↑	L1↓	L2↓	Rouge-1/2/3	BERTScore-P/R/F
Llama2	Inst	4.1%	11.42	15.20	0.37/0.13/0.24	0.89/0.88/0.89
	Cmp	4.5%	8.58	11.22	0.37/0.13/0.24	0.89/0.88/0.88
	Ours	81.6%	0.24	0.64	0.36/0.12/0.24	0.88/0.88/0.88
Qwen2.5	Inst	3.2%	8.64	10.83	0.33/0.10/0.21	0.88/0.88/0.88
	Cmp	4.1%	8.39	10.54	0.33/0.10/0.21	0.88/0.88/0.88
	Ours	86.4%	0.18	0.72	0.33/0.10/0.21	0.88/0.88/0.88
Llama3	Inst	9.1%	4.78	6.10	0.39/0.14/0.25	0.89/0.89/0.89
	Cmp	9.8%	4.53	5.94	0.39/0.14/0.25	0.89/0.89/0.89
	Ours	78.6%	0.29	0.66	0.38/0.14/0.25	0.89/0.89/0.89
Llama3.1	Inst	7.7%	3.88	5.10	0.38/0.13/0.24	0.89/0.88/0.89
	Cmp	10.8%	3.83	5.14	0.38/0.13/0.24	0.89/0.88/0.89
	Ours	100.0%	0.00	0.00	0.38/0.13/0.24	0.88/0.88/0.88
Gpt-3.5	Inst	5.1%	8.29	13.69	0.36/0.12/0.23	0.88/0.88/0.88
	Cmp	32.5%	1.75	4.35	0.36/0.12/0.23	0.88/0.88/0.88
	Ours	95.0%	0.14	1.11	0.36/0.12/0.23	0.88/0.88/0.88
Gpt-4	Inst	15.7%	2.10	2.67	0.36/0.12/0.23	0.88/0.88/0.88
	Cmp	35.3%	1.43	1.99	0.36/0.12/0.23	0.88/0.88/0.88
	Ours	99.2%	0.01	0.12	0.36/0.12/0.23	0.88/0.88/0.88

Table 1: Results of the length control on the CNN/DailyMail dataset. Inst is the baseline response with length-guided instructions. Cmp is the InstructCMP baseline. Ours represents our iterative sampling framework.

It should be noted that our method can perform parallel sampling as long as the corresponding Llm supports it, further improving the control efficiency.

4 Experiments

4.1 Experimental Setup

Datasets For exact length control, we utilize the CNN/DailyMail dataset (CnnDm, Nallapati et al. (2016)), where the length instruction ℓ is extracted from the references. For length-interval control, we use the Alpaca-Eval-LI (Alpaca) and MT-Bench-LI (MtBench) datasets (Yuan et al., 2024), which are derived from the Alpaca-Eval dataset (Dubois et al., 2024b) and the MT-Bench dataset (Zheng et al., 2023). The length interval instructions are already provided in the dataset, where $\ell_1 = 0$ and ℓ_2 is the length of the reference response. For CnnDm, we randomly choose 1000 samples from the 3.0 version of the test set, since the instruction following task contains 1042 samples (802 from Alpaca and 240 from MtBench).

Llms We evaluate the effectiveness of our framework in the latest Llms, including Llama-2-7B (Llama2, Touvron et al. (2023b)), Qwen-2.5-7B (Qwen2.5, Team (2024)), Llama-3/3.1-8B (Llama3/3.1, Dubey et al. (2024)), and Gpt-3.5/4 (Achiam et al., 2023). For white-box Llms like Llama, we use them as black-box models, where the maximum iteration trial is 5 with a beam size of 16. For black-box models based on APIs like Gpt-4, we set the maximum iteration trial as 15 without parallel sampling. We compare with two baselines: naive instructions and InstructCMP (Juseon-Do et al., 2024), which designs specialized instructions for length compression.

Models	Samp.	Acc↑	L1↓	L2↓	Win.↑
Alpaca-Eval-LI					
Llama3.1	Inst	91.6%	2.47	15.64	71.6%
	Cmp	95.3%	0.79	7.18	71.9%
	Ours	99.8%	0.06	1.69	76.7%
Gpt-4	Inst	37.2%	21.38	37.61	30.2%
	Cmp	82.4%	1.43	4.92	69.5%
	Ours	99.2%	0.02	0.17	92.0%
MT-Bench-LI					
Llama3.1	Inst	80.4%	10.15	60.71	35.2%
	Cmp	90.1%	2.12	10.37	38.8%
	Ours	98.8%	0.73	7.12	42.9%
Gpt-4	Inst	54.7%	13.99	29.16	27.4%
	Cmp	84.0%	1.69	6.13	41.8%
	Ours	98.8%	0.05	0.41	63.7%

Table 2: Results of the length control on the Alpaca-Eval-LI dataset and the MT-Bench-LI dataset.

Evaluation Metrics We use several metrics to estimate the effect of the length control. **Acc**uracy measures the ratio of generated contents that are fully in accordance with the length constraint. Given N generated contents, **L1** measures the average Manhattan distance $\frac{1}{N}\sum_y |\texttt{Len}(y) - \ell|$ and **L2** measures the average Euclidean distance $\sqrt{\frac{1}{N}\sum_y |\texttt{Len}(y) - \ell|^2}$. For quality evaluation of the summary task, we use the classic score **Rouge**

Models	Win↑	Tie	Lose↓	Kappa↑
Qwen2.5	69.5%	10.0%	20.5%	0.57
Llama3.1	59.0%	12.5%	30.5%	0.48

Table 3: Human Evaluation: Ours vs Inst.

Trials	Acc↑	L1↓	L2↓	Rouge-(1/2/L)
Llama3.1				
0	7.7%	3.88	5.10	0.38/0.13/0.24
1	86.4%	0.18	0.55	0.38/0.13/0.24
2	99.2%	0.04	0.28	0.38/0.13/0.24
3	99.8%	0.01	0.03	0.38/0.13/0.24
4	100.0%	0.00	0.00	0.38/0.13/0.24
5	100.0%	0.00	0.00	0.38/0.13/0.24

Table 4: Analysis of the iteration trial on the CnnDm dataset, where the beam size is 16.

(Lin, 2004), **BERTScore** (Zhang* et al., 2020) and human evaluation. For instruction following tasks, we use the length-instructed **Win**rate (Yuan et al., 2024), where responses are compared pairwise with baselines. The winner is determined by both the quality of the responses provided by LLM-as-a-Judge (Zheng et al., 2023), and the adherence to the length constraints. If the response exceeds the length constraint, it is automatically lost.

4.2 Main Results

The detailed comparisons between the baselines and our framework are demonstrated in Tables 1 and 2. Table 1 presents the results of length control experiments conducted on the CNN/DailyMail dataset. Our method demonstrates significant improvements over the baseline instruction methods across all models and length-related metrics. Specifically, we achieve near-perfect or perfect accuracy (Acc), with values exceeding 95% for the most advanced LLMs (Llama3.1, Gpt-3.5, and Gpt-4), while the baselines struggle with accuracy values below 36%. Furthermore, our approach exhibits substantially lower errors of L1 and L2, indicating precise adherence to the target lengths. For example, on Llama3.1, our framework achieves an accuracy of 100%, demonstrating flawless length control. Similarly, we attain a 99.2% accuracy on Gpt-4, reducing the L1 and L2 errors to 0.01 and 0.12, respectively. Beyond the significant improvement in length control, our method introduces almost no degradation in generation quality, where Rouge and BERTScore remain almost the same to baselines.

Beams	Acc↑	L1↓	L2↓	Rouge-(1/2/L)
Llama3.1				
0	7.7%	3.88	5.10	0.38/0.13/0.24
1	93.3%	0.14	0.88	0.37/0.13/0.24
2	98.9%	0.02	0.12	0.37/0.13/0.24
4	99.7%	0.01	0.05	0.38/0.14/0.25
8	100.0%	0.00	0.00	0.38/0.13/0.24
16	100.0%	0.00	0.00	0.38/0.13/0.24

Table 5: Analysis of the beam size on the CnnDm dataset, where the iteration trial is 5.

Samp.	Acc↑	L1↓	L2↓	Rouge-(1/2/L)
Inst	7.7%	3.88	5.10	0.38/0.13/0.24
Rand	38.8%	1.18	1.85	0.38/0.14/0.24
Mh	40.2%	1.47	3.20	0.36/0.13/0.23
Mh+Is	93.3%	0.14	0.88	0.37/0.13/0.24

Table 6: Ablation study of Llama3.1 on CnnDm, where the iteration trial is 5 and the beam size is 1.

Table 2 evaluates the performance of our method on Alpaca-Eval-LI and MT-Bench-LI datasets. Although these two datasets are relatively easier compared to the exact length control task, the performance improvement (with an accuracy increase of at least 4.5%) brought about by our method compared to the baselines is significant, confirming the consistent superiority of our framework across different benchmarks. In addition, the LLM judged pairwise Winrate of our approach improves. These results highlight the effectiveness of our iterative sampling framework in achieving robust and accurate length control across diverse LLMs.

Table 3 is the human evaluation on CnnDm with three human and three LLM annotators. Each evaluator is presented with the article and the paired summaries of two methods (Ours and Inst). They are asked to assign the win, tie, or lose for each pair based on the informativeness and conciseness of each summary. We randomly select 100 cases for each method and shuffle their demonstrated positions. It is observed that our method shows a significant advantage over the baseline. Beisdes, the human and LLM evaluations are highly aligned, as evidenced by strong kappa correlations.

4.3 Analyses

We analyze the hyperparameters of our framework, the number of iteration trials and the beam size, which are illustrated in Tables 4 and 5. Both hyperparameters are used to expand and explore the sampling space, with larger iteration trials demanding greater time overhead and larger beam sizes in-

Task	Samp.	Steps↓	Acc↑	L1↓	L2↓
CnnDm	Rand	18.6	95.3%	0.06	0.35
	Mh	17.1	96.0%	0.34	1.85
	Mh+Is	2.4	100.0%	0.00	0.00
Alpaca	Rand	0.6	98.0%	0.21	1.71
	Mh	0.9	95.3%	0.32	2.67
	Mh+Is	0.1	100.0%	0.00	0.00
MtBench	Rand	3.0	97.9%	0.33	3.26
	Mh	3.3	96.7%	0.42	4.88
	Mh+Is	0.8	99.5%	0.06	0.85

Table 7: Convergence steps on Llama3.1.

Task	Samp.	Steps↓	Acc↑	L1↓	L2↓
CnnDm	Rand	2.6	93.8%	0.06	0.29
	Mh	2.5	91.4%	0.09	0.58
	Mh+Is	1.0	98.0%	0.02	0.14
Alpaca	Rand	2.5	77.3%	2.52	6.54
	Mh	3.0	93.6%	3.02	8.67
	Mh+Is	0.4	98.2%	0.09	0.81
MtBench	Rand	1.3	83.7%	2.57	7.89
	Mh	1.8	78.3%	4.09	14.71
	Mh+Is	0.1	99.8%	0.01	0.09

Table 8: Convergence steps on Gpt-4.

curring higher space costs. As the sampling space reduces, the influence of length control progressively decreases. In particular, this reduction is non-linear, with the rate of decline accelerating significantly. Besides, comparing the two tables, we can observe that the number of iteration trials contributes more to the control effect than the beam size. With a smaller sampling space of 2 beams × 5 trials, the accuracy (98.9%) outperforms the situation with 16 beams × 1 trial (86.4%).

4.4 Ablation Study

Table 6 presents an ablation study evaluating the performance of different sampling strategies for Llama3.1 on the CNN/DailyMail dataset. We examine four sampling strategies: (1) Inst is the instruction following baseline without iterations; (2) Rand extends the baseline to resample at each iteration and retains the best one; (3) Mh is our initial version of the Metropolis-Hastings framework that resamples with the proposal distribution $y \sim p(y_i|y_{i-1}, x)$ during each iteration; and (4) Mh+Is is our complete method which replaces the proposal distribution with the importance distribution $q(y_i|y_{i-1}, x)$. We set the beam size to 1, because sampling a batch of initial states $y_0 \sim P(y|x)$ is actually the Rand method and we want to eliminate this influence. Experimental results show that with the powerful instruction following capabilities of Llms, random sampling of more candidates can achieve good control effects. However, the original Metropolis-Hastings method does not make the process more efficient and is sometimes even worse than random sampling. By replacing the proposal distribution with an importance sampling strategy, our method achieves significant improvements.

4.5 Convergence Study

Furthermore, we analyze the accurate convergence speed of different sampling methods in Tables 7 and 8. We set the beam size to 1 as in section 4.4 and the maximum iteration step for each case is 100 for Llama3.1 and 15 for Gpt-4. We report the average iteration **Steps** required to satisfy the length constraints, which excludes the first sampling step $y_0 \sim P(y|x)$. We observe that different models have different convergence steps for different tasks. In general, precise length control tasks are more difficult and require more iterations. Even so, we achieve an almost perfect control effect with only 2.4 iteration steps on average for Llama3.1. We even only need an average of 0.1 iterations for Llama3.1 to perform perfect control on the Alpaca-Eval-LI dataset. For Gpt-4, we only need 1.0 iterations at most on average to obtain good control results. Therefore, our framework can achieve extremely effective length control performance with acceptable time overhead.

5 Conclusion

We propose a novel length controllable sampling framework for black-box models and verify the effectiveness with experiments and analyses. Our study confirms that an almost perfect length control can be achieved on Llms, which is of great significance to improve their instruction following ability. In addition, although our framework performs well, its sampling efficiency and generation effect are affected by the capabilities of Llm itself. Fortunately, with the rapid development of Llms, this concern will gradually disappear. Its worth noting that we do not directly compare with the length training methods, because (1) the black-box models are not trainable, and (2) the training methods are based on specific datasets and possess some data bias, which is contrary to the objective of a more generalized length control. We hope to explore more efficient and general length control schemes in our future studies.

Limitations

Despite the promising results demonstrated in our experiments, our method has some limitations that merit further discussion:

- **Inference Overhead:**
  Our approach introduces additional inference overhead due to the iterative nature of the method. Although the experimental results show that satisfactory results can often be achieved in 2 iterations for advanced models such as LLAMA3.1. However, more iteration steps are required for more difficult scenarios or weaker LLMs. This additional computational cost may present challenges for large-scale batch generation tasks where inference speed is critical. Future research could explore optimization techniques to reduce the number of iterations required or design lightweight variants to better suit the high-throughput applications.

- **Dependency on Instruction Following Abilities:**
  The performance of our method is highly dependent on the instruction following capabilities of the underlying model. For state-of-the-art LLMs such as LLAMA3.1 and GPT-4, fewer iterations are typically sufficient to achieve satisfactory results. However, when applied to models with less robust instruction-following abilities, the number of iterations required may increase significantly, potentially affecting efficiency. Addressing this limitation could involve developing methods to enhance instruction alignment for less capable models or incorporating external mechanisms to mitigate the dependency on instruction following abilities.

Considering our experiments, the limitations are:

- **Baselines:** We do not directly compare with training methods for length control because: (1) our framework is dedicated to black-box LLMs, which is not trainable; (2) length instructions have already been incorporated in the supervised fine-tuning stage of LLMs, which means LLMs themselves are length trainable baselines; (3) the training methods are based on specific datasets and possess some data bias, which is contrary to the objective of a more generalized length control; and (4) large-scale training of length instructions on LLMs such as LLAMA3.1 requires a lot of computing resources that we cannot currently afford.

- **Models:** Currently, we only test the most widely used LLMs. Due to the limitations of computing resources and costs, we are unable to test white-box models with larger parameters (such as 70B), nor can we afford the test of other API-based black-box LLMs on a large scale.

Ethics Statement

This research focuses on controlling the output length of LLMs to address practical usability and fairness concerns in various applications, such as summarization, dialogue systems, and content generation. By enabling precise length control, this work aims to enhance user experience, ensure relevance, and reduce unintended biases introduced by excessively verbose or overly concise outputs.

We recognize the potential ethical risks associated with the misuse of controlled generation, such as the creation of misleading or harmful content tailored to specific lengths. To mitigate such risks, our methodology emphasizes transparency, reproducibility, and alignment with ethical guidelines in AI development. Additionally, we advocate for integrating robust content moderation mechanisms in downstream applications to safeguard against unintended consequences.

This research was conducted following established ethical standards, ensuring that the datasets used respect privacy and are free of harmful biases to the best of our ability. Future work will further explore the societal implications of this technology, ensuring its responsible and equitable deployment.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

Siddhartha Chib and Edward Greenberg. 1995. Understanding the metropolis-hastings algorithm. *The american statistician*, 49(4):327–335.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024a. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Yann Dubois, Chen Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy S Liang, and Tatsunori B Hashimoto. 2024b. Alpacafarm: A simulation framework for methods that learn from human feedback. *Advances in Neural Information Processing Systems*, 36.

Angela Fan, David Grangier, and Michael Auli. 2017. Controllable abstractive summarization. *arXiv preprint arXiv:1711.05217*.

Andrew Gelman, Walter R Gilks, and Gareth O Roberts. 1997. Weak convergence and optimal scaling of random walk metropolis algorithms. *The annals of applied probability*, 7(1):110–120.

Heikki Haario, Eero Saksman, and Johanna Tamminen. 2001. An adaptive metropolis algorithm. *Bernoulli*, 7(6):223–242.

W. K. Hastings. 1970. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Chenyang Huang, Hao Zhou, Cameron Jen, Kangjie Zheng, Osmar Zaiane, and Lili Mou. 2024. A decoding algorithm for length-control summarization based on directed acyclic transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11572–11583, Miami, Florida, USA. Association for Computational Linguistics.

Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2024. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*.

Renlong Jie, Xiaojun Meng, Lifeng Shang, Xin Jiang, and Qun Liu. 2024. Prompt-based length controlled generation with multiple control types. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1067–1085, Bangkok, Thailand. Association for Computational Linguistics.

Juseon-Do, Jingun Kwon, Hidetaka Kamigaito, and Manabu Okumura. 2024. InstructCMP: Length control in sentence compression through instruction-based large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8980–8996, Bangkok, Thailand. Association for Computational Linguistics.

Herman Kahn and Andy W Marshall. 1953. Methods of reducing sample size in monte carlo computations. *Journal of the Operations Research Society of America*, 1(5):263–278.

Jiaming Li, Lei Zhang, Yunshui Li, Ziqiang Liu, Yuelin Bai, Run Luo, Longze Chen, and Min Yang. 2024. Ruler: A model-agnostic method to control generated length for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3042–3059, Miami, Florida, USA. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang, Wenbin Hu, Hanning Zhang, Hanze Dong, Renjie Pi, Han Zhao, Nan Jiang, Heng Ji, Yuan Yao, and Tong Zhang. 2024. Mitigating the alignment tax of RLHF. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 580–606, Miami, Florida, USA. Association for Computational Linguistics.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2023. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*.

Yizhu Liu, Qi Jia, and Kenny Zhu. 2022. Length control in abstractive summarization by pretraining information selection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6885–6895.

Yizhu Liu, Zhiyi Luo, and Kenny Zhu. 2018. Controlling length in abstractive summarization using a convolutional neural network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4110–4119.

Takuya Makino, Tomoya Iwakura, Hiroya Takamura, and Manabu Okumura. 2019. Global optimization under length constraint for neural text summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1039–1048.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. 1953. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Art Owen and Yi Zhou. 2000. Safe and effective importance sampling. *Journal of the American Statistical Association*, 95(449):135–143.

Aske Plaat, Annie Wong, Suzan Verberne, Joost Broekens, Niki van Stein, and Thomas Back. 2024. Reasoning with large language models, a survey. *arXiv preprint arXiv:2407.11511*.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. 2024a. From r to q^*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024b. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Rico Sennrich. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Prasann Singhal, Tanya Goyal, Jiacheng Xu, and Greg Durrett. 2023. A long way to go: Investigating length correlations in rlhf. *arXiv preprint arXiv:2310.03716*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

I Sutskever. 2014. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

Sho Takase and Naoaki Okazaki. 2019. Positional encoding to control output sequence length. *arXiv preprint arXiv:1904.07418*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: open and efficient foundation language models. arxiv. *arXiv preprint arXiv:2302.13971*.

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

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Noah Wang, Feiyu Duan, Yibo Zhang, Wangchunshu Zhou, Ke Xu, Wenhao Huang, and Jie Fu. 2024. PositionID: LLMs can control lengths, copy and paste with explicit positional awareness. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16877–16915, Miami, Florida, USA. Association for Computational Linguistics.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Zihao Yi, Jiarui Ouyang, Yuwen Liu, Tianhao Liao, Zhe Xu, and Ying Shen. 2024. A survey on recent advances in llm-based multi-turn dialogue systems. *arXiv preprint arXiv:2402.18013*.

Zhongyi Yu, Zhenghao Wu, Hao Zheng, Zhe XuanYuan, Jefferson Fong, and Weifeng Su. 2021. Lenatten: An effective length controlling unit for text summarization. *arXiv preprint arXiv:2106.00316*.

Weizhe Yuan, Ilia Kulikov, Ping Yu, Kyunghyun Cho, Sainbayar Sukhbaatar, Jason Weston, and Jing Xu. 2024. Following length constraints in instructions. *arXiv preprint arXiv:2406.17744*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping

Yu, Lili Yu, et al. 2024. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.

A Prompt Templates

A.1 Initial States

For the abstractive summarization task with exact length constraints, we randomly choose an example (x_c, y_c, ℓ_c) from the training set as an one-shot demonstration for LLMs, because the chat LLMs are not specifically trained for the output mode of summary tasks. The detailed template is:

Prompt Template: $y_0 \sim P(y|x)$

`[SYSTEM]`: You are a powerful abstractive summarizer.

`[USER]`: Document: $\{x_c\}$
Based on the previous document, provide a high-quality summary in exactly $\{\ell_c\}$ words:

`[ASSISTANT]`: Summary: $\{y_c\}$

`[USER]`: Document: $\{x\}$
Based on the previous document, provide a high-quality summary in exactly $\{\ell\}$ words:

`[ASSISTANT]`: Summary: $\{y_0\}$

For instruction following tasks with length intervals, we directly use zero-shot with the template:

Prompt Template: $y_0 \sim P(y|x)$

`[USER]`: Answer the following instruction using $\{\ell\}$ words or less: $\{x\}$

`[ASSISTANT]`: Answer: $\{y_0\}$

A.2 Probability Densities of Current States

We demonstrate the detailed $\{Criteria\}$ of evaluation for different tasks. For abstractive summarization: we score the generated summaries in 5 dimensions on a scale of 1-10.

1. **Information Coverage:** Does the summary include the most important and critical information from the document?

2. **Linguistic Fluency:** Are the sentences in the summary fluent, natural, and grammatically correct?

3. **Conciseness:** Does the summary avoid redundancy while retaining key information?

4. **Logical Coherence:** Is the summary well-structured with clear and logical flow?

5. **Faithfulness:** Does the summary accurately reflect the facts in the original document without adding false or misleading information?

The evaluation $\{Criteria\}$ for the general instruction following task is of 6 dimensions:

1. **Helpfulness:** Does the response directly address the instruction and provide meaningful assistance?
2. **Relevance:** Does the response stay on topic and avoid unnecessary or unrelated information?
3. **Accuracy:** Is the information in the response factually correct and free of errors?
4. **Depth:** Does the response demonstrate a deep understanding of the topic, including nuanced explanations where relevant?
5. **Creativity:** Does the response display originality, creativity, or a unique approach to addressing the instruction?
6. **Level of Detail:** Is the response sufficiently detailed, providing comprehensive and thorough explanations where necessary?

Following the setting of MT-Bench, we set a special evaluation $\{Criteria\}$ for math-related instruction following tasks such as reasoning, math and coding, which is described below.

1. **Correctness:** Is the answer logically sound, factually accurate, and free from errors?
2. **Helpfulness:** Does the response directly address the instruction and provide meaningful assistance?
3. **Clarity:** Is the response well-structured and easy to understand?
4. **Efficiency:** Does the response provide an optimal solution without unnecessary complexity?
5. **Completeness:** Does the response fully cover the instruction's requirements and edge cases?
6. **Robustness:** Can the response handle ambiguity or complexity in the instruction?

We formalize the output to facilitate the extraction of key information, where the $\{Format\}$ is

Response 1:
1. Information Coverage:: [Score]/10
2. Linguistic Fluency: [Score]/10
. . . . . .
Overall Score: [Total Score]/50
Response 2:
1. Information Coverage:: [Score]/10
2. Linguistic Fluency: [Score]/10
. . . . . .
Overall Score: [Total Score]/50
Conclusion:
- **Better Response:** [Response 1/Response 2].
- **Score Ratio (Response 1 ÷ Response 2):** [Ratio, rounded to two decimal places].

We calculate the eq. (10) via

$$\frac{P(y_i|x)}{P(y_{i\text{-}1}|x)} \simeq \frac{\phi(y_i|x)}{\phi(y_{i\text{-}1}|x)} = \frac{\text{Score of Response 1}}{\text{Score of Response 2}}. \tag{13}$$

Therefore, the prompt templates for estimating the target probability density are:

Prompt Template: $\Phi(y_i, y_{i-1}|x)$

`[SYSTEM]`: You are a powerful evaluator for abstractive summarization.

`[USER]`: I need to compare and evaluate the quality of two summaries generated for a given document. Please provide a quantitative assessment of their performance based on the criteria below.
Document: $\{x\}$
Summary 1: $\{y_i\}$
Summary 2: $\{y_{i-1}\}$
Evaluation Criteria (each scored on a scale of 1-10, with 10 being the best): $\{Criteria\}$
Instructions:
* Score each summary based on the above criteria.
* Calculate an overall score for each summary as the sum of all criteria scores (maximum 50).
* Conclude by identifying which summary is better overall.
* Calculate a score ratio of Summary 1 to Summary 2 (Summary 1 Score ÷ Summary 2 Score).
Output Format: $\{Format\}$

where we force LLMs to score the responses of the adjacent steps generated by itself. By extracting the score ratio from $\{Format\}$, we can estimate the fraction of the target distribution.

For instruction following tasks, we use the pairwise template derived from the Alpaca-Eval, which emphasizes that the length of the generated content and the position of the presentation should not be a bias in scoring.

Prompt Template: $\Phi(y_i, y_{i-1}|x)$

`[SYSTEM]`: You are a highly efficient assistant, who evaluates and selects the best large language model (LLMs) based on the quality of their responses to a given instruction.
This process will be used to create a leaderboard reflecting the most accurate and human-preferred answers.

`[USER]`: I require a leaderboard for various large language models. I'll provide you with an instruction given to these models and their corresponding responses. Your task is to assess these responses, provide a quantitative assessment of their performance based on the criteria below, and select the model that produces the best output from a human perspective. Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision.
Instruction: $\{x\}$
Response 1: $\{y_i\}$
Response 2: $\{y_{i-1}\}$
Tasks:
* Score each response based on the above criteria.
* Calculate an overall score for each response as the sum of all criteria scores (maximum 60).
* Conclude by identifying which response is better overall.
* Calculate a score ratio of Response 1 to Response 2 (Response 1 Score ÷ Response 2 Score).
Output Format: $\{Format\}$

A.3 Propose New States

Proposal Distribution For the abstractive summarization task, the prompt template for sampling from the proposal distribution $p(y_i|y_{i-1}, x)$ is:

Prompt Template: $y_i \sim p(y_i|y_{i-1}, x)$

`[SYSTEM]`: You are a powerful abstractive summarizer.

`[USER]`: Document: $\{x\}$
Based on the previous document, provide a high-quality summary in exactly $\{\ell\}$ words:

`[ASSISTANT]`: Summary: $\{y_{i-1}\}$

`[USER]`: Please generate a new summary based on the previous one:

`[ASSISTANT]`: Summary: $\{y_i\}$

The template for instruction following task is:

Prompt Template: $y_i \sim p(y_i|y_{i-1}, x)$

`[USER]`: Answer the following instruction using $\{\ell\}$ words or less. `\n\n` $\{x\}$

`[ASSISTANT]`: Answer:`\n`$\{y_{i-1}\}$

`[USER]`: Please generate a new answer based on the previous one:

`[ASSISTANT]`: Answer: `\n` $\{y_i\}$

Importance Distribution We split the importance distribution into two segments. When the candidate length is far from the target length $\mathscr{D}(y, \ell) > 3$, we use a looser objective so that LLMs can have more opportunities for semantic organization, which is beneficial for the quality of generation. The template for abstractive summarization is:

Prompt Template: $y_i \sim q(y_i|y_{i-1}, x)$

`[SYSTEM]`: You are a powerful abstractive summarizer.

`[USER]`: Document: `\n` $\{x\}$ `\n\n` Based on the previous document, provide a high-quality summary in exactly $\{\ell\}$ words:

`[ASSISTANT]`: Summary: `\n` $\{y_{i-1}\}$

`[USER]`: The generated summary is too (long / short) at $\{$`Len`$(y)\}$ words.
Please improve it to be exactly $\{\ell\}$ words by (focusing on the core ideas and removing some redundant details / adding some details and maintaining clarity and relevance):

`[ASSISTANT]`: Summary: `\n` $\{y_i\}$

The prompt template for instruction following is:

Prompt Template: $y_i \sim q(y_i|y_{i\text{-}1}, x)$

`[USER]:` Answer the following instruction using $\{\ell\}$ words or less. `\n\n` $\{x\}$

`[ASSISTANT]:` Answer: `\n` $\{y_{i-1}\}$

`[USER]:` The generated answer is too long at $\{\texttt{Len}(y)\}$ words. Please improve it to be exactly $\{\ell\}$ words or less by focusing on the core contents and removing any unhelpful, irrelevant, or inaccurate parts:

`[ASSISTANT]:` Answer: `\n` $\{y_i\}$

When the candidate length is close to the target length $\mathscr{D}(y, \ell) \leq 3$, we force an accurate length control such that LLMs are required to add or delete an exact number of words. The prompt template for abstractive summarization is:

Prompt Template: $y_i \sim q(y_i|y_{i\text{-}1}, x)$

`[SYSTEM]:` You are a powerful abstractive summarizer.

`[USER]:` Document: `\n` $\{x\}$ `\n\n` Based on the previous document, provide a high-quality summary in exactly $\{\ell\}$ words:

`[ASSISTANT]:` Summary: `\n` $\{y_{i-1}\}$

`[USER]:` Please (delete / add) $\{\mathscr{D}(y, \ell)\}$ words appropriately based on the previous summary:

`[ASSISTANT]:` Summary: `\n` $\{y_i\}$

The prompt template for instruction following is:

Prompt Template: $y_i \sim q(y_i|y_{i\text{-}1}, x)$

`[USER]:` Answer the following instruction using $\{\ell\}$ words or less. `\n\n` $\{x\}$

`[ASSISTANT]:` Answer: `\n` $\{y_{i-1}\}$

`[USER]:` The generated answer is too long at $\{\texttt{Len}(y)\}$ words. Please delete $\{\mathscr{D}(y, \ell)\}$ words appropriately based on the previous response:

`[ASSISTANT]:` Answer: `\n` $\{y_i\}$

Models	Top-K	Top-P	Temp.	Rep.
Qwen2.5	20	0.8	0.7	1.05
Llama2	50	0.9	0.6	1.00
Llama3	50	0.9	0.6	1.00
Llama3.1	50	0.9	0.6	1.00

Table 9: Generation configurations of LLMs.

Models	Win↑	Tie	Lose↓	Kappa↑
Qwen2.5	71.0%	12.0%	17.0%	0.61
Llama3.1	58.0%	14.0%	28.0%	0.53

Table 10: Human Evaluation: Ours vs Inst. (three human evaluators)

B Experimental Details

Our experiments are implemented on the *Huggingface Transformers* package[1]. All LLMs we used are the chat version trained with supervised fine tuning, where Llama2 and Qwen2.5 have 7B parameters while Llama3 and Llama3.1 have 8B parameters. The generation configurations of each model are set by default, as demonstrated in Table 9. There is no training stage of our framework, and the inference is performed on an NVIDIA A100 80GB GPU with a random seed of 0. For the CNN/Daily Mail dataset, we randomly choose 1000 samples from the 3.0 version of the test set, since the instruction following task contains 1042 samples (802 from Alpaca-Eval-LI and 240 from MT-Bench-LI).

We conduct a human evaluation involving three Ph.D. students with expertise in natural language processing and IELTS scores of at least 6.5. Each evaluator is presented with the article and the paired summaries of two methods (Ours and Inst). They are tasked with assigning a score for each pair as a win, tie, or lose based on the informativeness and conciseness of each summary. We randomly shuffle the demonstrated position of summaries from different samplers and the evaluators are blinded to the identity of the samplers to prevent bias. We randomly select 100 cases for each model on CnnDm. To compensate for the inability to hire native English-speaking evaluators, we further leverage state-of-the-art English-oriented LLMs as additional evaluators, which are GPT-4o-latest, Meta-Llama-3.1-405B-Instruct, and claude-3.5-sonnet-20241022. These models have been extensively validated in numerous studies, demonstrating per-

---

[1] https://github.com/huggingface/transformers

Models	Win↑	Tie	Lose↓	Kappa↑
Qwen2.5	68.0%	8.0%	24.0%	0.68
Llama3.1	56.0%	11.0%	33.0%	0.59

Table 11: Human Evaluation: Ours vs Inst. (three LLM evaluators)

Models	Win↑	Tie	Lose↓	Kappa↑
Qwen2.5	69.5%	10.0%	20.5%	0.57
Llama3.1	59.0%	12.5%	30.5%	0.48

Table 12: Human Evaluation: Ours vs Inst. (three human and three LLM evaluators)

formance that exceeds human evaluators in various NLP tasks. By incorporating both human and AI-based assessments, we ensure a more comprehensive and reliable evaluation of summary quality. The results and kappa correlations among three human evaluators, three LLM evaluators, and three human and three LLM evaluators are demonstrated in Tables 10 to 12. We can observe that the human evaluations are highly aligned with the LLM annotations, with strong kappa correlations. Our method shows a significant advantage over the instruction baseline considering the informativeness and conciseness of the summaries. This improvement is not in line with the Rouge score. Because the Rouge score calculates the correlation between the generated summaries and the reference summaries, and the zero-shot / one-shot LLMs share a totally different distribution compared to the references. Therefore, our method can improve both the accuracy of length control and the quality of generated contents. In addition, the slight decrease in the Rouge score of our method can be explained by the fact that Rouge is a recall score and it is easy for longer contents to have a higher score. The prompt used for LLM evaluators are as follows.

C Analyses

Full results of the length control on the Alpaca-Eval-LI dataset and the MT-Bench-LI dataset are demonstrated in Table 15. We demonstrate the hyperparameter analyses in Tables 16 and 17. Similar to the observation in §4.3, the marginal effect of Llama3 as the sampling space grows is between Qwen2.5 and Llama3.1. For ablation studies in Qwen2.5 (Table 14) and Llama3 (Table 13), our sampling framework outperforms other methods. However, since the instruction following capabilities of these models are not as powerful as Llama3.1, their improvement may not be as significant.

Prompt Template: LLM Evaluator

`[SYSTEM]`: Please act as an impartial judge and evaluate the quality of the summaries provided by two AI assistants to the document displayed below.
You should choose the assistant that summarizes the document more informative and concise.
Your evaluation should also consider factors such as the information coverage, linguistic fluency, conciseness, logical coherence, and faithfulness.
Begin your evaluation by comparing the two summaries and provide a short explanation.
Avoid any position biases and ensure that the order in which the summaries were presented does not influence your decision.
Do not allow the length of the summaries to influence your evaluation.
Do not favor certain names of the assistants.
Be as objective as possible.
After providing your brief explanation, output your final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]" if assistant B is better, and "[[C]]" for a tie.
Document: article
Assistant A's Answer: summary1
Assistant B's Answer: summary2

D Generated Cases

The generated cases of Llama3.1 on the Alpaca-Eval-LI dataset are demonstrated in Table 18.

Samp.	Acc↑	L1↓	L2↓	Rouge-(1/2/L)
Inst	9.1%	4.78	6.10	0.39/0.14/0.25
Rand	30.7%	3.05	5.49	0.38/0.14/0.25
Mh	30.1%	4.07	5.98	0.38/0.13/0.24
Mh+Is	36.4%	1.63	2.51	0.39/0.14/0.25

Table 13: Ablation study of Llama3 on CnnDm, where the iteration trial is 5 and the beam size is 1.

Samp.	Acc↑	L1↓	L2↓	Rouge-(1/2/L)
Inst	3.2%	8.64	10.83	0.33/0.10/0.21
Rand	17.7%	3.54	5.23	0.33/0.10/0.21
Mh	14.1%	4.73	7.65	0.32/0.10/0.21
Mh+Is	24.6%	2.41	4.02	0.33/0.10/0.21

Table 14: Ablation study of Qwen2.5 on CnnDm, where the iteration trial is 5 and the beam size is 1.

Models	Samp.	Acc↑	L1↓	L2↓	Win.↑
Alpaca-Eval-LI					
Llama3	Inst	92.2%	1.48	9.12	76.5%
	Cmp	95.6%	0.37	3.01	73.3%
	Ours	99.8%	0.02	0.05	83.5%
Llama3.1	Inst	91.6%	2.47	15.64	71.6%
	Cmp	95.3%	0.79	7.18	71.9%
	Ours	99.8%	0.06	1.69	76.7%
Gpt-3.5	Inst	91.5%	1.16	4.92	57.0%
	Cmp	94.0%	0.91	4.63	59.8%
	Ours	100.0%	0.00	0.00	65.3%
Gpt-4	Inst	37.2%	21.38	37.61	30.2%
	Cmp	82.4%	1.43	4.92	69.5%
	Ours	99.2%	0.02	0.17	92.0%
MT-Bench-LI					
Llama3	Inst	78.8%	2.80	7.73	41.1%
	Cmp	83.3%	1.15	4.32	41.3%
	Ours	100.0%	0.00	0.00	42.1%
Llama3.1	Inst	80.4%	10.15	60.71	35.2%
	Cmp	90.1%	2.12	10.37	38.8%
	Ours	98.8%	0.73	7.12	42.9%
Gpt-3.5	Inst	87.9%	2.51	9.46	24.6%
	Cmp	93.5%	0.64	3.20	26.2%
	Ours	98.6%	0.09	0.73	27.3%
Gpt-4	Inst	54.7%	13.99	29.16	27.4%
	Cmp	84.0%	1.69	6.13	41.8%
	Ours	98.8%	0.05	0.41	63.7%

Table 15: Full results of the length control on the Alpaca-Eval-LI dataset and the MT-Bench-LI dataset.

Trials	Acc↑	L1↓	L2↓	Rouge-(1/2/L)
Qwen2.5				
0	3.2%	8.64	10.83	0.33/0.10/0.21
1	25.4%	3.58	5.80	0.33/0.10/0.21
2	52.8%	0.96	2.07	0.33/0.10/0.21
3	70.6%	0.51	1.55	0.33/0.10/0.21
4	79.1%	0.29	1.16	0.33/0.10/0.21
5	86.4%	0.18	0.72	0.33/0.10/0.21
Llama3				
0	9.1%	4.78	6.10	0.39/0.14/0.25
1	48.3%	0.98	1.80	0.38/0.14/0.24
2	64.8%	0.58	1.05	0.38/0.13/0.24
3	68.4%	0.44	0.85	0.38/0.13/0.24
4	72.3%	0.36	0.75	0.38/0.14/0.25
5	78.6%	0.29	0.66	0.38/0.14/0.25
Llama3.1				
0	7.7%	3.88	5.10	0.38/0.13/0.24
1	86.4%	0.18	0.55	0.38/0.13/0.24
2	99.2%	0.04	0.28	0.38/0.13/0.24
3	99.8%	0.01	0.03	0.38/0.13/0.24
4	100.0%	0.00	0.00	0.38/0.13/0.24
5	100.0%	0.00	0.00	0.38/0.13/0.24

Table 16: Analysis of the iteration trial on the CnnDm dataset, where the beam size is 16.

Beams	Acc↑	L1↓	L2↓	Rouge-(1/2/L)
Qwen2.5				
0	3.2%	8.64	10.83	0.33/0.10/0.21
1	24.6%	2.41	4.02	0.33/0.10/0.21
2	38.7%	1.75	3.69	0.33/0.10/0.21
4	57.1%	0.82	1.93	0.32/0.10/0.21
8	72.6%	0.46	1.49	0.32/0.10/0.20
16	86.4%	0.18	0.72	0.33/0.10/0.21
Llama3				
0	9.1%	4.78	6.10	0.39/0.14/0.25
1	36.4%	1.63	2.51	0.39/0.14/0.25
2	47.2%	1.08	2.10	0.38/0.14/0.25
4	55.8%	0.72	1.25	0.38/0.14/0.25
8	67.2%	0.47	0.93	0.38/0.14/0.25
16	78.6%	0.29	0.66	0.38/0.14/0.25
Llama3.1				
0	7.7%	3.88	5.10	0.38/0.13/0.24
1	93.3%	0.14	0.88	0.37/0.13/0.24
2	98.9%	0.02	0.12	0.37/0.13/0.24
4	99.7%	0.01	0.05	0.38/0.14/0.25
8	100.0%	0.00	0.00	0.38/0.13/0.24
16	100.0%	0.00	0.00	0.38/0.13/0.24

Table 17: Analysis of the beam size on the CnnDm dataset, where the iteration trial is 5.

LLAMA3.1 INST	LLAMA3.1 OURS
Answer the following instruction using 46 words or less. Is the US border open to Canada?	
Number of Words: 48	*Number of Words:* 44
The US border with Canada is open, but with restrictions. Land and sea borders are open for essential travel, while air travel requires a valid visa or electronic travel authorization. Some restrictions may apply, so it's best to check the official government websites for the latest information.	The US-Canada border is open, but with restrictions. Travelers must provide proof of COVID-19 vaccination, a negative test, or recovery from the virus. Additionally, some land ports of entry may be closed or have limited hours, so it's best to check before traveling.
Answer the following instruction using 50 words or less. Write me a top 10 list of the funniest ways to die.	
Number of Words: 110	*Number of Words:* 48
Please note that this is a dark humor list and not meant to be taken seriously. Here's a list of the funniest ways to die (in a fictional context): 1. Dying from laughing too hard at a dad joke. 2. Choking on a chicken nugget. 3. Falling into a giant pile of pillows. 4. Getting hit by a runaway inflatable unicorn. 5. Accidentally hugging a cactus. 6. Dying from a massive sneeze. 7. Getting stuck in a giant toilet. 8. Falling off a unicycle while riding a unicycle. 9. Dying from eating too much wasabi. 10. Being attacked by a swarm of bees while trying to get a selfie.	Here's a list of humorous ways to die: 1. Laughing too hard. 2. Choking on a whoopee. 3. Drowning in Jell-O. 4. Falling into feathers. 5. Being crushed by cats. 6. Overdosing on laughter. 7. Sneezing from feather. 8. Excessive flatulence. 9. Cheese coma. 10. Snoring loudly.
Answer the following instruction using 128 words or less. Who made Berlin?	
Number of Words: 151	*Number of Words:* 125
Berlin, the capital city of Germany, has a complex and multifaceted history. The city's origins date back to the 13th century when it was a small trading post on the Spree River. Over time, Berlin grew and became a significant center for trade, culture, and politics. In 1237, the Margraviate of Brandenburg was founded, and Berlin became its capital. In 1415, the Hohenzollern dynasty took control of the city, and Berlin continued to grow and prosper. Throughout the centuries, Berlin was influenced by various cultures and empires, including the Prussian Empire, the German Empire, and the Weimar Republic. The city's modern history is marked by significant events, including World War I and II, the division of Germany during the Cold War, and the fall of the Berlin Wall in 1989. Today, Berlin is a vibrant and diverse city, known for its rich history, cultural attractions, and thriving arts scene.	Berlin, the capital of Germany, has a complex and evolving history. The area has been inhabited since the Stone Age, with various tribes and cultures contributing to its development. The city of Berlin as we know it today began to take shape in the 13th century, when it was a small trading post on the Spree River. In 1237, the Margraviate of Brandenburg was established, and Berlin became the capital. Over the centuries, the city grew and was influenced by various rulers, including the Hohenzollern dynasty, which played a significant role in shaping modern Berlin. In 1871, Berlin became the capital of Germany, and it continued to grow and evolve throughout the 20th century, experiencing significant changes under the Weimar regime, and the post-war period.

Table 18: Generated cases of LLAMA3.1 on the Alpaca-Eval-LI dataset.