Improving Self Consistency in LLMs through Probabilistic Tokenization

Ashutosh Sathe [*] [1] [2] **Divyanshu Aggarwal** [*] [2] **Sunayana Sitaram** [2]

Abstract

Prior research has demonstrated noticeable performance gains through the use of probabilistic tokenizations, an approach that involves employing multiple tokenizations of the same input string during the training phase of a language model. Despite these promising findings, modern large language models (LLMs) have yet to be trained using probabilistic tokenizations. Interestingly, while the tokenizers of these contemporary LLMs have the capability to generate multiple tokenizations, this property remains underutilized.

In this work, we propose a novel method to leverage the multiple tokenization capabilities of modern LLM tokenizers, aiming to enhance the self-consistency of LLMs in reasoning tasks. Our experiments indicate that when utilizing probabilistic tokenizations, LLMs generate logically diverse reasoning paths, moving beyond mere surface-level linguistic diversity. We carefully study probabilistic tokenization and offer insights to explain the self consistency improvements it brings through extensive experimentation on 5 LLM families and 4 reasoning benchmarks.

1. Introduction

Being able to view a problem from multiple standpoints has been shown to be correlated to higher problem solving abilities and creativity (Gorenflo and Crano, 1998; Wang et al., 2006; Runco and Acar, 2012). In contrast, modern large language models (LLMs) such as MISTRAL-7B (Jiang et al., 2023), OLMO-7B (Groeneveld et al., 2024), MAMBA-2.8B (Gu and Dao, 2023) etc. often view language as a *unique* sequence of tokens. Byte-Pair Encoding (Gage, 1994; Sennrich et al., 2016) is a popular tokenization method employed by many LLMs to convert a given string into a sequence of tokens. Tokens in Byte-Pair Encoding (BPE) are often "merges" of smaller tokens which are also present in the vocabulary. E.g. token "_token" could be a merge of tokens "_to" and "ken". This means that there are multiple valid tokenizations of a given string depending on which (sub)tokens the tokenizer encoding function chooses to merge. Table 1 illustrates the phenomenon in action.

Prior work (Kudo, 2018; Provilkov et al., 2020) on "subword regularization" has shown that including such multiple tokenizations when training neural machine translators helps the model learn better token embeddings and become robust to noisy inputs. While modern LLMs are often trained *without* subword regularization, their tokenizers (mainly BPE with byte-fallback) maintain the ability to generate multiple tokenizations for a given input string as shown in Table 1. In this work, we aim to use these multiple tokenizations to improve self consistency in LLM reasoning.

An intuitive way to improve self consistency of an LLM in a reasoning task is to generate diverse reasoning paths (Wang et al., 2023) for a given problem. This is parallel to the multi-perspective reasoning as studied in Wang et al. (2006). Given these multiple reasoning paths, Wang et al. (2023) propose to select the answer produced by majority of reasoning paths as the final answer. Crucially, Wang et al. (2023) and many works that follow (Aggarwal et al., 2023; Li et al., 2024a; Jain et al., 2024; Li et al., 2024b) rely on diversity promoting text generation techniques such as Nucleus sampling (Holtzman et al., 2020) or temperature sampling to get diversity in the reasoning paths.

In this work, we propose to use the multiple tokenizations as a primary way to generate diverse reasoning paths. Our hypothesis is that different sequences of tokens for the same input string should naturally lead to different and diverse generations. This is advantageous since unlike prior sampling methods (Holtzman et al., 2020; Hewitt et al., 2022; Meister et al., 2023), it does not rely on the model's next-token distribution to have sufficient diversity. It is important to note that we need a principled approach to generate multiple tokenizations of the given string since randomly dropping $p\%$ of BPE merges can lead to degradation in performance as shown by Jain et al. (2023). We extend Kudo (2018) and present an approach (Section 2) to assign likelihood to a given tokenization and sampling a tokenization proportional to its likelihood.

[*]Equal contribution [1]IIT Bombay [2]Microsoft Research India. Correspondence to: Ashutosh Sathe <absathe@cse.iitb.ac.in>.

ICML 2024 Workshop on LLMs and Cognition

Table 1. **Multiple tokenizations of a given sentence using MISTRAL-7B BPE tokenizer.** The original input string (top row) can be tokenized into multiple possible sequences of valid tokens from the MISTRAL-7B vocabulary. "–" represents whitespace and "|" is used to indicate token boundary. The sequence of token IDs is also presented below the tokenization.

A sentence can have multiple tokenizations with the BPE or Unigram tokenizer.
–A|–sentence|–can|–have|–multiple|–token|izations|–with|–the|–|BP|E|–or|–Un|i|gram|–token|izer|.   {330, 12271, 541, 506, 5166, 6029, 13809, 395, 272, 28705, 9399, 28749, 442, 935, 28710, 1596, 6029, 4024, 28723}
–A|–sentence|–can|–have|–multiple|–token|izations|–with|–the|–B|PE|–or|–U|ni|gram|–token|ize|r|.   {330, 12271, 541, 506, 5166, 6029, 13809, 395, 272, 365, 1767, 442, 500, 3023, 1596, 6029, 653, 28712, 28723}
–A|–sentence|–can|–have|–multiple|–token|izations|–with|–the|–B|PE|–or|–Un|igr|am|–to|ken|izer|.   {330, 12271, 541, 506, 5166, 6029, 13809, 395, 272, 365, 1767, 442, 935, 3421, 314, 298, 2314, 4024, 28723}

2. Probabilistic Tokenization

Given an input string $\mathbf{X}$ and an existing vocabulary $\mathcal{V}$, we want to sample m different tokenizations $\{\mathbf{x}^1_{\text{tok}}, \ldots, \mathbf{x}^m_{\text{tok}}\}$ such that each $\mathbf{x}^i_{\text{tok}}$ is sampled proportional to $\Pr(\mathbf{x}^i_{\text{tok}}|\mathbf{X})$. Each tokenization $\mathbf{x}^i_{\text{tok}}$ is a sequence of tokens $[t^1, \ldots, t^{k_i}]$ where each $t^j \in \mathcal{V}$ and decoding $\mathbf{x}^i_{\text{tok}}$ using $\mathcal{V}$ gives $\mathbf{X}$ back.

2.1. Sampling a Tokenization

We follow Kudo (2018) and use a unigram language model to estimate $\Pr(\mathbf{x}^i_{\text{tok}}|\mathbf{X})$. This means we can write $\Pr(\mathbf{x}^i_{\text{tok}}|\mathbf{X})$ using the unigram probabilities $p(t^j)$ as,

$$\Pr(\mathbf{x}^i_{\text{tok}}|\mathbf{X}) = \prod_{j=1}^{k_i} p(t^j), \quad \text{where } \mathbf{x}^i_{\text{tok}} = [t^1, \ldots, t^{k_i}], \sum_{t^j \in \mathcal{V}} p(t^j) = 1 \tag{1}$$

Given the vocabulary $\mathcal{V}$ and dataset $\mathcal{D}_{\text{train}}$ of sentences, Kudo (2018) propose using the Expectation Maximization algorithm to estimate $p(t^j)$. This EM algorithm aims to maximize the marginal likelihood over the entire dataset considering $p(t^j)$ as latent variables. If $\mathbf{T}(\mathbf{X})$ denotes all possible tokenizations of a given sentence $\mathbf{X}$, the marginal can be written as,

$$\mathcal{L} = \sum_{s=1}^{|\mathcal{D}_{\text{train}}|} \log \Pr(\mathbf{X}_s) = \sum_{s=1}^{|\mathcal{D}_{\text{train}}|} \log \Pr\left(\sum_{\mathbf{x}_{\text{tok}} \in \mathbf{T}(\mathbf{X})} \Pr(\mathbf{x}_{\text{tok}})\right) \tag{2}$$

In practice, we opted for a simple counting based method to estimate $p(t^j)$. For every document in the $\mathcal{D}_{\text{train}}$, we first obtain a tokenization $\mathbf{x}^{\text{BPE}}_{\text{tok}}$ using the existing BPE tokenizer and simply count the total occurrences of $t^j \in \mathbf{x}^{\text{BPE}}_{\text{tok}}$ to estimate $p(t^j)$ as $\log p(t^j) = \log(\text{counts}(t^j)/N)$. Here, N indicates the total number of BPE tokens produced by the tokenizer on the entire $\mathcal{D}_{\text{train}}$. For special tokens such as beginning/end of sequence, padding or unknown tokens, we set $\log p(t^j) = 0$. We provide additional discussion on counting vs EM based estimation in Appendix A.3.

Once the unigram likelihoods are estimated for the entire vocabulary, we can efficiently get l-best tokenizations according to $\Pr(\mathbf{x}_{\text{tok}}|\mathbf{X})$ using the Forward-DP Backward-A* algorithm (Nagata, 1994). Following Kudo (2018), we sample from these l tokenizations as $\Pr(\mathbf{x}^i_{\text{tok}}|\mathbf{X}) \sim \Pr(\mathbf{x}^i_{\text{tok}})^\alpha / \sum_{i=1}^{l} \Pr(\mathbf{x}^i_{\text{tok}})^\alpha$ with α as a smoothing coefficient. We can now sample m tokenizations for a given $\mathbf{X}$ from the best $l \geq m$-best tokenizations.

Kudo (2018) also suggests a way to accurately sample from *all* ($l \to \infty$) possible tokenizations using Forward-Filtering and Backward-Sampling algorithm (Scott, 2002). We study effects of both $l \to \infty$ and fixed l on our sampled tokenizations and downstream tasks. We find that setting l to a fixed value or a value dependent on m can lead to less or superficial diversity in the reasoning paths sampled. More discussion on this is presented in Appendix A.4.

2.2. Evaluating Models with Multiple Tokenizations

We can use probabilistic tokenization to improve self-consistency of LLMs on various types of tasks. In this work, we are interested in specifically interested in improving self consistency of reasoning based tasks.

In these tasks, we expect the model to reason through a problem (chain-of-thought) step-by-step (Wei et al., 2022) before producing the final answer. Wang et al. (2023) has shown that sampling diverse reasoning paths from the model and picking the most common answer greatly improves the performance. While Wang et al. (2023) rely on sampling methods such as Nucleus sampling (Holtzman et al., 2020), we propose to use probabilistic tokenization as a way to boost diversity even further. Sampling based methods rely on the next-token distribution to be sufficiently diverse in order to generate diverse reasoning paths. As opposed to this, probabilistic tokenization directly changes the input to the model (i.e. the sequence of tokens) which should naturally lead to diverse generations. Given the reasoning problem as an input string $\mathbf{X}$, we sample m different tokenizations $\{\mathbf{x}_{\text{tok}}^1, \ldots, \mathbf{x}_{\text{tok}}^m\}$ as described above and generate m reasoning paths leading to m different answers as $\{\mathbf{Y}_{\text{pred}}^1, \ldots, \mathbf{Y}_{\text{pred}}^m\}$. The final answer is selected using majority vote similar to Wang et al. (2023). We also report "Oracle" task accuracy where we consider $\mathbf{X}$ solved if *any* one of the $\{\mathbf{Y}_{\text{pred}}^1, \ldots, \mathbf{Y}_{\text{pred}}^m\}$ matches the gold answer.

3. Experiments

We conduct experiments to evaluate the efficacy of the proposed probabilistic tokenization on various reasoning tasks. Our findings suggest that probabilistic tokenization can robustly improve the reasoning capabilities of many LLMs.

3.1. Setup

Probabilistic Tokenization We use a subset of the FINEWEB dataset (Penedo et al., 2024) consisting of roughly 10B tokens to estimate $p(t^j)$. For both reasoning and log-likelihood based tasks, we use $l \to \infty$ i.e. we sample m different from all possible tokenizations. The ablations on fixed l are presented in Appendix A.4.

Language Models We experiment with four transformer-based language model families: OLMO-7B (Groeneveld et al., 2024), GEMMA-2B, GEMMA-7B (Gemma Team et al., 2024), LLAMA3-8B (AI@Meta, 2024) and MISTRAL-7B (Jiang et al., 2023). We also include MAMBA-2.8B (Gu and Dao, 2023) as a representative non-transformer based language model.

Reasoning Tasks We consider four reasoning based tasks: MATH (Hendrycks et al., 2021), AQuA (Ling et al., 2017), GSM8k (Cobbe et al., 2021) and PIQA (Bisk et al., 2020). For each model, we report "Baseline" numbers which use the standard BPE tokenization. In "CoT + SC" baseline, we use the chain-of-thought prompting (Wei et al., 2022) with self-consistency (Wang et al., 2023) over 64 sampled reasoning paths with standard BPE tokenization. To sample diverse reasoning paths, we set the temperature $T = 0.2$ with top-k ($k = 64$) sampling. With "Probabilistic Tokenization", we use the same chain-of-thought prompt. However, we sample $m = 64$ different tokenizations and use greedy decoding to generate diverse reasoning paths.

3.2. Main Results

Table 2 shows results when applying consistency improving methods to reasoning tasks. The best (relative) performance gains for a particular task is **bolded**. We find that average performance gain obtained from "Probabilistic Tokenization" is higher than the average performance gain obtained by "CoT + SC". Interestingly, the "Probabilistic Oracle" is significantly better than "CoT + SC Oracle". By manually inspecting the reasoning traces, we find that "Probabilistic" tokenization is able to produce significantly diverse reasoning paths while "CoT + SC" reasoning paths are often just syntactically different. This can be advantageous as it gives the model more chances to improve the "Oracle" accuracy.

In our experiments, the only non-transformer based model i.e. MAMBA-2.8B showed mixed results when "CoT + SC" was applied. While the model was able to sample diverse reasoning paths, majority of them were often wrong or incomplete. Probabilistic tokenization on the other hand was able to consistently outperform "CoT + SC".

3.3. Error Analysis

To better understand the behavior of probabilistic tokenization, we study the cases where "CoT + SC" and "Probabilistic Tokenization" predictions do not match each other.

In Table 3, we show a representative example from GSM8k. The model used for generating these reasoning paths is LLAMA3-8B. We find that the 2 reasoning paths sampled by "CoT + SC" are logically equivalent. They use the same steps to in the calculation of total selling price. Importantly, they both stop early after that and get incorrect answer due to stopping early. This is a common failure mode for "CoT + SC" in reasoning tasks. As opposed to this, "Probabilistic Tokenization" is able to generate logically different reasoning paths. The second path sampled using probabilistic tokenization individually calculates the selling prices for different boards while second path calculates the total selling price directly from cost price. Notably, the difference in prompt tokenization also changes how the currency symbol is being used. In many sequences where the model verbalizes the currency ("dollar" as opposed to "$") in the reasoning paths, we noticed that the input sequence had tokenized each digit in the value separately i.e. "$16" would be tokenized as "$|1|6" as opposed to BPE's "$|16".

4. Conclusion

In this work, we propose "Probabilistic Tokenization" as a method to improve self consistency in LLMs. We present a principled way to sample multiple tokenizations of a given string using existing BPE tokenizers of pretrained LLMs. As probabilistic tokenization is an input transformation method rather than an output manipulation method, it can be generally applied to any task. We find that probabilistic tokenizations offers significant and consistent gains over baseline in 4 reasoning based tasks. Our analysis shows that the primary reason for success of probabilistic tokenization on reasoning tasks is its ability to generate logically diverse reasoning paths. We hope that probabilistic tokenization can be useful for providing deeper insights into the LLM's reasoning process and true abilities ("Oracle" choices), paving the way for more robust and interpretable AI systems.

Table 2. **Results with probabilistic tokenization on chain-of-thought based reasoning tasks.** The "Baseline" reports the accuracy or exact-match value with the standard BPE tokenization on that task *without* any chain-of-thought prompting. All other columns report changes relative to "Baseline". "Probabilistic Tokenization" (greedy decoding) outperforms chain-of-thought with self consistency ("CoT + SC") which uses temperature based diversity promoting sampling.

Task	Model	Baseline	CoT + SC Majority	CoT + SC Oracle	Probabilistic Majority	Probabilistic Oracle
MATH	OLMo-7B	3.90	**+45.64%**	+54.08%	+15.34%	+90.77%
	Gemma-2B	5.97	**+22.45%**	+85.37%	+19.03%	+94.97%
	Gemma-7B	10.91	+18.97%	+42.45%	**+20.87%**	+120.99%
	LLaMA3-8B	12.99	**+28.87%**	+82.92%	+17.45%	+81.60%
	LLaMA3-70B	18.18	+16.66%	+81.18%	**+17.91%**	+83.20%
	Mistral-7B	9.35	**+45.99%**	+78.86%	+20.10%	+67.27%
	Mamba-2.8B	3.12	-23.72%	+83.01%	**+18.29%**	+216.03%
AQuA	OLMo-7B	23.08	**+23.35%**	+53.27%	+18.45%	+38.60%
	Gemma-2B	15.38	+15.99%	+96.52%	**+18.63%**	+68.79%
	Gemma-7B	23.08	**+19.67%**	+37.63%	+16.62%	+31.63%
	LLaMA3-8B	11.54	**+17.68%**	+258.46%	+16.84%	+242.29%
	LLaMA3-70B	30.77	+11.25%	+13.85%	**+12.22%**	+15.15%
	Mistral-7B	23.54	**+19.75%**	+67.14%	+17.80%	+51.44%
	Mamba-2.8B	15.38	-7.67%	+95.85%	**+10.21%**	+91.81%
GSM8k	OLMo-7B	25.71	+12.29%	+99.04%	**+13.22%**	+96.42%
	Gemma-2B	5.91	+3.38%	+64.91%	**+20.81%**	+155.84%
	Gemma-7B	25.37	**+20.10%**	+89.27%	+12.61%	+79.90%
	LLaMA3-8B	37.71	**+20.53%**	+20.03%	+13.66%	+21.03%
	LLaMA3-70B	55.55	+19.15%	+35.54%	**+21.11%**	+32.33%
	Mistral-7B	29.66	+20.18%	+87.57%	**+20.40%**	+86.01%
	Mamba-2.8B	3.12	+5.45%	+113.46%	**+28.85%**	+68.59%
PIQA	OLMo-7B	75.89	+17.35%	+31.77%	**+20.04%**	+31.77%
	Gemma-2B	77.17	+16.77%	+22.46%	**+20.41%**	+22.46%
	Gemma-7B	79.89	+18.02%	+25.17%	**+19.40%**	+25.17%
	LLaMA3-8B	80.43	+16.26%	+17.49%	**+17.02%**	+17.49%
	LLaMA3-70B	80.98	+12.61%	+13.67%	**+13.67%**	+13.67%
	Mistral-7B	79.35	**+20.82%**	+26.02%	+16.52%	+26.02%
	Mamba-2.8B	73.91	+1.10%	+27.86%	**+1.62%**	+27.86%
	Average Δ		+16.39%	+64.46%	**+17.11%**	+71.40%

Table 3. **Comparison of reasoning paths.** "CoT + SC" paths often have superficial linguistic diversity while "Probabilistic Tokenization" paths have meaningful diversity in the reasoning paths.

Question	Sasha notices that prices for lumber have gone up 50% in the last few months after she bought some lumber. Since she has leftovers, she decides to sell them. She has leftover ten 2 x 4 x 10 boards that cost her $10 each. She also has five 4 x 4 x 10 boards she bought for $16 each. If she sells them all, how much profit does she make?
Answer	90
CoT + SC Path 1	Sasha has 10 2 x 4 x 10 boards that cost her $10 each. So she has 10 x 10 = $100 worth of 2 x 4 x 10 boards. She also has 5 4 x 4 x 10 boards she bought for $16 each. So she has 5 x 16 = $80 worth of 4 x 4 x 10 boards. In total, she has $100 + $80 = $180 worth of lumber. Since prices went up 50%, she can sell them for 1.5 times as much. 1.5 x 180 = 270. The answer is $270.
CoT + SC Path 2	Sasha has ten 2 x 4 x 10 boards that cost her $10 each. So she has 10 x $10 = $100 worth of 2 x 4 x 10 boards. She also has five 4 x 4 x 10 boards she bought for $16 each. So she has 5 x $16 = $80 worth of 4 x 4 x 10 boards. In total, she has $100 + $80 = $180 worth of lumber. Since prices have gone up 50%, she can sell them for 1.5 times the price she bought them for. So she can sell them for 1.5 x $180 = $270. The answer is $270.
Probabilistic Path 1	Sasha has 10 boards that cost her 10 dollars each. She also has 5 boards that cost her 16 dollars each. So she spent 10 x 10 + 5 x 16 = 100 + 80 = 180. Since prices went up 50%, she can sell the boards for 1.5 times the price she bought them for. So she can sell them for 1.5 x 180 = 270. She makes 270 - 180 = 90 dollars profit. The answer is 90.
Probabilistic Path 2	Sasha bought ten 2 x 4 x 10 boards for $10 each. So she spent 10 x 10 = $100. She also bought five 4 x 4 x 10 boards for $16 each. So she spent 5 x 16 = $80. In total, she spent $100 + $80 = $180. Since prices went up 50%, she can sell the boards for 1.5 times the price she bought them for. So she can sell the 2 x 4 x 10 boards for 1.5 x 10 = $15 each. She can sell the 4 x 4 x 10 boards for 1.5 x 16 = $24 each. So she can sell the 2 x 4 x 10 boards for 10 x 15 = $150. She can sell the 4 x 4 x 10 boards for 5 x 24 = $120. In total, she can sell the boards for $150 + $120 = $270. So she makes $270 - $180 = $90 profit. The answer is $90.

5. Limitations

In order to estimate the unigram probabilities $p(t^j)$ from Equation 1, we need access to a sufficiently large and diverse dataset of documents. While we used a sample of 10B tokens from a large scale web corpus, this may not always be the optimal choice for all the tasks. On domain specific tasks such as medical question answering or code generation, a web corpus might not be appropriate. Availability to such corpus is essential since errors in estimating $p(t^j)$ can result in suboptimal tokenizations which can hurt performance (Jain et al., 2023). On many of the general purpose tasks, this limitation can be addressed by making use of high quality, open source web corpora such as FINEWEB (Penedo et al., 2024) or DOLMA (Soldaini et al., 2024). Even in the cases where the actual corpus is not available but token level counts are available, our method can still be applied.

Furthermore, the model capability analysis (Appendix A.2) shows that improvements from probabilistic tokenization are greater for more capable models. If the base model is not very capable, it may not be robust to changes in tokenizations or generate superficially diverse reasoning paths. In such cases, probabilistic tokenization may even *hurt* the performance rather than improving. We highlight that this limitation is also present with other methods using chain-of-thought prompting or self-consistency.

References

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12375–12396, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.761. URL https://aclanthology.org/2023.emnlp-main.761.

AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Schulze Buschhoff, Charvi Jain, Alexander Arno Weber, Lena Jurkschat, Hammam Abdelwahab, Chelsea John, Pedro Ortiz Suarez, Malte Ostendorff, Samuel Weinbach, Rafet Sifa, Stefan Kesselheim, and Nicolas Flores-Herr. Tokenizer choice for llm training: Negligible or crucial?, 2024.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press, 2020. doi: 10.1609/AAAI.V34I05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Rozière. Getting the most out of your tokenizer for pre-training and domain adaptation, 2024.

Philip Gage. A new algorithm for data compression. *C Users J.*, 12(2):23–38, feb 1994. ISSN 0898-9788.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste

Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. Gemma: Open models based on gemini research and technology, 2024.

Daniel W Gorenflo and William D Crano. The multiple perspectives inventory: A measure of perspective-taking. *Swiss Journal of Psychology*, 57(3):163–177, 1998.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. Olmo: Accelerating the science of language models, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023.

Hao Guo, Kang Zheng, Xiaochuan Fan, Hongkai Yu, and Song Wang. Visual attention consistency under image transforms for multi-label image classification. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 729–739, 2019. doi: 10.1109/CVPR.2019.00082.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/a19744e268754fb0148b017647355b7b-Paper.pdf.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

John Hewitt, Christopher Manning, and Percy Liang. Truncation sampling as language model desmoothing. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3414–3427, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.249. URL https://aclanthology.org/2022.findings-emnlp.249.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. Baseline defenses for adversarial attacks against aligned language models, 2023.

Siddhartha Jain, Xiaofei Ma, Anoop Deoras, and Bing Xiang. Lightweight reranking for language model generations, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=e2TBb5y0yFf.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL https://aclanthology.org/P18-1007.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. Making language models better reasoners with step-aware verifier. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.291. URL https://aclanthology.org/2023.acl-long.291.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Bin Sun, Xinglin Wang, Heda Wang, and Kan Li. Turning dust into gold: Distilling complex reasoning capabilities from llms by leveraging negative data. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 18591–18599. AAAI Press, 2024a. doi: 10.1609/AAAI.V38I17.29821. URL https://doi.org/10.1609/aaai.v38i17.29821.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=ndR8Ytrzhh.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1015. URL https://aclanthology.org/P17-1015.

Siyang Liu, Naihao Deng, Sahand Sabour, Yilin Jia, Minlie Huang, and Rada Mihalcea. Task-adaptive tokenization: Enhancing long-form text generation efficacy in mental health and beyond. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15264–15281, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.944. URL https://aclanthology.org/2023.emnlp-main.944.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11:102–121, 2023. doi: 10.1162/tacl_a_00536. URL https://aclanthology.org/2023.tacl-1.7.

Masaaki Nagata. A stochastic Japanese morphological analyzer using a forward-DP backward-A* n-best search algorithm. In *COLING 1994 Volume 1: The 15th International Conference on Computational Linguistics*, Kyoto, Japan, August 1994. URL https://aclanthology.org/C94-1032.

Ranjita Naik, Varun Chandrasekaran, Mert Yuksekgonul, Hamid Palangi, and Besmira Nushi. Diversity of thought improves reasoning abilities of llms, 2024.

Guilherme Penedo, Hynek Kydlíček, Leandro von Werra, and Thomas Wolf. Fineweb, April 2024. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-dropout: Simple and effective subword regularization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.170. URL https://aclanthology.org/2020.acl-main.170.

Mark A Runco and Selcuk Acar. Divergent thinking as an indicator of creative potential. *Creativity research journal*, 24(1):66–75, 2012.

Steven L. Scott. Bayesian methods for hidden markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97(457):337–351, 2002. ISSN 01621459. URL http://www.jstor.org/stable/3085787.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162.

Aaditya K. Singh and DJ Strouse. Tokenization counts: the impact of tokenization on arithmetic in frontier llms, 2024.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. *arXiv preprint*, 2024.

Asa Cooper Stickland and Iain Murray. Diverse ensembles improve calibration, 2020.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Yan Wang, Enis Dogan, and Xiaodong Lin. The effect of multiple-perspective thinking on problem solving. In *Proceedings of the 7th International Conference on Learning Sciences*, ICLS '06, page 812–817. International Society of the Learning Sciences, 2006. ISBN 0805861742.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

Teresa Yeo, Oğuzhan Fatih Kar, and Amir Zamir. Robustness via cross-domain ensembles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12189–12199, October 2021.

Vilém Zouhar, Clara Meister, Juan Luis Gastaldi, Li Du, Tim Vieira, Mrinmaya Sachan, and Ryan Cotterell. A formal perspective on byte-pair encoding, 2023.

A. Appendix

A.1. Related Work

We compare probabilistic tokenization to works in multi view learning, tokenization based improvements in language modeling and methods improving self consistency.

Consistency Enhancements using Input Transforms We use a simple 2-layer neural network as our classifier to select a class given likelihood and selected option from multiple tokenizations. In principle, one can follow the rich literature on ensemble diversity (Stickland and Murray, 2020; Yeo et al., 2021; Han et al., 2018) to build a more sophisticated reranker for both log-likelihood and reasoning based tasks that is aware of the transformed input. (Guo et al., 2019) explores similar ideas to improve consistency of outputs in computer vision.

Tokenization Methods to Improve LLM Performance Tokenization plays a crucial role in the reasoning and domain understanding capabilities for LLMs (Dagan et al., 2024; Singh and Strouse, 2024). The popular BPE (Sennrich et al., 2016; Gage, 1994) and Unigram (Kudo, 2018) tokenizers are still being studied for better understanding (Zouhar et al., 2023). Some recent works argue that BPE might not be an optimal tokenization method for all tasks or domains (Liu et al., 2023; Ali et al., 2024). Our work is orthogonal to these directions since we do not aim to modify the existing tokenizer and LLM in any way.

Self Consistency and Diversity of Thoughts in Reasoning Several works improve LLM reasoning capabilities using self consistency and chain-of-thought prompting(Wei et al., 2022; Kojima et al., 2022; Wang et al., 2023; Yao et al., 2023). Following this, many works improve self consistency by either changing the stopping criteria (Aggarwal et al., 2023; Li et al., 2024b) or by designing a sophisticated voting fucntion to replace majority voting (Li et al., 2024a; Jain et al., 2024). Diversity of thoughts (reasoning paths) is also shown to be an important factor limiting LLM's reasoning ability (Li et al., 2023; Naik et al., 2024). Our work is relevant in this direction as it aims to improve the diversity in reasoning paths using tokenization.

A.2. Effect of model scale

We study the effectiveness of probabilistic tokenization at various model parameter scales. As shown in Table 4, we find that probabilistic tokenization robustly improves the performance across many model parameters scale. We find that relative improvements are highest at 7-8B parameter range, roughly when the "reasoning" capabilities start to emerge (Brown et al., 2020). At smaller parameter counts, less gains could be explained by the model's weaker reasoning

Table 4. **Comparing effectiveness of probabilistic tokenization at various model parameter scales.** Average task performance is reported for each method with percent improvement over "Baseline" in the bracket.

Model	Baseline	CoT + SC Majority	CoT + SC Oracle	Probabilistic Majority	Probabilistic Oracle
GEMMA-2B	26.11	29.93(+14.65%)	43.68(+67.31%)	31.26(+19.72%)	48.43(+85.51%)
GEMMA-7B	34.81	41.49(+19.19%)	51.74(+42.54%)	40.86(+17.38%)	57.24(+64.42%)
LLAMA3-8B	35.67	43.10(+20.83%)	69.45(+94.73%)	41.46(+16.24%)	67.98(+90.60%)
LLAMA3-70B	46.37	53.29(+14.92%)	63.09(+36.06%)	53.89(+16.23%)	63.10(+36.09%)

abilities (Brown et al., 2020). Similarly, larger models that are already capable of generating diverse reasoning paths (as evidenced by lower improvements in "Oracle" numbers) show comparable gains over "Baseline" with both "CoT + SC" and "Probabilstic" methods.

A.3. Counting vs EM for estimating $p(t^j)$

As opposed to using EM to estimate $p(t^j)$ in Equation 1, we resort to a simpler counting based method. We acknowledge that this could result in somewhat distorted unigram probabilities but is significantly faster as it does not have to enumerate over $\mathbf{T}(\mathbf{X})$. We provide additional results (Table 5) on applying EM to a smaller, 100M token subset to conclude that both methods perform comparably.

A.4. Fixed l vs $l \to \infty$ for sampling tokenizations

We compare using fixed vs infinite l for sampling a tokenization in Table 6. Our findings suggest that considering a fixed window of top-l tokenizations may not offer sufficient diversity leading to redundant generations which explain lesser improvements in "Oracle" as well as "Majority" numbers.

A.5. Resources Used

We used a single NVIDIA A100 GPU with a 64 core AMD CPU to run our inferences. The estimated total GPU hours is 600 hours. Our implementation is based on the `lm-evaluation-harness`: `https://github.com/EleutherAI/lm-evaluation-harness` and `sentencepiece`: `https://github.com/google/sentencepiece`. We use 7 models for our experiments, all of which are opensource. The list of model and the URL with checkpoints available and licenses are listed below:

OLMO-7B : `https://huggingface.co/allenai/OLMo-7B` **License:** Apache-2.0

GEMMA-2B : `https://huggingface.co/google/gemma-2b` **License:** Gemma

GEMMA-7B : `https://huggingface.co/google/gemma-7b` **License:** Gemma

LLAMA3-8B : `https://huggingface.co/meta-llama/Meta-Llama-3-8B` **License:** llama3

LLAMA3-70B : `https://huggingface.co/meta-llama/Meta-Llama-3-70B` **License:** llama3

MISTRAL-7B : `https://huggingface.co/mistralai/Mistral-7B-v0.1` **License:** Apache-2.0

MAMBA-2.8B : `https://huggingface.co/state-spaces/mamba-2.8b-hf` **License:** Apache-2.0

Subsequently we used FINEWEB dataset available at `https://huggingface.co/datasets/HuggingFaceFW/fineweb` to collect frequencies of the tokens in the for probabilistic tokenization mentioned in Section 2.

Table 5. **Comparing effect of using counting vs EM to estimate EM probabilities.** Average task performance is reported. Both methods perform comparably on a GEMMA-2B model.

Reasoning Tasks					
Model	Baseline	CoT + SC Majority	CoT + SC Oracle	Probabilistic Majority	Probabilistic Oracle
GEMMA-2B (EM)	26.11	29.13	38.68	29.26	45.45
GEMMA-2B (Counting)	26.11	27.93	37.68	29.81	45.45

Loglikelihood Tasks					
Model	Baseline	Most Likely	Majority	Classifier	Oracle
GEMMA-2B (EM)	39.00	38.13	41.49	49.69	54.01
GEMMA-2B (Counting)	39.00	37.63	41.58	51.01	55.56

Table 6. **Comparing effect of using counting vs EM to estimate EM probabilities.** Average task performance is reported. Both methods perform comparably on a GEMMA-2B model.

Reasoning Tasks					
Model	Baseline	CoT + SC Majority	CoT + SC Oracle	Probabilistic Majority	Probabilistic Oracle
GEMMA-2B ($l = m^2$)	26.11	27.13	33.18	26.26	35.35
GEMMA-2B ($l \rightarrow \infty$)	26.11	**29.93**	**43.68**	**31.26**	**48.43**

Loglikelihood Tasks					
Model	Baseline	Most Likely	Majority	Classifier	Oracle
GEMMA-2B ($l = m^2$)	39.00	**38.98**	39.19	45.16	46.15
GEMMA-2B ($l \rightarrow \infty$)	39.00	38.64	**42.98**	**53.96**	**56.06**