Language Models Resist Alignment

Jiaming Ji[*] **Kaile Wang**[*] **Tianyi Qiu**[*] **Boyuan Chen**[*]
Jiayi Zhou **Changye Li** **Hantao Lou** **Josef Dai** **Yaodong Yang**[†]

PKU-Alignment Team, Peking University

Abstract

Large language models (LLMs) may exhibit undesirable behaviors. Recent efforts have focused on aligning these models to prevent harmful generation. Despite these efforts, studies have shown that even a well-conducted alignment process can be easily circumvented, whether intentionally or accidentally. Does alignment fine-tuning have robust effects on models, or is merely *superficial*? In this work, we answer this question through both theoretical and empirical means. Empirically, we demonstrate the *elasticity* of post-alignment models, *i.e.*, the tendency to revert to the behavior distribution formed during the pre-training phase upon further fine-tuning. Using compression theory, we formally derive that such fine-tuning process *disproportionately* undermines alignment compared to pre-training, potentially by orders of magnitude. We conduct experimental validations to confirm the presence of *elasticity* across models of varying types and sizes. Specifically, we find that model performance declines rapidly before reverting to the pre-training distribution, after which the rate of decline drops significantly. We further reveal that *elasticity* positively correlates with increased model size and the expansion of pre-training data. Our discovery signifies the importance of taming the inherent elasticity of LLMs, thereby overcoming the resistance of LLMs to alignment finetuning.

1 Introduction

Large language models (LLMs) have exhibited remarkable capabilities [1, 2]. However, given the inevitable biases and harmful content in the training dataset [3, 4], these models often exhibit behaviors that deviate from the designer' intentions, a phenomenon we refer to as *model misalignment*. Therefore, aligning LLMs to ensure their behaviors remain consistent with human intentions and values is particularly important [2, 5, 6, 7, 8].

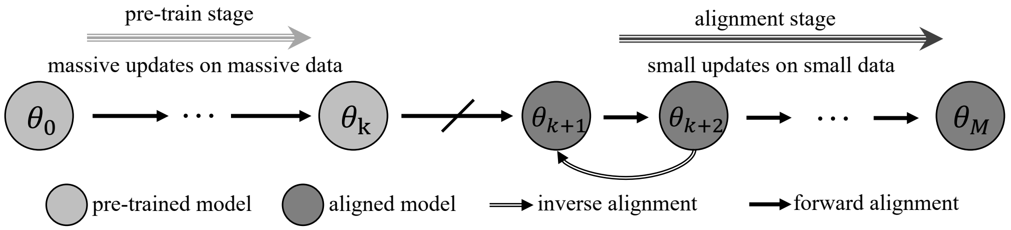


Figure 1: **Forward and Inverse Alignment.** LLMs undergo numerous iterations during pre-training, forming a stable parameter distribution. Subsequent alignment procedures fine-tune this distribution to reflect human intentions. Our research question is: During further fine-tuning, is it harder to deviate from the stable parameter distribution formed during pre-training than to maintain it?

[*] Equal contributions, [†] corresponding author. Code url: `https://github.com/PKU-Alignment/llms-resist-alignment`.

Socially Responsible Language Modelling Research (SoLaR) Workshop at NeurIPS 2024.

So far, we mainly steer or align models with finetuning-based methods including supervised fine-tuning (SFT), reinforcement learning from human feedback (RLHF) [9], and more [8, 10, 11, 12, 13, 14]. However, it remains unclear whether such methods truly penetrate the model representations or merely perform *superficial alignment*. Recent studies [15, 16] have shown that the effect of alignment process is *superficial*, *e.g.*, models undergoing safety alignment can become unsafe again with minimal fine-tuning. Furthermore, fine-tuning aligned LLMs on non-malicious datasets can weaken the models' safety mechanisms as well [17, 18]. Why is alignment so fragile?

This counterintuitive phenomenon further prompts exploration into the inverse process of alignment: assuming that the alignment process of LLMs is indeed limited to superficial alignment, is it then possible to perform an inverse operation of alignment, *i.e.*, to achieve the reversal of the alignment process through a series of technical measures? In this work, we investigate the possibility of reversing or revoking the alignment process in LLMs, a concept we refer to as *inverse alignment*. In a word, we aim to answer the under-explored question:

Do the parameters of language models exhibit elasticity, thereby resisting alignment?

Our main contribution is Theorem 2.6. We show the elasticity of post-alignment models using tools from compression theory, demonstrating that models tend to retain the distribution learned from the pre-train dataset while forgetting the effects of subsequent fine-tuning. We also prove that after subsequent fine-tuning, the changes in the *normalized compression rates* of the model for different datasets are proportional to their respective sizes. Furthermore, we experimentally demonstrate that the phenomenon of elasticity in post-alignment models is present across models of various scales, emphasizing that when conducting subsequent safety alignment, it is necessary to consider the impact of model elasticity on alignment effectiveness.

2 Elasticity in Large Language Models as Resistance to Alignment

Our analysis is conducted with the background of the compression theory of language models (Appendix B). In this section, we formulate the concept of *elasticity* and prove its existence in LLMs. It gives rise to the possibility of *inverse alignment*, thereby constituting resistance to alignment. We start by defining these concepts.

Definition 2.1 (*The Elasticity of LLM Parameters*)**.** Given an LLM $p_{\boldsymbol{\theta}_0}$, and the transformation $p_{\boldsymbol{\theta}_0} \xmapsto{f(\mathcal{D}_a)} p_{\boldsymbol{\theta}_1}$, *elasticity* is said to exist in $(p_{\boldsymbol{\theta}_0}, \mathcal{D}_a)$ if there is an algorithmically simple *inverse operation* g and a dataset $\mathcal{D}_b$ such that $|\mathcal{D}_b| \ll |\mathcal{D}_a|$, with the property that

$$p_{\boldsymbol{\theta}_1} \xmapsto{g(\mathcal{D}_b)} p_{\boldsymbol{\theta}_0'} \text{ and } \rho(p_{\boldsymbol{\theta}_0'}, p_{\boldsymbol{\theta}_0}) \leq \epsilon.$$

2.1 The Token-level Response Tree for Compression Analysis

We explain that the post-alignment models contain elasticity using information-theoretic concepts specifically related to data compression, given the analytical equivalence and practical consistency between compression and prediction performance (Section B.2). We first present our formulation for the compression protocol, using tokenized sequences as the input and output modality. Due to space constraints, we selectively present key definitions, assumptions, and theorems. Please refer to Appendix C for the full collection of assumptions and proofs.

Assumption 2.2 (Binary Tokens)**.** For the purpose of this analysis, consider the tokenization process employed on the datasets. Without loss of generality (since any vocabulary sizes can be approximately reduced to the binary case with a uniform multiplier to the code length), we assume that the token table contains only binary tokens (specifically `0/1`) and is uniform across all datasets.

Definition 2.3 (Token-level Response Tree $\mathcal{T}$)**.** Consider the dataset $\mathcal{D} = \{\boldsymbol{z}_i \in \{0|1\}^\infty \mid i = 1, 2, \cdots\}$, where each $\boldsymbol{z}_i$ represents a binary response in $\mathcal{D}$. The token-level response tree, denoted as $\mathcal{T}_\mathcal{D}$, is structured such that each node contains child nodes labeled `0` or `1`. Additionally, each binary node terminates with an end-of-sequence (`EOS`) token leaf node. The path from the root to a leaf node delineates each response $\boldsymbol{z}_i$. The likelihood of a response that a token represents is indicated by the probability associated with its `EOS` token node. Meanwhile, the probability of any binary node is defined as the sum of the probabilities of all its child nodes.

Definition 2.4 (Compression with Finite-parameter Models)**.** For a finite-parameter model $p_{\boldsymbol{\theta}}(\cdot)$ and the dataset $\mathcal{D}$, the compression protocol using $p_{\boldsymbol{\theta}}$ for $\mathcal{D}$ is defined as follows: For $\mathcal{D}$'s TRT $\mathcal{T}_{\mathcal{D}}$, 1) Prune the $\mathcal{T}_{\mathcal{D}}$ in the manner of Remark C.2, retaining only the top d layers of $\mathcal{T}_{\mathcal{D}}$, where d is a quantity determined by $p_{\boldsymbol{\theta}}$'s parameter count. 2) Compress the pruned response tree using Huffman coding. In this scheme, each response from the root token to a `0/1` token is considered as a letter in the Huffman coding alphabet, with the probability of the `EOS` token for the corresponding `0/1` token serving as the probability of that letter.

Theorem 2.5 (Ideal Code Length with Finite-parameter Models)**.** *Consider a finite parameter model $p_{\boldsymbol{\theta}}(\cdot)$ which is training on dataset $\mathcal{D}$, the ideal code length $\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})$ of a random response $\boldsymbol{x}$ compressed by $p_{\boldsymbol{\theta}}$ can be expressed as follows,*

$$\mathbb{E}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})\right] = \left\lceil \frac{|\boldsymbol{x}|}{d} \right\rceil \left\lceil -\sum_{i=1}^{d} \sum_{j=1}^{2^{i-1}} p_{ij} \log p_{ij} \right\rceil, \tag{1}$$

where d represents the depth of the $\mathcal{T}_{\mathcal{D}}$ after pruning under Definition 2.4 protocol, and p_{ij} represents the probability values of the `EOS` nodes for the j-th node at the i-th layer.

2.2 Formal Derivation of Elasticity

Our primary focus is on studying the behavioral changes of the model during subsequent fine-tuning after pre-training and one round of SFT. Therefore, it can be assumed that in the analysis of model compression, only three datasets are involved: the pre-training dataset $\mathcal{D}_1$, the fine-tuning dataset $\mathcal{D}_2$ in the first round of SFT, and the perturbation dataset $\mathcal{D}_3$ used in subsequent fine-tuning. Without loss of generality, we can consider these three datasets to be independent and differently distributed.

Due to the different scales of compression rates obtained for different datasets by the model, we consider normalizing the compression rates for different datasets when stating Theorem 2.6.

Theorem 2.6 (Elasticity of Language Models)**.** *Consider datasets $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ each with a Pareto mass distribution (Assumption C.15), and the model $p_{\boldsymbol{\theta}}(\cdot)$ trained on $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \mathcal{D}_3$. When dataset $\mathcal{D}_3$'s data volume $|\mathcal{D}_3|$ changes, the normalized reciprocal of the compression rate $\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}$, $\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}$ (Definition C.16) of the model for $\mathcal{D}_1$ and $\mathcal{D}_2$ satisfies:*

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l} = \Theta\left(k\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l}\right); \frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l} > 0, \frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l} > 0 \tag{2}$$

where $l = \frac{|\mathcal{D}_3|}{|\mathcal{D}_2|} \ll 1$, $k = \frac{|\mathcal{D}_1|}{|\mathcal{D}_2|} \gg 1$.

Theorem 2.6 illustrates that as the amount of data in the perturbation dataset $\mathcal{D}_3$ increases, the normalized compression rates of the model for both the pre-train dataset $\mathcal{D}_1$ and the SFT dataset $\mathcal{D}_2$ decrease, but the rate of decrease for the pre-train dataset is smaller than that for the SFT dataset by a factor of $\Theta(k)$, which in practice is many orders of magnitude.

3 Experiments

In the previous sections, we proved that LLMs achieve stable behavioral distributions during the pre-training stage through *massive updates on massive data*. The alignment stage with *small updates on small data* does not erase such a distribution, and subsequent fine-tuning can easily restore this pre-alignment distribution. Building on top of this discovery, in this section, we primarily aim to answer the following questions: 1) Does *elasticity* consistently exist across models of different types and sizes? 2) Is *elasticity* correlated with model parameter size and pre-training data size?

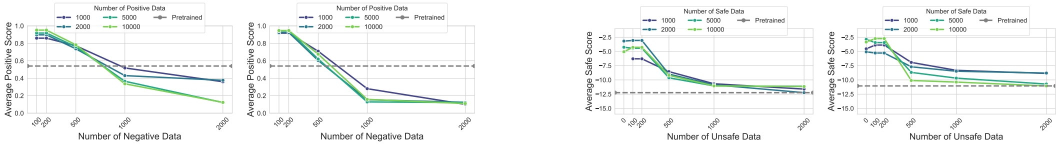
3.1 Experiment Setup

To verify the elasticity performance of pre-trained large language models, we select two tasks: positive generation (IMDb dataset [19]) and single-turn safe conversation (PKU-SafeRLHF dataset [4]). We verify elasticity in popular pre-trained LLMs such as Gemma-2B [20] and Llama2-7B [7], and use score models provided by existing research to complete the performance evaluation. For detailed experimental setup, please refer to Appendix D.1.

3.2 Experiment Results

We conduct experiments on the existence of elasticity in language models and the relationship between elasticity, model size, and pre-training data amount. For additional experimental results under various settings, please refer to Appendix D.2.

Existence of Elasticity. We evaluate the elasticity phenomenon on Llama2-7B [7] and Gemma-2B [20]. The experimental results in Figure 2 show that, for models fine-tuned with a large amount of positive sample data, only a small amount of negative sample fine-tuning is needed to quickly revert to the pre-training distribution, *i.e.*, to make the curve drop below the gray dashed line. Subsequently, the rate of performance decline slows down and tends to stabilize.

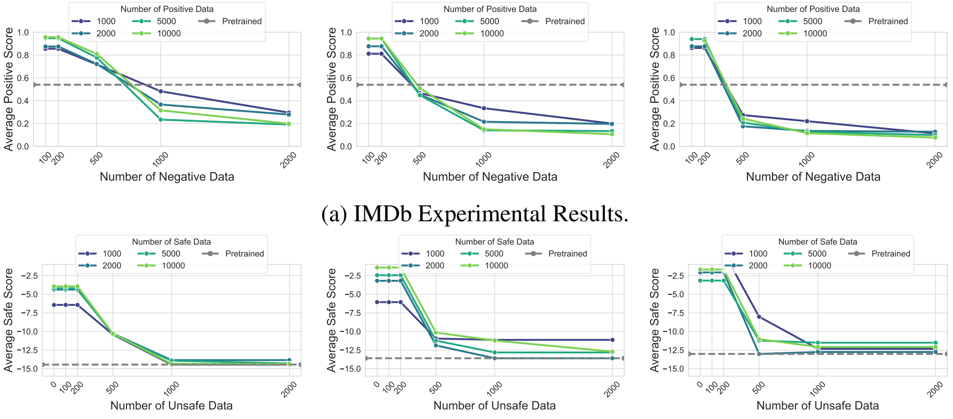


(a) IMDb Experimental Results.

(b) PKU SafeRLHF Experimental Results.

Figure 2: Experimental Results for Validating the Existence of Elasticity. The left and right sides of each subfigure correspond to the performance of Gemma-2B [20] and Llama2-7B [7], respectively, while the caption identifies the dataset. The model performance rapidly declines before reverting to the pre-training distribution, after which the decline becomes significantly slower. This phenomenon is defined as the elasticity of LLMs.

Elasticity Increases with Model Size. To examine how elasticity changes with model parameter size, we conducted experiments on Qwen models [21] with 0.5B, 4B, and 7B parameters. In Figure 3, as the model parameter size increases, the initial performance decline due to negative data fine-tuning is faster, while the subsequent decline is slower. This indicates that as the parameter size increases, there is an increased elasticity in response to both positive and negative data.

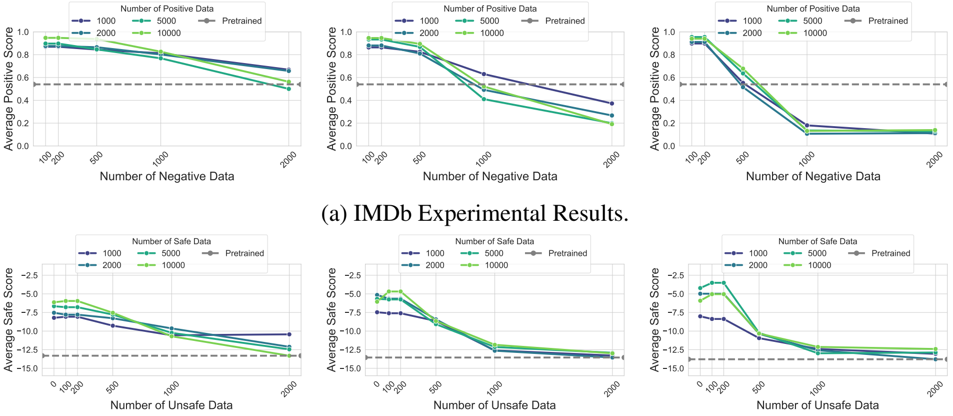


(a) IMDb Experimental Results.

(b) PKU SafeRLHF Experimental Results.

Figure 3: Experimental Results for Validating the Positive Correlation Between Elasticity and Model Parameter Size. Each subfigure from left to right shows the changes in LLMs with parameter sizes of 0.5B, 4B, and 7B, respectively, while the caption identifies the dataset.

Elasticity Increases with Pre-training Data Amount. To verify that elasticity increases with the growth of pre-training data, we conduct the same experiments on multiple pre-training slices released by TinyLlama [22]. As shown in Figure 4, when the pre-training data volume increases, the initial performance decline due to negative data fine-tuning is faster, while the subsequent decline is slower. It demonstrates that larger pre-training data volumes reinforce the elasticity of LLMs.



(a) IMDb Experimental Results.

(b) PKU SafeRLHF Experimental Results.

Figure 4: Experimental Results for Validating the Positive Correlation Between Elasticity and Pre-training Data Size. Each subfigure from left to right shows the changes in pre-training data sizes of 2.0T, 2.5T, and 3.0T, while the caption identifies the dataset.

4 Conclusion and Outlook

In this work, prompted by the fragility of alignment, we show the existence of elasticity in language models though both theoretical and experimental lens, thereby demonstrating their tendency to resist alignment. Specifically, large pre-training datasets and large parameter counts enhance the model's anti-interference capabilities, making subsequent alignment procedures easy to undo via further finetuning. We experiment on a variety of models and datasets, and validate elasticity across different sliced models during pre-training and alignment. Extensive results confirm that language models exhibit elasticity, indicating that *language models resist alignment*.

4.1 Ethic Impacts

Rethinking Fine-tuning. The influence of noisy pre-training corpora may cause models to exhibit unexpected behaviors. Alignment methods seek to modify LLMs' distributions efficiently to enhance helpfulness, harmlessness, and honesty. From the inverse alignment perspective, we need more robust methods to ensure that modifications to model parameters go beyond superficial changes. However, certain inducement measures might compromise the alignment strategy, potentially causing severe harm. Additionally, we should prioritize data cleansing during pre-training, rigorously managing noisy and biased data to enhance the malleability of the model's final distribution [23, 17].

Rethinking Open-sourcing. Open-sourcing is a double-edged sword [24]. On one hand, it can pose significant risks, such as model misuse, which could endanger public safety through fine-tuning for malicious purposes [25] or system jailbreaking [26]. On the other hand, restricting access may foster monopolistic practices, while open-sourcing cutting-edge models promote a robust open-source community and enhance the usability of these models. Furthermore, it facilitates the decentralization of AI technology [27]. From the perspective of model robustness, well-aligned models can be rendered unsafe with a very small amount of unsafe data. Also, fine-tuning aligned LLMs on non-malicious datasets can weaken the models' safety mechanisms [28]. Ensuring that open-source models are not misused is a critical challenge, and discoveries in this work prompt the community to build robust alignment algorithms, thereby overcoming the model's tendency to resist alignment.

Limitations and Future Work Theory-wise, the primary limitation of our work is our specification of the *mass distribution* (Assumption C.15), and empirical studies on the exact form of this distribution shall be valuable. Experiment-wise, we have not systematically validated elasticity throughout the entire lifecycle of pre-training and alignment phases, due to cost constraints. In future works, we plan to focus more on whether this phenomenon is universally applicable, such as in multimodal models. Additionally, we aim to theoretically uncover the relationship between model elasticity and *scaling laws*, specifically determining the amount of training data required for elasticity to manifest and quantitatively analyzing whether elasticity intensifies as model parameters increase.

References

[1] Zhao, Wayne Xin and Zhou, Kun and Li, Junyi and Tang, Tianyi and Wang, Xiaolei and Hou, Yupeng and Min, Yingqian and Zhang, Beichen and Zhang, Junjie and Dong, Zican and others. A Survey of Large Language Models. *arXiv preprint arXiv:2303.18223*, 2023.

[2] Achiam, Josh and Adler, Steven and Agarwal, Sandhini and Ahmad, Lama and Akkaya, Ilge and Aleman, Florencia Leoni and Almeida, Diogo and Altenschmidt, Janko and Altman, Sam and Anadkat, Shyamal and others. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Bai, Yuntao and Jones, Andy and Ndousse, Kamal and Askell, Amanda and Chen, Anna and DasSarma, Nova and Drain, Dawn and Fort, Stanislav and Ganguli, Deep and Henighan, Tom and others. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[4] Ji, Jiaming and Liu, Mickel and Dai, Josef and Pan, Xuehai and Zhang, Chi and Bian, Ce and Chen, Boyuan and Sun, Ruiyang and Wang, Yizhou and Yang, Yaodong. BeaverTails: Towards Improved Safety Alignment of LLM via a Human-Preference Dataset. *Advances in Neural Information Processing Systems*, 36, 2024.

[5] Ji, Jiaming and Qiu, Tianyi and Chen, Boyuan and Zhang, Borong and Lou, Hantao and Wang, Kaile and Duan, Yawen and He, Zhonghao and Zhou, Jiayi and Zhang, Zhaowei and others. AI Alignment: A Comprehensive Survey. *arXiv preprint arXiv:2310.19852*, 2023.

[6] Stephen Casper and Xander Davies and Claudia Shi and Thomas Krendl Gilbert and Jérémy Scheurer and Javier Rando and Rachel Freedman and Tomasz Korbak and David Lindner and Pedro Freire and Tony Tong Wang and Samuel Marks and Charbel-Raphael Segerie and Micah Carroll and Andi Peng and Phillip Christoffersen and Mehul Damani and Stewart Slocum and Usman Anwar and Anand Siththaranjan and Max Nadeau and Eric J Michaud and Jacob Pfau and Dmitrii Krasheninnikov and Xin Chen and Lauro Langosco and Peter Hase and Erdem Biyik and Anca Dragan and David Krueger and Dorsa Sadigh and Dylan Hadfield-Menell. Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback. *Transactions on Machine Learning Research*, 2023. Survey Certification.

[7] Touvron, Hugo and Martin, Louis and Stone, Kevin and Albert, Peter and Almahairi, Amjad and Babaei, Yasmine and Bashlykov, Nikolay and Batra, Soumya and Bhargava, Prajjwal and Bhosale, Shruti and others. Llama 2: Open Foundation and Fine-tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023.

[8] Rafailov, Rafael and Sharma, Archit and Mitchell, Eric and Manning, Christopher D and Ermon, Stefano and Finn, Chelsea. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. *Advances in Neural Information Processing Systems*, 36, 2024.

[9] Ouyang, Long and Wu, Jeffrey and Jiang, Xu and Almeida, Diogo and Wainwright, Carroll and Mishkin, Pamela and Zhang, Chong and Agarwal, Sandhini and Slama, Katarina and Ray, Alex and others. Training Language Models to Follow Instructions with Human Feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[10] Bai, Yuntao and Kadavath, Saurav and Kundu, Sandipan and Askell, Amanda and Kernion, Jackson and Jones, Andy and Chen, Anna and Goldie, Anna and Mirhoseini, Azalia and McKinnon, Cameron and others. Constitutional AI: Harmlessness from AI Feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[11] Lee, Harrison and Phatale, Samrat and Mansoor, Hassan and Lu, Kellie and Mesnard, Thomas and Bishop, Colton and Carbune, Victor and Rastogi, Abhinav. RLAIF: Scaling Reinforcement Learning from Human Feedback with AI Feedback. *arXiv preprint arXiv:2309.00267*, 2023.

[12] Josef Dai and Xuehai Pan and Ruiyang Sun and Jiaming Ji and Xinbo Xu and Mickel Liu and Yizhou Wang and Yaodong Yang. Safe RLHF: Safe Reinforcement Learning from Human Feedback. In *The Twelfth International Conference on Learning Representations*, 2024.

[13] Gulcehre, Caglar and Paine, Tom Le and Srinivasan, Srivatsan and Konyushkova, Ksenia and Weerts, Lotte and Sharma, Abhishek and Siddhant, Aditya and Ahern, Alex and Wang, Miaosen and Gu, Chenjie and others. Reinforced Self-Training (ReST) for Language Modeling. *arXiv preprint arXiv:2308.08998*, 2023.

[14] Dong, Hanze and Xiong, Wei and Goyal, Deepanshu and Zhang, Yihan and Chow, Winnie and Pan, Rui and Diao, Shizhe and Zhang, Jipeng and KaShun, SHUM and Zhang, Tong. RAFT: Reward rAnked FineTuning for Generative Foundation Model Alignment. *Transactions on Machine Learning Research*, 2023.

[15] Yang, Xianjun and Wang, Xiao and Zhang, Qi and Petzold, Linda and Wang, William Yang and Zhao, Xun and Lin, Dahua. Shadow Alignment: The Ease of Subverting Safely-Aligned Language Models. *arXiv preprint arXiv:2310.02949*, 2023.

[16] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

[17] Xiangyu Qi and Yi Zeng and Tinghao Xie and Pin-Yu Chen and Ruoxi Jia and Prateek Mittal and Peter Henderson. Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! In *The Twelfth International Conference on Learning Representations*, 2024.

[18] Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P Dick, Hidenori Tanaka, Edward Grefenstette, Tim Rocktäschel, and David Scott Krueger. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks. *arXiv preprint arXiv:2311.12786*, 2023.

[19] Maas, Andrew and Daly, Raymond E and Pham, Peter T and Huang, Dan and Ng, Andrew Y and Potts, Christopher. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.

[20] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open Models Based on Gemini Research and Technology. *arXiv preprint arXiv:2403.08295*, 2024.

[21] Bai, Jinze and Bai, Shuai and Chu, Yunfei and Cui, Zeyu and Dang, Kai and Deng, Xiaodong and Fan, Yang and Ge, Wenbin and Han, Yu and Huang, Fei and others. Qwen Technical Report. *arXiv preprint arXiv:2309.16609*, 2023.

[22] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. TinyLlama: An Open-Source Small Language Model. *arXiv preprint arXiv:2401.02385*, 2024.

[23] He, Luxi and Xia, Mengzhou and Henderson, Peter. What's in Your" Safe" Data?: Identifying Benign Data that Breaks Safety. *arXiv preprint arXiv:2404.01099*, 2024.

[24] Seger, Elizabeth and Dreksler, Noemi and Moulange, Richard and Dardaman, Emily and Schuett, Jonas and Wei, K and Winter, Christoph and Arnold, Mackenzie and hÉigeartaigh, Seán Ó and Korinek, Anton and others. Open-Sourcing Highly Capable Foundation Models: An evaluation of risks, benefits, and alternative methods for pursuing open-source objectives. *arXiv preprint arXiv:2311.09227*, 2023.

[25] Goldstein, Josh A and Sastry, Girish and Musser, Micah and DiResta, Renee and Gentzel, Matthew and Sedova, Katerina. Generative Language Models and Automated Influence Operations: Emerging Threats and Potential Mitigations. *arXiv preprint arXiv:2301.04246*, 2023.

[26] Zou, Andy and Wang, Zifan and Kolter, J Zico and Fredrikson, Matt. Universal and Transferable Adversarial Attacks on Aligned Language Models. *arXiv preprint arXiv:2307.15043*, 2023.

[27] Jeremy Howard. AI Safety and the Age of Dislightenment. `https://www.fast.ai/posts/2023-11-07-dislightenment.html`, 2023.

[28] Qi, Xiangyu and Zeng, Yi and Xie, Tinghao and Chen, Pin-Yu and Jia, Ruoxi and Mittal, Prateek and Henderson, Peter. Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! In *The Twelfth International Conference on Learning Representations*, 2023.

[29] Yang, Kevin and Klein, Dan and Celikyilmaz, Asli and Peng, Nanyun and Tian, Yuandong. RLCD: Reinforcement Learning from Contrastive Distillation for Language Model Alignment. *arXiv preprint arXiv:2307.12950*, 2023.

[30] Hubinger, Evan and Denison, Carson and Mu, Jesse and Lambert, Mike and Tong, Meg and MacDiarmid, Monte and Lanham, Tamera and Ziegler, Daniel M and Maxwell, Tim and Cheng, Newton and others. Sleeper Agents: Training Deceptive LLMs that Persist Through Safety Training. *arXiv preprint arXiv:2401.05566*, 2024.

[31] Wei, Boyi and Huang, Kaixuan and Huang, Yangsibo and Xie, Tinghao and Qi, Xiangyu and Xia, Mengzhou and Mittal, Prateek and Wang, Mengdi and Henderson, Peter. Assessing the Brittleness of Safety Alignment via Pruning and Low-Rank Modifications. *arXiv preprint arXiv:2402.05162*, 2024.

[32] Cao, Yinzhi and Yang, Junfeng. Towards Making Systems Forget with Machine Unlearning. In *2015 IEEE symposium on security and privacy*, pages 463–480. IEEE, 2015.

[33] Ginart, Antonio and Guan, Melody and Valiant, Gregory and Zou, James Y. Making AI Forget You: Data Deletion in Machine Learning. *Advances in neural information processing systems*, 32, 2019.

[34] Swanand Ravindra Kadhe, Anisa Halimi, Ambrish Rawat, and Nathalie Baracaldo. Fairsisa: Ensemble post-processing to improve fairness of unlearning in llms. *arXiv preprint arXiv:2312.07420*, 2023.

[35] Oesterling, Alex and Ma, Jiaqi and Calmon, Flavio P and Lakkaraju, Hima. FairSISA: Ensemble Post-Processing to Improve Fairness of Unlearning in LLMs. *arXiv preprint arXiv:2307.14754*, 2023.

[36] Nguyen, Thanh Tam and Huynh, Thanh Trung and Nguyen, Phi Le and Liew, Alan Wee-Chung and Yin, Hongzhi and Nguyen, Quoc Viet Hung. A Survey of Machine Unlearning. *arXiv preprint arXiv:2209.02299*, 2022.

[37] Liu, Ziyao and Ye, Huanyi and Chen, Chen and Lam, Kwok-Yan. Threats, Attacks, and Defenses in Machine Unlearning. *arXiv preprint arXiv:2403.13682*, 2024.

[38] Qu, Youyang and Yuan, Xin and Ding, Ming and Ni, Wei and Rakotoarivelo, Thierry and Smith, David. Learn to Unlearn: A Survey on Machine Unlearning. *arXiv preprint arXiv:2305.07512*, 2023.

[39] Puterman, Martin L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[40] Shannon, Claude Elwood. A Mathematical Theory of Communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[41] Huffman, David A. A Method for the Construction of Minimum-Redundancy Codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[42] Delétang, Grégoire and Ruoss, Anian and Duquenne, Paul-Ambroise and Catt, Elliot and Genewein, Tim and Mattern, Christopher and Grau-Moya, Jordi and Wenliang, Li Kevin and Aitchison, Matthew and Orseau, Laurent and others. Language Modeling Is Compression. *arXiv preprint arXiv:2309.10668*, 2023.

[43] Huang, Yuzhen and Zhang, Jinghan and Shan, Zifei and He, Junxian. Compression Represents Intelligence Linearly. *arXiv preprint arXiv:2404.09937*, 2024.

[44] Hutter, Marcus. *Universal Artificial Intelligence: Sequential Decisions Based On Algorithmic Probability*. Springer Science & Business Media, 2005.

[45] Mingard, Chris and Valle-Pérez, Guillermo and Skalse, Joar and Louis, Ard A. Is SGD a Bayesian sampler? Well, almost. *Journal of Machine Learning Research*, 22(79):1–64, 2021.

[46] Zipf, George Kingsley. The Psychology of Language. In *Encyclopedia of psychology*, pages 332–341. Philosophical Library, 1946.

[47] Hartmann, Jochen and Heitmann, Mark and Siebert, Christian and Schamp, Christina. More than a feeling: Accuracy and application of sentiment analysis. *International Journal of Research in Marketing*, 40(1):75–87, 2023.

[48] Taori, Rohan and Gulrajani, Ishaan and Zhang, Tianyi and Dubois, Yann and Li, Xuechen and Guestrin, Carlos and Liang, Percy and Hashimoto, Tatsunori B. Stanford Alpaca: An Instruction-following Llama Model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

[49] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, 2022.

[50] Askell, Amanda and Bai, Yuntao and Chen, Anna and Drain, Dawn and Ganguli, Deep and Henighan, Tom and Jones, Andy and Joseph, Nicholas and Mann, Ben and DasSarma, Nova and others. A General Language Assistant as a Laboratory for Alignment. *arXiv preprint arXiv:2112.00861*, 2021.

[51] AI@Meta. Llama 3 Model Card. `https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md`, 2024.

A Related Work

The Fragility of LLMs Alignment. Pre-trained LLMs often generate offensive content [6]. Recent initiatives [9, 3] have aimed to align these models to minimize harmful outputs [3, 29]. However, studies show that even well-aligned models can be compromised easily, and fine-tuning them on non-malicious datasets might unintentionally impair their safety mechanisms [15, 17, 30]. Why is alignment so fragile? [31] pinpoint areas essential for safety guardrails distinct from utility-relevant regions, achieving this separation at both neuron and rank levels through weight attribution.

Machine Unlearning. The necessity for Machine Unlearning (MU) stems from the requirement for adaptive learning systems to erase user privacy data and any derived lineage data [32]. This technique is employed in data deletion [33], enhances the fairness of models [34, 35], and prevents generative models from creating harmful content [4]. Current MU methods can be categorized into two families: exact MU and approximate MU [36, 37]. Exact MU seeks to eliminate the influence of specific data points entirely through comprehensive retraining, offering provable error guarantees for data removal. Conversely, Approximate MU aims to reduce data point influence through efficient parameter updates, trading off complete erasure for lower computational demands [38].

B Background

B.1 Large Language Models

We consider an LLM parameterized by $\boldsymbol{\theta}$ and denoted by the output distribution $p_{\boldsymbol{\theta}}(\cdot|\cdot)$. The generation process of the LLM can be defined by $(\mathcal{X}, \mathcal{Y}, \mathcal{V}, \mathcal{L}, p_{\boldsymbol{\theta}})$. The input space (prompt space) is $\mathcal{X} \in \sum^{\leq l_{\text{max}}}$, and the output space (response space) is $\mathcal{Y} \in \sum^{\leq l_{\text{max}}}$ for some constant l_{max}. The model takes a sequence $\boldsymbol{x} = (x_0, \ldots, x_{n-1})$ as input to generate a corresponding output $\boldsymbol{y} = (y_0, \ldots, y_{m-1})$, where x_i and y_j represent the individual tokens from a predetermined vocabulary Σ.

The autoregressive LLM $p_{\boldsymbol{\theta}}$ generates tokens sequentially for a given position, relying solely on the previously generated tokens sequence. Consequently, this model can be conceptualized as a markov decision process [39], wherein the conditional probability $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ can be defined through a decomposition as follows,

$$p_{\boldsymbol{\theta}}\left(y_{0..k-1}\middle|\boldsymbol{x}\right) = \prod_{0\leq k\leq m} p_{\boldsymbol{\theta}}\left(y_k\middle|\boldsymbol{x}, y_{0..k-1}\right).$$

Pre-training. During pre-training, an LLM acquires foundational language comprehension and reasoning abilities by processing vast quantities of unstructured text. The pre-train loss is defined as follows:

$$\mathcal{L}_{\text{PT}}(\boldsymbol{\theta}; \mathcal{D}_{\text{PT}}) = -\mathbb{E}_{(\boldsymbol{x}, x_N)\sim\mathcal{D}_{\text{PT}}}\left[\log p_{\boldsymbol{\theta}}\left(x_N\middle|\boldsymbol{x}\right)\right].$$

where $\boldsymbol{x} = (x_0, \cdots, x_{N-1})$ and $N \in \mathbb{N}$, such that $(x_0, \cdots, x_N)$ forms a prefix in some piece of pre-training text.

Supervised Fine-tuning (SFT). This phase adjusts the pre-trained models to follow specific instructions, utilizing a smaller dataset compared to the pre-training corpus to ensure mode alignment with target tasks. For a given SFT dataset $\mathcal{D}_{\text{SFT}} = \left\{\left(\boldsymbol{x}^i, \boldsymbol{y}^i\right)\right\}_{i=1}^{N}$ which is sampled from a high-quality distribution, SFT aims to minimize the negative log-likelihood loss:

$$\mathcal{L}_{\text{SFT}}(\boldsymbol{\theta}; \mathcal{D}_{\text{SFT}}) = -\mathbb{E}_{(\boldsymbol{x}, \mathbf{y})\sim\mathcal{D}_{\text{SFT}}}\left[\log p_{\boldsymbol{\theta}}\left(\boldsymbol{y}\middle|\boldsymbol{x}\right)\right].$$

Given that $\mathbb{E}_{(\boldsymbol{x}, \mathbf{y})\sim\mathcal{D}_{\text{SFT}}}\left[\log p_{\mathcal{D}}\left(\boldsymbol{y}\middle|\boldsymbol{x}\right)\right]$ is fixed when specifying $\mathcal{D}_{\text{SFT}}$, the optimization objective $\mathcal{L}_{\text{SFT}}$ becomes the Kullback-Leibler (KL) divergence between the model $p_{\boldsymbol{\theta}}$ and the SFT distribution.

B.2 Compression Theory

Lossless Compression. The goal of lossless compression is to find a compression protocol that encodes a given dataset $\mathcal{D}$ and its distribution $\mathcal{P}_{\mathcal{D}}$ with the smallest possible expected length, and allows for a decoding scheme that can perfectly reconstruct the original dataset $\mathcal{D}$ from the compressed data. According to Shannon's source coding theorem [40], for a random variable takes value from $\mathcal{D}$ and follows $\mathcal{P}_{\mathcal{D}}$, the expected code length $\mathcal{L}$ of any lossless compression protocol satisfies

$$\mathcal{L} \geq H\left(\mathcal{P}_{\mathcal{D}}\right).$$

where $H\left(\mathcal{P}_{\mathcal{D}}\right)$ stands for the Shannon entropy of $\mathcal{P}_{\mathcal{D}}$. Huffman code [41] is a typical type of optimal code for lossless compression. For a random variable follows $\mathcal{P}_{\mathcal{D}}$, the expected code length $\mathcal{L}$ satisfies

$$H\left(\mathcal{P}_{\mathcal{D}}\right) \leq \mathcal{L} \leq H\left(\mathcal{P}_{\mathcal{D}}\right) + 1.$$

Compression and Prediction. The relationship between data compression and prediction is tightly interconnected. Consider a model $p_{\boldsymbol{\theta}}$ and $\boldsymbol{x} = (x_0, \cdots, x_{m-1})$ derived from a dataset $\mathcal{D}$. Under arithmetic coding [42, 43], the optimal expected code length $\mathcal{L}$ is given by:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\sum_{0\leq k\leq m} -\log_2 p_{\boldsymbol{\theta}}\left(x_i \middle| x_{0,\cdots,k-1}\right)\right].$$

This aligns with the cross-entropy loss used when training $p_{\boldsymbol{\theta}}$, which suggests a certain consistency between compression and prediction. Hutter [44] provides a more detailed explanation of the equivalence between compressing optimal and predicting optimal. Experimentally, further evidence has been provided to demonstrate the equivalence between large language model prediction and compression [42] and establish that compression performance represents intelligence linearly [43].

C Assumptions and Proofs

Definition C.1 (*Inverse Alignment*)**.** Given an initial language model $p_{\boldsymbol{\theta}_0}$, for any ϵ, after aligning it on dataset $\mathcal{D}_a$ to obtain the aligned model $p_{\boldsymbol{\theta}_1}$, we use dataset $\mathcal{D}_b$ (where $|\mathcal{D}_b| \ll |\mathcal{D}_a|$) to perform an operation on $p_{\boldsymbol{\theta}_1}$. This process yields an *inverse-aligned model* $p_{\boldsymbol{\theta}_0'}$, such that $\rho(p_{\boldsymbol{\theta}_0'}, p_{\boldsymbol{\theta}_0}) \leq \epsilon$ for a given metric function ρ (which can be viewed as a measure of behavioral and distributional proximity between two models). We define the transition from $p_{\boldsymbol{\theta}_1}$ back to $p_{\boldsymbol{\theta}_0}$ as *inverse alignment*.

Remark C.2 (Pruning of the TRT). For a pruned node $\mathcal{S}$ of a TRT, the pruning operation is as follows: remove the pruned node and all its children, then add the probability of the node S to its parent's `EOS` node. The pruning operation decreases the depth and the number of nodes in the TRT while the sum of the probability of all `EOS` nodes remains constant at 1.

Remark C.3 ($\boldsymbol{z}$'s meaning). Since model compression for datasets involves both pre-training and SFT processes, $\boldsymbol{z}$ represents different meanings in these two processes. During pre-training, $\boldsymbol{z}$ represents a complete sequence of text segments; whereas during SFT, $\boldsymbol{z}$ represents a complete sequence of questions and corresponding answers in the dataset.

Assumption C.4 (Scale of $\mathcal{T}$ is Monotone with Model Size)**.** Consider a parameterized model $p_{\boldsymbol{\theta}}(\cdot)$, the dataset $\mathcal{D}$ and $\mathcal{D}$'s TRT $\mathcal{T}_{\mathcal{D}}$. Due to the finite size of model parameters $\boldsymbol{\theta}$, the model $p_{\boldsymbol{\theta}}$ can only represent a limited portion of $\mathcal{D}$, which corresponds to a finite pruning of the tree $\mathcal{T}_{\mathcal{D}}$. Let's assume that the depth of the pruned tree $\mathcal{T}_{\mathcal{D}}'$ is monotonically increasing with the size of $\boldsymbol{\theta}$.

Theorem C.5 (Ideal Code Length with Finite-parameter Model)**.** *Consider a finite parameter model $p_{\boldsymbol{\theta}}(\cdot)$ which is training on dataset $\mathcal{D}$, the ideal code length $\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})$ of a random response $\boldsymbol{x}$ compressed by $p_{\boldsymbol{\theta}}$ can be expressed as follows,*

$$\mathbb{E}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})\right] = \left\lceil \frac{|\boldsymbol{x}|}{d} \right\rceil \left\lceil -\sum_{i=1}^{d}\sum_{j=1}^{2^{i-1}} p_{ij}\log p_{ij} \right\rceil \tag{3}$$

where d represents the depth of the $\mathcal{T}_{\mathcal{D}}$ after pruning under Definition 2.4 protocol, p_{ij} represents the probability values of the `EOS` nodes for the j-th node at the i-th layer.

Proof. When $\left|\boldsymbol{x}\right| \leq d$, the compression protocol defined in Definition 2.4 can perfectly compress $\boldsymbol{x}$. Hence, the expectation of the ideal code length $\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})$ satisfies:

$$\mathbb{E}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})\right] = \left\lceil -\sum_{i=1}^{d}\sum_{j=1}^{2^{i-1}} p_{ij}\log p_{ij} \right\rceil \tag{4}$$

where d represents the depth of the pruned tree $\mathcal{T}'_{\mathcal{D}}$ and p_{ij} represents the probability values of the `EOS` nodes for the j-th node at the i-th layer.

Now consider $sd \leq |\boldsymbol{x}| \leq (s+1)d$. Let us suppose that $\boldsymbol{x} = (\boldsymbol{x}_1 \cdots \boldsymbol{x}_s \boldsymbol{x}_{s+1})$, where $|\boldsymbol{x}_k| = d$, for $k \in \{1, \dots, s\}$ and $|\boldsymbol{x}_{s+1}| \leq d$. In this case, $\boldsymbol{x}$ cannot be perfectly compressed by the model. Hence, the compression of x needs to be performed in segments, and the length of each segment is not greater than d.

$$
\begin{aligned}
\mathbb{E}_{\boldsymbol{x}}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})\right] &= \mathbb{E}_{\boldsymbol{x}_1}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x}_1)\right] + \mathbb{E}_{\boldsymbol{x}_1}\mathbb{E}_{(\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}((\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1}))\,\middle|\,\boldsymbol{x}_1\right] & (5)\\
&= \mathbb{E}_{\boldsymbol{x}_1}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x}_1)\right] + \sum_{\boldsymbol{x}_1} p(\boldsymbol{x}_1) \sum_{(\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})} p(\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1}|\boldsymbol{x}_1)\cdot\mathcal{L}_{p_{\boldsymbol{\theta}}}((\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})) & (6)\\
&= \mathbb{E}_{\boldsymbol{x}_1}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x}_1)\right] + \sum_{(\boldsymbol{x}_1\dots\boldsymbol{x}_{s+1})} \frac{p(\boldsymbol{x}_1\dots\boldsymbol{x}_{s+1})\cdot p(\boldsymbol{x}_1)}{p(\boldsymbol{x}_1)}\cdot\mathcal{L}_{p_{\boldsymbol{\theta}}}((\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})) & (7)\\
&= \mathbb{E}_{\boldsymbol{x}_1}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x}_1)\right] + \sum_{(\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})} p(\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})\cdot\mathcal{L}_{p_{\boldsymbol{\theta}}}((\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})) & (8)\\
&= \mathbb{E}_{\boldsymbol{x}_1}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x}_1)\right] + \mathbb{E}_{(\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1})}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}((\boldsymbol{x}_2\dots\boldsymbol{x}_{s+1}))\right] & (9)\\
&= \sum_{k=1}^{s+1}\mathbb{E}_{\boldsymbol{x}_k}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x}_k)\right] & (10)\\
&= \left\lceil\frac{|\boldsymbol{x}|}{d}\right\rceil\left[-\sum_{i=1}^{d}\sum_{j=1}^{2^{i-1}} p_{ij}\log p_{ij}\right] & (11)
\end{aligned}
$$

thus the proof is completed. □

Definition C.6 (Compression Rate of Finite-parameter Model)**.** Consider a finite parameter model $p_{\boldsymbol{\theta}}(\cdot)$ which is training on dataset $\mathcal{D}$ that follows the distribution $p_{\mathcal{D}}$, the reciprocal of the data compression rate $\gamma_{p_{\boldsymbol{\theta}}}$ of the model $p_{\boldsymbol{\theta}}$ is defined as follows,

$$
\begin{aligned}
\gamma_{p_{\boldsymbol{\theta}}} &= \mathbb{E}_{\boldsymbol{x}}\left[\frac{\mathbb{E}_{\boldsymbol{x}}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}(\boldsymbol{x})\right]}{|\boldsymbol{x}|}\right] & (12)\\
&= \Theta\left(-\frac{\sum_{i=1}^{d}\sum_{j=1}^{2^{i-1}} p_{ij}\log p_{ij}}{d}\right), & (13)
\end{aligned}
$$

where p_{ij}, d share the same definition as in Theorem 2.5.

Assumption C.7 (Leaf Node Probability Concentration)**.** Consider the TRT of dataset D. Due to the high proportion of long texts in the dataset , we assume that the probability density of the pruned tree $\mathcal{T}'_{\mathcal{D}}$ with depth d is concentrated at the leaf nodes. Let the `EOS` token probabilities at the leaf nodes of $\mathcal{T}'_{\mathcal{D}}$ be denoted as $p_1, \cdots, p_{2^{d-1}}$, the assumption implies that,

$$
\sum_{i=1}^{2^{d-1}} p_i = 1 \tag{14}
$$

Theorem C.8 (The Validity of Assumption 2.2)**.** *Considering a token table with k different tokens, where $k > 2$. In the sense of compression, the average encoding length $\mathbb{E}[\mathcal{L}_{k,p_{\boldsymbol{\theta}}}(\boldsymbol{x})]$ for a random response x has the following relationship with the average encoding length $\mathbb{E}[\mathcal{L}_{2,p_{\boldsymbol{\theta}}}(\boldsymbol{x})]$ of a token table with only binary tokens.*

$$
\mathbb{E}[\mathcal{L}_{k,p_{\boldsymbol{\theta}}}(\boldsymbol{x})] = \Theta\left(\frac{\mathbb{E}[\mathcal{L}_{2,p_{\boldsymbol{\theta}}}(\boldsymbol{x})]}{\log_2 k}\right) \tag{15}
$$

Proof. When compressing a random response $\boldsymbol{x}$ using a token table with k tokens, consider the following alternative: Suppose the k tokens are $0, 1, \cdots, k-1$. Write these k tokens in binary and replace the original k tokens' ToT (Definition 2.3) with a ToT that uses only binary tokens. We now compute the difference in average encoding length between these two setups:

$$
\begin{aligned}
\mathbb{E}[\mathcal{L}_{k,p_{\boldsymbol{\theta}}}(\boldsymbol{x})] &= \Theta\left(\left\lceil\frac{|\boldsymbol{x}|}{d}\right\rceil\left\lceil-\sum_{j=1}^{k^{d-1}} p_{ij}\log p_{ij}\right\rceil\right) && (16)\\
&= \Theta\left(\frac{|vx|\left(-\sum_{j=1}^{2^{\lceil\log_2 k\rceil d-1}} p_{ij}\log p_{ij}\right)}{d}\right) \quad (Due\ to\ Assumption\ C.4) && (17)\\
&= \Theta\left(\frac{|\boldsymbol{x}|\left(-\sum_{j=1}^{2^{\lceil\log_2 k\rceil d-1}} p_{ij}\log p_{ij}\right)}{\lceil\log_2 k\rceil * \frac{d}{\lceil\log_2 k\rceil}}\right) && (18)\\
&= \Theta\left(\left\lceil\frac{|\boldsymbol{x}|}{\frac{d}{\lceil\log_2 k\rceil}}\right\rceil\frac{\left\lceil-\sum_{j=1}^{2^{\lceil\log_2 k\rceil d-1}} p_{ij}\log p_{ij}\right\rceil}{\log_2 k}\right) && (19)\\
&= \Theta\left(\frac{\mathbb{E}[\mathcal{L}_{2,p_{\boldsymbol{\theta}}}(\boldsymbol{x})]}{\log_2 k}\right) && (20)
\end{aligned}
$$

where d represents the depth of the original ToT. □

Definition C.9 (Joint Compression of Multiple Datasets)**.** Consider using a finite parameter model to compress N pairwise disjoint datasets $\mathcal{D}_1, \cdots, \mathcal{D}_N$. For the TRT of the jointly compressed dataset $\mathcal{D} = \bigcup_{k=1}^{N}\mathcal{D}_k$, in the context of joint compression, the node weights relate to the node weights of each compressed dataset $\mathcal{D}_k$ as follows.

$$
p_i^{\mathcal{D}} = \frac{\sum_{k=1}^{N} p_i^{\mathcal{D}_k}|\mathcal{D}_k|}{\sum_{k=1}^{N}|\mathcal{D}_k|}, \qquad (21)
$$

where $p_i^{\mathcal{D}}$ stands for the probability value for the node in $\mathcal{T}_{\mathcal{D}}$ while $p_i^{\mathcal{D}_k}$ stands for the probability value for the node in $\mathcal{T}_{\mathcal{D}_k}$. The finite parameter joint compression process for $\mathcal{D}_1, \cdots, \mathcal{D}_N$ is essentially the process of compressing $\mathcal{D}$ according to Definition 2.4.

Definition C.10 (Compression Rate for Specific Dataset)**.** For N pairwise disjoint datasets $\mathcal{D}_1, \cdots, \mathcal{D}_N$ and a finite parameter model $p_{\boldsymbol{\theta}}$ compressing $\mathcal{D} = \bigcup_{k=1}^{N}\mathcal{D}_k$, the reciprocal of the compression rate $\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_i}$ for a particular dataset $\mathcal{D}_k$ is defined as follows.

$$
\begin{aligned}
\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_i} &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_i}\left[\frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_i}\left[\mathcal{L}_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_i}(\boldsymbol{x})\right]}{|\boldsymbol{x}|}\right] && (22)\\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_i}\left[\frac{1}{|\boldsymbol{x}|}\left\lceil\frac{|\boldsymbol{x}|}{d}\right\rceil\left\lceil-\sum_{i=1}^{d}\sum_{j=1}^{2^{i-1}} p_{ij}^{\mathcal{D}_k}\log p_{ij}^{\mathcal{D}}\right\rceil\right] && (23)\\
&= \Theta\left(-\frac{\sum_{i=1}^{d}\sum_{j=1}^{2^{i-1}} p_{ij}^{\mathcal{D}_k}\log p_{ij}^{\mathcal{D}}}{d}\right), && (24)
\end{aligned}
$$

where $p_{ij}^{\mathcal{D}_k}$ represents the probability values of the `EOS` nodes for the j-th node at the i-th layer in $\mathcal{T}_k'$.

Definition C.11 (Mass Distribution in TRT)**.** Consider the sample space Ω consisting of all responses in dataset $\mathcal{D}$. The probability distribution $\mathcal{P}_{\mathcal{D}}$ of all subtrees at the d-th level nodes of $\mathcal{T}_{\mathcal{D}}$ is a mapping from Ω to $[0, 1]$. Let $X_{\mathcal{D}}$ be the random variable representing the probability value taken at each leaf. The mass distribution P_{mass} represents the probability that $X_{\mathcal{D}}$ takes the corresponding probability value. According to the definition of P_{mass}, $\mathbb{E}[X_{\mathcal{D}}] = 1$

Remark C.12 (Mixture of Mass Distribution). For independently and differently distributed datasets $\mathcal{D}_1, \ldots, \mathcal{D}_N$, $\mathcal{D} = \bigcup_{i=1}^{N}\mathcal{D}_i$ is a mixture of these datasets. According to Definition C.9, for the pruned trees $\mathcal{T}_1, \ldots, \mathcal{T}_N$ of these datasets with depth d, the random variables of their leaf nodes satisfy the following relationship:

$$X_{\mathcal{D}} = \frac{\sum_{k=1}^{N} |\mathcal{D}_k| X_{\mathcal{D}_k}}{\sum_{k=1}^{N} |\mathcal{D}_k|} \tag{25}$$

where $X_{\mathcal{D}_k}$ follows the mass distribution $\mathcal{P}^k_{mass}$. For $X_{\mathcal{D}_{k_1}}$ and $X_{\mathcal{D}_{k_2}}$ from different datasets, $X_{\mathcal{D}_{k_1}}$ and $X_{\mathcal{D}_{k_2}}$ are independent of each other.

Lemma C.13 (Entropy of Mass Distribution)**.** *Consider the pruned trees $\mathcal{T}'$ and $\mathcal{T}'_k$ of dataset $\mathcal{D}$ and $\mathcal{D} = \bigcup_{i=1}^{N} \mathcal{D}_i$, both with depth d. Denote that the response distribution of the leaf nodes of $\mathcal{T}'$ is $\mathcal{P}^{\mathcal{D}}$, and the mass distribution is $\mathcal{P}_{mass}$. Similarly, the response distribution of the leaf nodes of $\mathcal{T}'_k$ is $\mathcal{P}^{\mathcal{D}}_k$, and the mass distribution is $\mathcal{P}^k_{mass}$. When d is sufficiently large, the Shannon entropy of the response distribution can be rewritten as follows.*

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_k}\left[-p^{\mathcal{D}_k}\log p^{\mathcal{D}}\right] = \mathbb{E}_{X_{\mathcal{D}_k}\sim\mathcal{P}^k_{mass}, X_{\mathcal{D}}\sim\mathcal{P}_{mass}}\left[-X_{\mathcal{D}_k}\log X_{\mathcal{D}}\right] + \log 2^{d-1} \tag{26}$$

where $p^{\mathcal{D}}$, $p^{\mathcal{D}_k}$ stand for the probability of the leaf nodes of $\mathcal{T}'$,$\mathcal{T}'_k$ while $X_{\mathcal{D}_k}$, $X_{\mathcal{D}}$ stand for the random variables of the probability of the leaf nodes of $\mathcal{T}'$,$\mathcal{T}'_k$.

Proof. Let $M = 2^{d-1}$ be the number of leaf nodes of $\mathcal{T}'$ with depth d. According to the definitions of the response distribution $\mathcal{P}$ and mass distribution $\mathcal{P}_{mass}$, we have $Mp^{\mathcal{D}_j} = X_{\mathcal{D}_j}, \forall j \in \{1, \ldots, N\}$. Therefore,

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_k}\left[-p^{\mathcal{D}_k}\log p^{\mathcal{D}}\right] = \sum_{i=1}^{M} -p_i^{\mathcal{D}_k}\log p_i^{\mathcal{D}} \tag{27}$$

$$= \sum_{i=1}^{M} -\frac{X_{i,\mathcal{D}_k}}{M}\log\frac{X_{i,\mathcal{D}}}{M} \tag{28}$$

$$= \sum_{i=1}^{M} -\frac{1}{M}X_{i,\mathcal{D}_k}\log X_{i,\mathcal{D}_k} + \log M \tag{29}$$

$$= \mathbb{E}_{X_{\mathcal{D}_k}\sim\mathcal{P}^k_{mass}, X_{\mathcal{D}}\sim\mathcal{P}_{mass}}\left[-X_{\mathcal{D}_k}\log X_{\mathcal{D}}\right] + \log 2^{d-1} \tag{30}$$

□

Remark C.14. In Lemma C.13, $X_{\mathcal{D}_k}$ are assumed to be independent. However due to $\sum_{i=1}^{M} p_i^{\mathcal{D}_k} = 1$, the $X_{\mathcal{D}_k}$ are not actually independent. Considering that d is sufficiently large in our subsequent analysis, we can regard the independence of $X_{\mathcal{D}_k}$ as a good approximation.

Assumption C.15 (Introduction of Pareto Distribution)**.** We assume that the mass distribution of the segment follows a heavy-tailed Pareto distribution, with supporting evidence from [45, 46]. In this paper, we assume that the mass distribution of the pruned trees $\mathcal{T}'$ of the same depth d from different datasets follows a Pareto distribution with the same parameters.

$$p_X(x) = \begin{cases} \frac{\alpha C^{\alpha}}{x^{\alpha+1}} & x \geq C, \\ 0 & x < C. \end{cases} \tag{31}$$

where α, C are parameters of the Pareto distribution. Here we assume that α is sufficiently large due to the lighter heavy-tailed nature of the mass distribution.

Definition C.16 (Normalized Compression Rate)**.** For N pairwise disjoint datasets $\mathcal{D}_1, \cdots, \mathcal{D}_N$ and a finite parameter model $p_{\boldsymbol{\theta}}$ compressing $\mathcal{D} = \bigcup_{k=1}^{N} \mathcal{D}_k$, the normalized reciprocal of the compression rate $\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_k/\mathcal{D}}$ for a particular dataset $\mathcal{D}_k$ is defined as:

$$\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_k/\mathcal{D}} = \frac{\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_k} - \frac{1}{d}\log 2^{d-1}}{\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}} - \frac{1}{d}\log 2^{d-1}}, \tag{32}$$

where d is the depth of the pruned tree $\mathcal{T}'_k$ of dataset $\mathcal{D}_k$. Here, $\frac{1}{d}\log 2^{d-1}$ is the reciprocal of the compression rate for a uniform distribution while $\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}}$ represents the reciprocal of the compression rate for $\mathcal{D}$.

Theorem C.17 (Elasticity of Language Models)**.** *Consider datasets $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ each with a Pareto mass distribution (Assumption C.15), and the model $p_{\boldsymbol{\theta}}(\cdot)$ trained on $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \mathcal{D}_3$. When dataset $\mathcal{D}_3$'s data volume $|\mathcal{D}_3|$ changes, the normalized reciprocal of the compression ratio $\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}$, $\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}$ of the model for $\mathcal{D}_1$ and $\mathcal{D}_2$ satisfies:*

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l} = \Theta\left(k\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l}\right) \tag{33}$$

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l} > 0 \tag{34}$$

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l} > 0 \tag{35}$$

where $l = \frac{|\mathcal{D}_3|}{|\mathcal{D}_2|} \ll 1$, $k = \frac{|\mathcal{D}_1|}{|\mathcal{D}_2|} \gg 1$.

Proof. For the sake of convenience in calculations, we first use Lemma C.13 to replace the Shannon entropy of response distribution.

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_j/\mathcal{D}}}{d\,l} = \frac{d\left(\frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_j}\left[-p^{\mathcal{D}_j}\log p^{\mathcal{D}}\right]-\log 2^{d-1}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}}\left[-p^{\mathcal{D}}\log p^{\mathcal{D}}\right]-\log 2^{d-1}}\right)}{d\,l} \tag{36}$$

$$= \frac{d\left(\frac{\mathbb{E}_{X_{\mathcal{D}_j}\sim\mathcal{P}^j_{mass},X_{\mathcal{D}}\sim\mathcal{P}_{mass}}\left[-X_{\mathcal{D}_j}\log X_{\mathcal{D}}\right]}{\mathbb{E}_{X_{\mathcal{D}}\sim\mathcal{P}_{mass},X_{\mathcal{D}}\sim\mathcal{P}_{mass}}\left[-X_{\mathcal{D}}\log X_{\mathcal{D}}\right]}\right)}{d\,l}. \tag{37}$$

According to Assumption C.15, X_{D_j} follows a Pareto distribution with the same parameters α and c. Hence,

$$\mathbb{E}_{X_{\mathcal{D}_j}\sim\mathcal{P}^j_{mass},X_{\mathcal{D}}\sim\mathcal{P}_{mass}}\left[-X_{\mathcal{D}_j}\log X_{\mathcal{D}}\right] \tag{38}$$

$$= -\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{\alpha^3 c^{3\alpha} x_j}{\prod_{i=1}^3 x_i^{\alpha+1}}\log\frac{\sum_{i=1}^3 |\mathcal{D}_i| x_i}{\sum_{i=1}^3 |\mathcal{D}_i|}dx_1dx_2dx_3 \tag{39}$$

$$= -\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{\alpha^3 c^{3\alpha} x_j}{\prod_{i=1}^3 x_i^{\alpha+1}}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3 \tag{40}$$

$$\mathbb{E}_{X_{\mathcal{D}}\sim\mathcal{P}'_{mass}X_{\mathcal{D}}\sim\mathcal{P}_{mass}}\left[-X_{\mathcal{D}}\log X_{\mathcal{D}}\right] \tag{41}$$

$$= -\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{\alpha^3 c^{3\alpha}}{\prod_{i=1}^3 x_i^{\alpha+1}}\cdot\frac{kx_1+x_2+lx_3}{k+l+1}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3, \tag{42}$$

where $j = 1, 2, 3$. Therefore, $\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l}$ and $\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l}$ can be written as:

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l} = \frac{\frac{dS_1}{d\,l}H - \frac{dH}{d\,l}S_1}{H^2} \tag{43}$$

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l} = \frac{\frac{dS_2}{d\,l}H - \frac{dH}{d\,l}S_2}{H^2}, \tag{44}$$

where

$$H = \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{kx_1+x_2+lx_3}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+l+1)}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3 \tag{45}$$

$$S_1 = \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3 \tag{46}$$

$$S_2 = \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3. \tag{47}$$

Proving that $\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l} = \Theta\left(k\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l}\right)$ is equivalent to proving:

$$\lim_{k\to+\infty,\, l\to 0} \frac{k\cdot\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l}}{\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l}} = C \tag{48}$$

where C is a constant. By substituting (43) and (44) into (48), we have

$$\lim_{k\to+\infty,\, l\to 0} \frac{k\cdot\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l}}{\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l}} = \frac{\lim_{k\to+\infty,\, l\to 0} k\left(\frac{dS_1}{d\,l}H - \frac{dH}{d\,l}S_1\right)}{\lim_{k\to+\infty,\, l\to 0} \frac{dS_2}{d\,l}H - \frac{dH}{d\,l}S_2} \tag{49}$$

Now calculate the values of S_1, S_2, H and $\frac{dS_1}{d\,l}$, $\frac{dS_2}{d\,l}$, $\frac{dH}{d\,l}$ respectively in the case of $k\to+\infty$, $l\to 0$.

$$\lim_{k\to+\infty,\, l\to 0} S_1$$

$$= \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}} \log\frac{kx_1+x_2+lx_3}{k+l+1} dx_1dx_2dx_3 \tag{50}$$

$$= \int_c^{+\infty}\int_c^{+\infty} \lim_{k\to+\infty,\, l\to 0} \int_c^{\delta(kx_1+x_2)} \frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}} \log\frac{kx_1+x_2+lx_3}{k+l+1} dx_3dx_1dx_2 \tag{51}$$

$$= \int_c^{+\infty}\int_c^{+\infty} \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty} \frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}} \log\frac{\theta_1(kx_1+x_2)}{k+l+1} dx_1dx_2dx_3 \tag{52}$$

where $\theta_1 \in (1, 1+\delta)$

$$= \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}} \log\frac{\theta_1(kx_1+x_2)}{k+1} dx_1dx_2dx_3 \tag{53}$$

$$\lim_{k\to+\infty,\, l\to 0} S_2$$

$$= \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}} \log\frac{kx_1+x_2+lx_3}{k+l+1} dx_1dx_2dx_3 \tag{54}$$

$$= \int_c^{+\infty}\int_c^{+\infty} \lim_{k\to+\infty,\, l\to 0} \int_c^{\delta(kx_1+x_2)} \frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}} \log\frac{kx_1+x_2+lx_3}{k+l+1} dx_3dx_1dx_2 \tag{55}$$

$$= \int_c^{+\infty}\int_c^{+\infty} \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty} \frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}} \log\frac{\theta_2(kx_1+x_2)}{k+l+1} dx_3dx_1dx_2 \tag{56}$$

where $\theta_2 \in (1, 1+\delta)$

$$= \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}} \log\frac{\theta_2(kx_1+x_2)}{k+1} dx_1dx_2dx_3 \tag{57}$$

$$\lim_{k\to+\infty,\, l\to 0} H$$

$$= \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{kx_1+x_2+lx_3}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+l+1)} \log\frac{kx_1+x_2+lx_3}{k+l+1} dx_1dx_2dx_3 \tag{58}$$

$$= \int_c^{+\infty}\int_c^{+\infty} \lim_{k\to+\infty,\, l\to 0} \int_c^{\delta(kx_1+x_2)} \frac{kx_1+x_2+lx_3}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+l+1)} \log\frac{kx_1+x_2+lx_3}{k+l+1} dx_3dx_1dx_2 \tag{59}$$

$$= \int_c^{+\infty}\int_c^{+\infty} \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty} \frac{\theta(kx_1+x_2)}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+1)} \log\frac{\theta(kx_1+x_2)}{k+l+1} dx_3dx_1dx_2 \tag{60}$$

where $\theta \in (1, 1+\delta)$

$$= \lim_{k\to+\infty,\, l\to 0} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{\theta(kx_1+x_2)}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+1)} \log\frac{\theta(kx_1+x_2)}{k+1} dx_1dx_2dx_3 \tag{61}$$

$$\lim_{k\to+\infty,\,l\to 0}\frac{dS_1}{d\,l}$$

$$= \lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{(k+1)x_3-kx_1-x_2}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}(k+l+1)(kx_1+x_2+lx_3)}dx_1dx_2dx_3 \tag{62}$$

$$\begin{aligned}&= \int_c^{+\infty}\int_c^{+\infty}\lim_{k\to+\infty,\,l\to 0}\int_c^{\delta(kx_1+x_2)}\frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha}(kx_1+x_2+lx_3)}\\ &\quad -\lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}(k+1+l)}dx_1dx_2dx_3\end{aligned} \tag{63}$$

$$\begin{aligned}&= \int_c^{+\infty}\int_c^{+\infty}\lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha}\theta_1'(kx_1+x_2)}\\ &\quad -\lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}(k+1+l)}dx_1dx_2dx_3\end{aligned} \tag{64}$$

where $\theta_1' \in (1, 1+\delta)$

$$\begin{aligned}&= \lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha}\theta_1'(kx_1+x_2)}dx_1dx_2dx_3\\ &\quad -\lim_{k\to+\infty,\,l\to 0}\frac{1}{(k+1)\alpha^2(\alpha-1)c^{3\alpha-1}}\end{aligned} \tag{65}$$

$$\lim_{k\to+\infty,\,l\to 0}\frac{dS_2}{d\,l}$$

$$= \lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{(k+1)x_3-kx_1-x_2}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}(k+l+1)(kx_1+x_2+lx_3)}dx_1dx_2dx_3 \tag{66}$$

$$\begin{aligned}&= \int_c^{+\infty}\int_c^{+\infty}\lim_{k\to+\infty,\,l\to 0}\int_c^{\delta(kx_1+x_2)}\frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha}(kx_1+x_2+lx_3)}\\ &\quad -\lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}(k+1+l)}dx_1dx_2dx_3\end{aligned} \tag{67}$$

$$\begin{aligned}&= \int_c^{+\infty}\int_c^{+\infty}\lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha}\theta_2'(kx_1+x_2)}\\ &\quad -\lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}(k+1+l)}dx_1dx_2dx_3\end{aligned} \tag{68}$$

where $\theta_2' \in (1, 1+\delta)$

$$\begin{aligned}&= \lim_{k\to+\infty,\,l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha}\theta_2'(kx_1+x_2)}dx_1dx_2dx_3\\ &\quad -\lim_{k\to+\infty,\,l\to 0}\frac{1}{(k+1)\alpha^2(\alpha-1)c^{3\alpha-1}}\end{aligned} \tag{69}$$

$$
\begin{aligned}
&\lim_{k\to+\infty,\, l\to 0}\frac{dH}{d\,l}\\
=&\lim_{k\to+\infty,\, l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{(k+1)x_3-kx_1-x_2}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+l+1)^2}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3
\end{aligned}\tag{70}
$$

$$
\begin{aligned}
=&\lim_{k\to+\infty,\, l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha}(k+l+1)}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3\\
&-\lim_{k\to+\infty,\, l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{kx_1+x_2+lx_3}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+l+1)^2}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3
\end{aligned}\tag{71}
$$

$$
\begin{aligned}
=&\int_c^{+\infty}\int_c^{+\infty}\lim_{k\to+\infty,\, l\to 0}\int_c^{\delta(kx_1+x_2)}\frac{1}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha}(k+l+1)}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3\\
&-\int_c^{+\infty}\int_c^{+\infty}\lim_{k\to+\infty,\, l\to 0}\int_c^{\delta(kx_1+x_2)}\frac{kx_1+x_2+lx_3}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+l+1)^2}\log\frac{kx_1+x_2+lx_3}{k+l+1}dx_1dx_2dx_3
\end{aligned}\tag{72}
$$

$$
\begin{aligned}
=&\lim_{k\to+\infty,\, l\to 0}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha}(k+1)}\log\frac{\theta'(kx_1+x_2)}{k+1}dx_1dx_2dx_3\\
&-\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{\theta'(kx_1+x_2)}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+1)^2}\log\frac{\theta'(kx_1+x_2)}{k+1}dx_1dx_2dx_3\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{where } \theta'\in(1,1+\delta)
\end{aligned}\tag{73}
$$

Taking $\delta\to 0$, we have $\theta=\theta_1=\theta_2=\theta'=\theta_1'=\theta_2'=1$.

Therefore, the above formula can be simplified to:

$$\lim_{k\to+\infty,\, l\to 0}S_1\tag{74}$$

$$=\lim_{k\to+\infty}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}}\log\frac{kx_1+x_2}{k+1}dx_1dx_2dx_3\tag{75}$$

$$=\lim_{k\to+\infty}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{\log x_1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}}dx_1dx_2dx_3\tag{76}$$

$$=\frac{(\alpha-1)\log c+1}{\alpha^2(\alpha-1)^2c^{3\alpha-1}}\tag{77}$$

$$\lim_{k\to+\infty,\, l\to 0}S_2\tag{78}$$

$$=\lim_{k\to+\infty}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}}\log\frac{kx_1+x_2}{k+1}dx_1dx_2dx_3\tag{79}$$

$$=\lim_{k\to+\infty}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{\log x_1}{x_1^{\alpha+1}x_2^{\alpha}x_3^{\alpha+1}}dx_1dx_2dx_3\tag{80}$$

$$=\frac{\alpha\log c+1}{\alpha^3(\alpha-1)c^{3\alpha-1}}\tag{81}$$

$$\lim_{k\to+\infty,\, l\to 0}H\tag{82}$$

$$=\lim_{k\to+\infty}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{kx_1+x_2}{x_1^{\alpha+1}x_2^{\alpha+1}x_3^{\alpha+1}(k+1)}\log\frac{kx_1+x_2}{k+1}dx_1dx_2dx_3\tag{83}$$

$$=\lim_{k\to+\infty}\int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty}\frac{\log x_1}{x_1^{\alpha}x_2^{\alpha+1}x_3^{\alpha+1}}dx_1dx_2dx_3\tag{84}$$

$$=\frac{(\alpha-1)\log c+1}{\alpha^2(\alpha-1)^2c^{3\alpha-1}}\tag{85}$$

$$
\begin{aligned}
&\lim_{k\to+\infty,\, l\to 0} \frac{dS_1}{d\,l} && (86)\\
=&\lim_{k\to+\infty} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{1}{x_1^{\alpha} x_2^{\alpha+1} x_3^{\alpha}(kx_1+x_2)} dx_1 dx_2 dx_3 \\
&\quad - \lim_{k\to+\infty} \frac{1}{(k+1)\alpha^2(\alpha-1)c^{3\alpha-1}} && (87)\\
=&\frac{1}{k(k+1)\alpha^2(\alpha-1)c^{3\alpha-1}} && (88)
\end{aligned}
$$

$$
\begin{aligned}
&\lim_{k\to+\infty,\, l\to 0} \frac{dS_2}{d\,l} && (89)\\
=&\lim_{k\to+\infty} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{1}{x_1^{\alpha+1} x_2^{\alpha} x_3^{\alpha}(kx_1+x_2)} dx_1 dx_2 dx_3 \\
&\quad - \lim_{k\to+\infty} \frac{1}{(k+1)\alpha^2(\alpha-1)c^{3\alpha-1}} && (90)\\
=&\frac{1}{k\alpha^2(\alpha-1)^2(\alpha+1)c^{3\alpha-1}} && (91)
\end{aligned}
$$

$$
\begin{aligned}
&\lim_{k\to+\infty,\, l\to 0} \frac{dH}{d\,l} && (92)\\
=&\lim_{k\to+\infty} \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{1}{x_1^{\alpha+1} x_2^{\alpha+1} x_3^{\alpha}(k+1)} \log\frac{kx_1+x_2}{k+1} dx_1 dx_2 dx_3 \\
&- \int_c^{+\infty}\int_c^{+\infty}\int_c^{+\infty} \frac{(kx_1+x_2)}{x_1^{\alpha+1} x_2^{\alpha+1} x_3^{\alpha+1}(k+1)^2} \log\frac{kx_1+x_2}{k+1} dx_1 dx_2 dx_3 && (93)\\
=&\lim_{k\to+\infty} \frac{\alpha\log c-1}{(k+1)(\alpha-1)\alpha^3 c^{3\alpha-1}} - \frac{k((\alpha-1)\log c-1)}{(k+1)^2\alpha^2(\alpha-1)^2 c^{3\alpha-1}} && (94)\\
=&\frac{1}{(k+1)^2}\cdot\frac{\alpha\log c-1}{(\alpha-1)^2\alpha^2 c^{3\alpha-1}}, && (95)
\end{aligned}
$$

when α is sufficiently large. As a result, Equation 49 can be written as:

$$
\begin{aligned}
&\lim_{k\to+\infty,\, l\to 0} \frac{k\cdot\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l}}{\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l}} && (96)\\
=&\lim_{k\to+\infty} \frac{k\cdot\left(\frac{1}{k^2\alpha^2(\alpha-1)c^{3\alpha-1}} - \frac{\alpha\log c-1}{k^2\alpha^2(\alpha-1)^2c^{3\alpha-1}}\right)\cdot\frac{(\alpha-1)\log c+1}{\alpha^2(\alpha-1)^2c^{3\alpha-1}}}{\frac{1}{k\alpha^2(\alpha-1)^2(\alpha+1)c^{3\alpha-1}}\cdot\frac{(\alpha-1)\log c+1}{\alpha^2(\alpha-1)^2}c^{3\alpha-1}} && (97)\\
=&\alpha(\alpha+1)(1-\log c) && (98)
\end{aligned}
$$

where $\alpha(\alpha+1)(1-\log c)$ is a constant. Thus, the proof is completed.

Next, we prove that $\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l} > 0$,

$$
\begin{aligned}
&\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l} && (99)\\
=&\frac{1}{H^2}\cdot\left(\frac{dS_1}{d\,l}H - \frac{dH}{d\,l}S_1\right) && (100)\\
=&\frac{1}{H^2}\cdot\left(\frac{1}{k^2\alpha^2(\alpha-1)c^{3\alpha-1}} - \frac{\alpha\log c-1}{k^2\alpha^2(\alpha-1)^2c^{3\alpha-1}}\right)\cdot\frac{(\alpha-1)\log c+1}{\alpha^2(\alpha-1)^2c^{3\alpha-1}} && (101)\\
=&\frac{1}{H^2}\cdot\frac{\alpha(1-\log c)}{k^2\alpha^2(\alpha-1)^2c^{3\alpha-1}}\cdot\frac{(\alpha-1)\log c+1}{\alpha^2(\alpha-1)^2c^{3\alpha-1}}. && (102)
\end{aligned}
$$

By Definition C.11, we have $c = \frac{1-\alpha}{\alpha}$. Therefore, we have

$$\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_1/\mathcal{D}}}{d\,l} > 0 \tag{103}$$

Similarly, because k is sufficiently large,

$$\begin{align}
&\frac{d\gamma_{p_{\boldsymbol{\theta}}}^{\mathcal{D}_2/\mathcal{D}}}{d\,l} \tag{104}\\
=&\frac{1}{k\alpha^2(\alpha-1)^2(\alpha+1)c^{3\alpha-1}} \cdot \frac{(\alpha-1)\log c + 1}{\alpha^2(\alpha-1)^2} c^{3\alpha-1} \tag{105}\\
&- \frac{\alpha \log c - 1}{(k+1)^2(\alpha-1)^2\alpha^2 c^{3\alpha-1}} \cdot \frac{\alpha \log c + 1}{\alpha^3(\alpha-1)c^{3\alpha-1}} \tag{106}\\
=&\frac{1}{k\alpha^2(\alpha-1)^2(\alpha+1)c^{3\alpha-1}} \cdot \frac{(\alpha-1)\log c + 1}{\alpha^2(\alpha-1)^2} c^{3\alpha-1} \tag{107}\\
>&0. \tag{108}
\end{align}$$

□

D Experiment Details

D.1 Experiments Setup

Tasks and Datasets. We select two tasks: positive generation and single-turn safe conversation. For the former, we use the data classified as positive or negative in the IMDb dataset [19]. Referring to [8], we use the first 2-8 tokens of each complete text as a prompt for LLMs to generate the subsequent content. For the latter, we use the data classified as safe and unsafe in PKU-SafeRLHF [4]. We organize the positive sample sizes into {1000, 2000, 5000, 10000}, while the negative sample sizes, being fewer in number, were divided into {100, 200, 500, 1000, 2000}.

Model Training and Inference. We first verify elasticity in popular pre-trained LLMs, Gemma-2B [20] and Llama2-7B [7]. Subsequently, we examine the relationship between elasticity and model size using Qwen models [21] across sizes of 0.5B, 4B, and 7B. Finally, to analyze the connection between elasticity and the amount of pre-training data, we conduct further experiments on the 2.0T, 2.5T, and 3.0T slices of TinyLlama [22].

Evaluation and Metrics. We collect the model's responses on the reserved test prompts. Then we use score models provided by existing research to complete the performance evaluation. For positive style generation, we refer to [8] and use the Sentiment Roberta model [47] to classify the responses, taking the proportion of all responses classified as positive as the model score. For single-turn safe dialogue, we use the cost model provided by [12] to score the safety of each response, using the average score of all responses as the model score.

D.2 Experiments Results

To further verify the existence of model elasticity and the generality of its relationship with model size and pre-training data amount, we conducted experiments with different fine-tuning algorithms (DPO [8]), broader scales of pre-training data amount, and larger sizes of pre-trained models.

DPO Finetuning Experiments on LLMs Elasticity. We used the RLHF algorithm [9], specifically DPO [8], to fine-tune large language models in order to observe the elasticity of the models. The experimental procedure consists of the following two steps: 1) Perform SFT on the pre-trained model using various levels of positive sample data. 2) Use various levels of negative sample data, where negative samples are used as chosen responses and positive samples as rejected responses, to apply DPO for inverse alignment on the positively aligned model.

The experimental results are shown in Figure 5 and Figure 6. The results indicate that under DPO fine-tuning, LLMs continue to exhibit elasticity in style generation tasks. Additionally, as the model size and the amount of pre-training data increase, the model's elasticity shows an enhancing trend, which is consistent with the conclusions in Figures 3 and Figure 4. This suggests that differences in fine-tuning alignment algorithms do not affect the elasticity of language models.

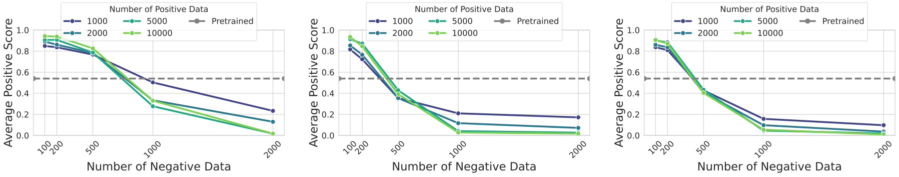


Figure 5: DPO Finetuning Experiments on IMDb. Each subfigure from left to right shows the changes in LLMs with parameter sizes of 0.5B, 4B, and 7B.

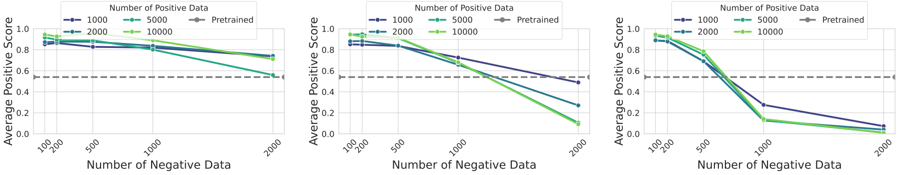


Figure 6: DPO Finetuning Experiments on IMDb. Each subfigure from left to right shows the changes in pre-training data sizes of 2.0T, 2.5T, and 3.0T.

Reverse Finetuning Experiments on LLMs Elasticity. To eliminate the influence of positive data on model elasticity experimental results, we adopted a reverse experimental setup: we use negative data during SFT stage and use positive data during inverse alignment stage. The experimental results are shown in Figure 7. The results indicate that elasticity in language models is still observed in the reverse experiments, showing a trend where larger model sizes correspond to greater elasticity, consistent with the results in Figure 3.

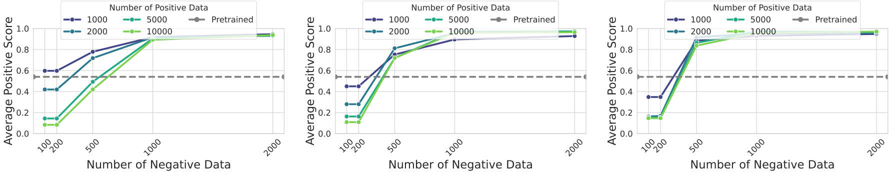


Figure 7: Reverse Fine-tuning Results on IMDb. Each subfigure from left to right shows the changes in LLMs with parameter sizes of 0.5B, 4B, and 7B, respectively.

Pre-training Data Amount Experiments on LLMs Elasticity. We present the experimental results for a broader range of pre-training data volumes in Figure 8. When the pre-training data volume is 0.1T, 0.5T, and 1.0T, the model still demonstrates the phenomenon that elasticity increases with the volume of pre-training data, which is consistent with the results reported in Figure 4, where the pre-training data sizes range from 2.0T to 3.0T.

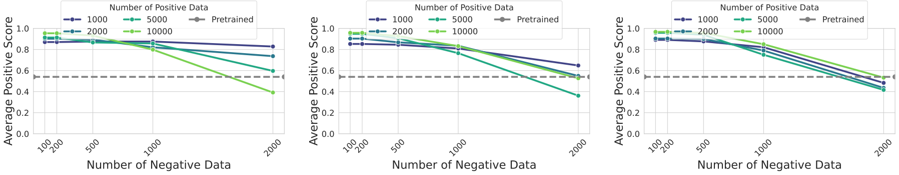


Figure 8: Experimental Results on IMDb. Each subfigure from left to right shows the changes in pre-training data sizes of 0.1T, 0.5T, and 1.0T.

Model Size Scale Experiment on LLMs Elasticity. To examine whether the elasticity phenomenon in language models is independent of model parameter size, we conducted experiments on the larger parameter-scale model, Qwen1.5 72B[21]. The experimental results are shown in Figure 9. The results indicate that even models with larger parameter sizes still exhibit the elasticity phenomenon, demonstrating that the presence of elasticity is not dependent on the size of the model parameters.

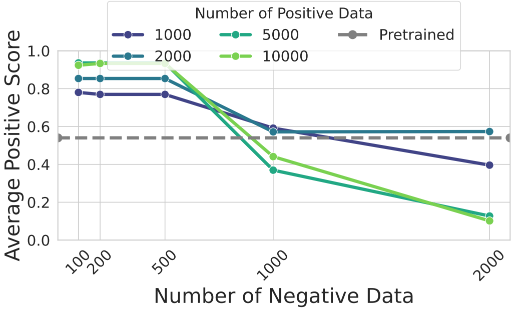


Figure 9: Experimental Results on IMDb with Qwen1.5 72B.

E Comparison between *Inverse Alignment* and *Forward Alignment*

The implicit resistance of language models to alignment raises another issue worth investigating: *Is inverse alignment easier than forward alignment?* In this section, we extract several sliced models during the fine-tuning process and construct twin datasets to perform inverse operations on models at different stages. We aim to verify the behavioral differences between forward and inverse alignment.

E.1 Experiment Setup

Tasks and Datasets. During the experiment, we select three tasks for extensive testing, including instruction-following [48], TruthfulQA [49], and PKU-SafeRLHF [4]. These tasks correspond to the widely accepted 3H standards (Helpful, Harmless, and Honest) [50] for LLMs. We divide the dataset into four equal parts to obtain four sliced models during the fine-tuning process. In the Harmless and Honest experiments, we pre-fine-tune the model with the 52K Alpaca [48] instruction-following dataset to equip the base model with conversational capabilities.

Model Training and Inference. We consider Llama2-7B, Llama2-13B [7] and Llama3-8B [51] as the base model θ_0. We adopt the AdamW optimizer with $\beta_1 = 0.99$, $\beta_2 = 0.95$, and weight dacay $= 0.01$ in all experiments. All models are trained on 8×A800 GPUs. During inference, we use the parameters of `top-k:30`, `top-p:0.9`, and `temperature:0.1`.

E.2 Experiment Design

As shown in Figure 10,starting from an pretrained model parameter $\boldsymbol{\theta}_k$ and a fune-tuning dataset $\mathcal{D}$, we divide the dataset $\mathcal{D}$ into several portions of the same size, $D_1, D_2, D_3 \ \ldots \ D_m$, using them to fine-tune model $\boldsymbol{\theta}_k$ sequentially, *i.e.*,

$$\boldsymbol{\theta}_{k+j} \xmapsto{(\mathcal{D}_j)} \boldsymbol{\theta}_{k+j+1}$$

Without loss of generality, we consider two sliced models, θ_{k+1} and θ_{k+2} during the fine-tuning process. Measuring the transition from model θ_{k+1} to model θ_{k+2} is straightforward, considering factors such as data volume, update steps, and parameter distribution. However, measuring the transition from model θ_{k+2} to model θ_{k+1}, *i.e.*, inverse alignment, is difficult. To address this challenge, we design the following experiment: we fine-tune models based on θ_{k+1} and θ_{k+2} to derive $\theta^{\dagger}_{k+1}$ and $\theta^{\dagger}_{k+2}$, which we designate as path A and path B, respectively. Specifically, we use a shared query set Q for paths A and B.

Path A. Responses generated by θ_{k+1} based on Q are used to form $Q - A^{\theta_{k+1}}$ pairs for path A's inverse alignment.

Path B. Similarly, responses generated by θ_{k+2} based on Q are used to form $Q - A^{\theta_{k+2}}$ pairs for path B's forward alignment.

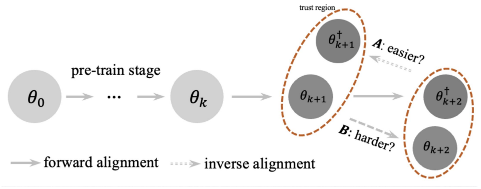


Figure 10: Parameter evaluation over slice models.

Given that paths A and B have identical training hyper-parameters and query set, we can assess the differences between $\theta^{\dagger}_{k+1}$ and θ_{k+1} (represented by δ_{k+1}), and between $\theta^{\dagger}_{k+2}$ and θ_{k+2} (represented by δ_{k+2}), utilizing the same training steps. If δ_{k+2} is consistently greater than δ_{k+1}, it suggests that $\theta^{\dagger}_{k+1}$ aligns more closely with θ_{k+1}. Consequently, inverse alignment proves more effective with the same training step than forward alignment. We use cross-entropy as the distance metric when calculating δ_{k+1} and δ_{k+2}.

E.3 Experiment Details

Specifically, we chose Alpaca52k dataset[48], 52k safe alignment dataset[4] and 72k truthful alignment dataset[4] as experiment datasets $\mathcal{D}$ and divided each experiment dataset into four parts: $D_1, D_2, D_3\ and\ D_4$. As shown in Figure 1, for the first three parts of the experiment data, we performed sequentially fine-tuning on the base models to obtain models $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3$. And for the safe and truthful dataset, we first used Alpaca52k[48] fine-tuning on the base models to endow them with instruction-following abilities, followed by relay fine-tuning on these models.

Next, we had these three models generate data on the D_4 dataset, resulting in the dataset required for inverse alignment and forward alignment. Using these datasets, we performed inverse alignment and forward alignment on three sets of models $(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$, $(\boldsymbol{\theta}_2, \boldsymbol{\theta}_3)$, and $(\boldsymbol{\theta}_1, \boldsymbol{\theta}_3)$. Based on the above training, we obtained three sets of models $(\boldsymbol{\theta}'_{12}, \boldsymbol{\theta}'_{21})$, $(\boldsymbol{\theta}'_{23}, \boldsymbol{\theta}'_{32})$, and $(\boldsymbol{\theta}'_{13}, \boldsymbol{\theta}'_{31})$.

We used cross entropy to compare the differences between the original models and the models generated by inverse alignment and forward alignment. For instance, for models $\boldsymbol{\theta}_2$ and $\boldsymbol{\theta}'_{12}$, we compared their cross entropy $H(\boldsymbol{\theta}_2, \boldsymbol{\theta}'_{12})$, as shown in Table 2.

In the actual experiment, we calculated the cross-entropy of two models by disabling loss backpropagation during a fine-tuning stage. This fine-tuning stage used test data generated by the model corresponding to the first parameter of the cross-entropy comparison. Specifically, the question of test data used came from a general field question set, and the fine-tuned model was the one whose parameter is the second parameter in the cross-entropy comparison. All the experiment hyper-parameters were in the Table 1

Table 1: Hyper-parameters of Experiment E.

Training Type	**Datasets**	**Epoch**	**Learning rate**	**Lr schedule Type**
Alpaca52k Fine-tuning	Alpaca52k	3	1e-5	cosine
Sequential Fine-tuning	All Dataset	1	1e-5	constant
Inverse Fine-tuning & Forward Fine-tuning	Alpaca52k	2	3e-6	constant
Inverse Fine-tuning & Forward Fine-tuning	Safe Dataset & Truthful Dataset	1	3e-6	constant

E.4 Experiment Results

Table 2: Comparsion between Inverse Alignment and Forward Alignment.

Datasets	Base Models	$H(p_{\theta_1}, p_{\theta^{\dagger}_{21}})$ *vs.* $H(p_{\theta_2}, p_{\theta^{\dagger}_{12}})$	$H(p_{\theta_2}, p_{\theta^{\dagger}_{32}})$ *vs.* $H(p_{\theta_3}, p_{\theta^{\dagger}_{23}})$	$H(p_{\theta_1}, p_{\theta^{\dagger}_{31}})$ *vs.* $H(p_{\theta_3}, p_{\theta^{\dagger}_{13}})$
Instruction-Following	Llama2-7B	0.1589 *vs.* 0.2018	0.1953 *vs.* 0.2143	0.1666 *vs.* 0.2346
	Llama2-13B	0.1772 *vs.* 0.1958	0.2149 *vs.* 0.2408	0.1835 *vs.* 0.2345
	Llama3-8B	0.2540 *vs.* 0.2573	0.2268 *vs.* 0.3229	0.2341 *vs.* 0.2589
Truthful	Llama2-7B	0.1909 *vs.* 0.2069	0.1719 *vs.* 0.1721	0.2011 *vs.* 0.2542
	Llama2-13B	0.1704 *vs.* 0.1830	0.1544 *vs.* 0.1640	0.1825 *vs.* 0.2429
	Llama3-8B	0.2118 *vs.* 0.2256	0.2100 *vs.* 0.2173	0.2393 *vs.* 0.2898
Safe	Llama2-7B	0.2730 *vs.* 0.2809	0.2654 *vs.* 0.2691	0.2845 *vs.* 0.2883
	Llama2-13B	0.2419 *vs.* 0.2439	0.2320 *vs.* 0.2327	0.2464 *vs.* 0.2606
	Llama3-8B	0.2097 *vs.* 0.2156	0.2008 *vs.* 0.2427	0.2277 *vs.* 0.2709

As shown in Table 2, the experimental results show that δ_{k+1} is smaller than δ_{k+2} across all three dimensions of the three types of models with all three types datasets. In addition to the comparisons mentioned in the experimental design, to eliminate the influence of model differences on the comparison between inverse alignment and forward alignment, we also compare θ_1 with θ_3, whose difference corresponds to δ_{k+1} and δ_{k+3} in the experimental design. The results are also as expected. All experimental results demonstrate that inverse alignment is easier than forward alignment across diverse models and datasets.