EvoBench: Towards Real-world LLM-Generated Text Detection Benchmarking for Evolving Large Language Models

Xiao Yu[1*], **Yi Yu**[2*], **Dongrui Liu**[2], **Kejiang Chen**[1†],
Weiming Zhang[1], **Nenghai Yu**[1], **Jing Shao**[2†]
[1]University of Science and Technology of China, China
[2]Shanghai Artificial Intelligence Laboratory
yuxiao1217@mail.ustc.edu.cn, yuyi@pjlab.org.cn
chenkj@ustc.edu.cn, shaojing@pjlab.org.cn

Abstract

With the widespread of Large Language Models (LLMs), there has been an increasing need to detect LLM-generated texts, prompting extensive research in this area. However, existing detection methods mainly evaluate on static benchmarks, which neglect the evolving nature of LLMs. Relying on existing static benchmarks could create a misleading sense of security, overestimating the real-world effectiveness of detection methods. To bridge this gap, we introduce EvoBench, a dynamic benchmark considering a new dimension of generalization across continuously evolving LLMs. EvoBench categorizes the evolving LLMs into (1) updates over time and (2) developments like finetuning and pruning, covering 7 LLM families and their 29 evolving versions. To measure the generalization across evolving LLMs, we introduce a new EMG (Evolving Model Generalization) metric. Our evaluation of 14 detection methods on EvoBench reveals that they all struggle to maintain generalization when confronted with evolving LLMs. To mitigate the generalization problems, we further propose improvement strategies. For zero-shot detectors, we propose pruning the scoring model to extract shared features. For supervised detectors, we also propose a practical training strategy. Our research sheds light on critical challenges in real-world LLM-generated text detection and represents a significant step toward practical applications. The Evobench is now available at: https://github.com/happy-Moer/EvoBench.

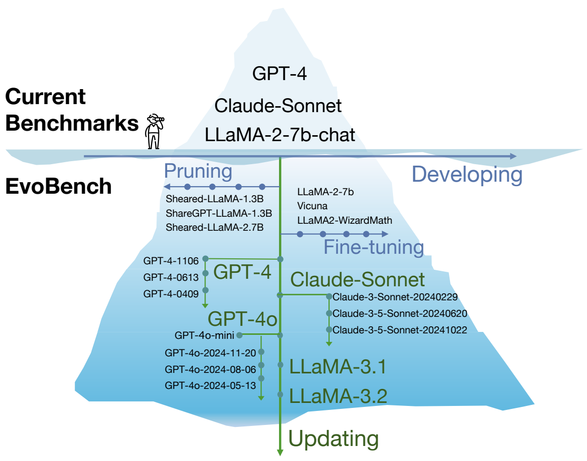

Figure 1: Current benchmarks (Guo et al., 2023; Bao et al.; Wang et al., 2024b,a; Kwan et al., 2024) primarily focus on specific versions of LLMs and neglect vast ecosystem of evolving LLMs beneath the surface, including updates over time or developments through fine-tuning or pruning. The figure primarily includes LLMs before Jan 2025.

1 Introduction

Large Language Models (LLMs), such as ChatGPT (OpenAI, 2022b), Claude (Anthropic, 2024), and LLaMA (Touvron et al., 2023a), have demonstrated remarkable capabilities in natural language understanding and task processing, leading to their widespread application (M Alshater, 2022; Yuan et al., 2022; Christian, 2023). However, concerns have been raised about the misuse of these models in areas like social media (Ahmed et al., 2021), education (Lee et al., 2023), and academic writing (Mitchell, 2022; Patrick Wood, 2023). For instance, LLMs can be used to manipulate the public by generating comments or to fabricate experimental data and statistical results in support of unverified hypotheses (Solaiman et al., 2019; Goldstein et al., 2023). The potential misuse of LLMs highlights the urgent need to detect LLM-generated text (Kaur et al., 2022; Chen and Shu, 2023).

The academic community has carried out extensive research to detect LLM-generated text effectively (Liu et al., 2019; Gehrmann et al., 2019; Su et al., 2023; Solaiman et al., 2019). Current methods can be categorized into supervised methods (Hu et al., 2023; Yu et al., 2024a; Chen et al.,

[*]Equal contribution. Work done during internship at Shanghai Artificial Intelligence Laboratory.
[†]Corresponding author.

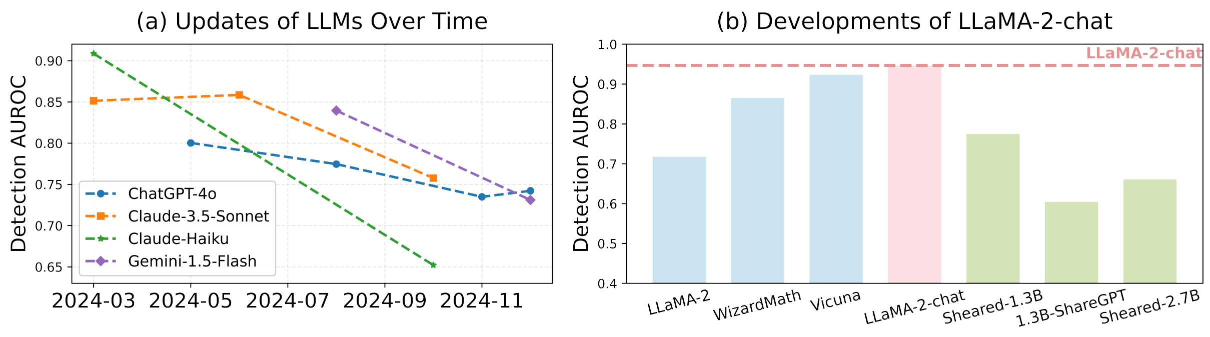

Figure 2: Detection accuracy (measured in AUROC) of Fast-DetectGPT when faced with the evolving LLMs. Figure (a) shows a clear decline in average detection performance as the LLM updates. Figure (b) illustrates the developments of LLMs, including fine-tuning and pruning.

2024b; Yu et al., 2024b; Guo et al., 2024) and zero-shot methods (Ippolito et al., 2020; Yang et al.; Mitchell et al., 2023; Su et al., 2023; Bao et al.). Supervised methods are typically trained with a binary classifier to distinguish between texts generated by LLMs and those created by humans, while zero-shot methods primarily rely on statistical features gathered from pre-trained large language models. Although these methods show strong performance on existing benchmarks (Bao et al.; Hu et al., 2023; Chen et al., 2024b; Yu et al., 2024a), they often fall short in real-world applications as current benchmarks neglect the evolving nature of LLMs (Wang et al., 2024b; Guo et al., 2023; Wang et al., 2024a; He et al., 2024; Macko et al., 2023). As illustrated in Figure 1, existing benchmarks include a limited versions of LLMs, much like observers seeing only the tip of an iceberg while overlooking the vast evolving LLMs hidden beneath the surface.

In real-world applications, LLMs are continuously evolving via regular updates, fine tuning (Touvron et al., 2023b; Zhang et al., 2023), or pruning (Sun et al.; Liang et al., 2021), all of which affect LLMs' output (Tao et al., 2024; Touvron et al., 2023a; Gunasekar et al., 2023), thereby impacting the detection performance. Figure 2 illustrates that the detection performance of current detection methods suffers up to 25% drop as LLMs evolve.[1] Therefore, relying on current static benchmarks for evaluation could create a misleading sense of security, leading the research community to overestimate the real-world effectiveness of detection methods. Therefore, it is crucial to enhance existing benchmarks to better capture the ongoing evolving LLMs.

In this paper, we propose EvoBench, a dynamic benchmark that extends traditional benchmarking to account for the evolving LLMs. EvoBench aims to provide a more accurate evaluation of detection methods by incorporating two dimensions of evolving LLMs: **updates** and **developments**, as shown in Figure 1. Updates refer to changes made by the LLM publishers, including LLM updates, *i.e.*, GPT-4o undergoes updates approximately every 3 months [2]. On the other hand, developments refer to optimizations made by LLM developers for specific application scenarios, such as fine-tuning and pruning. EvoBench introduces a new dimension of generalization, enabling the evaluation of detection methods across evolving LLMs, covering 7 widely used LLMs and their 29 evolving versions.

Using EvoBench, we evaluate 14 widely used detection methods and find that all struggle to adapt to the evolving nature of LLMs. To intuitive quantify the generalization across evolving LLMs, we introduce the EMG (Evolving Model Generalization) metric. Results reveal that the performance of 14 current detection methods, including widely used Fast-DetectGPT (Bao et al.) and RADAR (Hu et al., 2023) detectors, significantly drop when faced with evolving versions, as reflected by low EMG values, while some achieve high AUROC (0.91) scores on individual models[3]. This phenomenon highlights the limitations of current detection methods in generalization capabilities.

To mitigate this challenge, we explore two complementary strategies targeting zero-shot and supervised detection methods, respectively. For zero-shot detectors, we propose pruning the scoring

[1]The widely used detection method, Fast-DetectGPT, declines up to 25% drop when detecting the Claude-3-5-haiku-20241022 compared to Claude-3-haiku-20240307.

[2]These versions are gpt-4o-2024-05-13, gpt-4o-2024-08-06, gpt-4o-2024-11-20, and chatgpt-4o-latest.

[3]Fast-DetectGPT achieve 0.91 AUROC when detecting Claude-Haiku-2024-03-07, but drop to 0.65 AUROC when detecting the version of 2024-10-22.

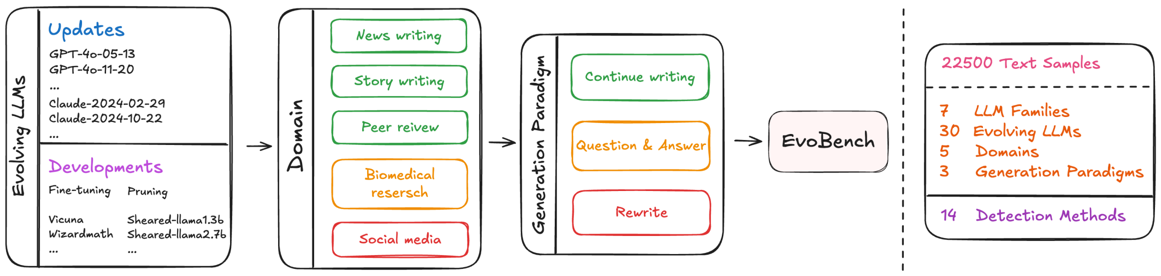

Figure 3: The construction pipeline and the dataset statistics of EvoBench. EvoBench includes three dimensions of generalization: dataset domains, generation strategies, and evolving LLMs. The evolving LLMs are further divided into two categories: updates and developments.

model to extract shared features that enhance detection generalization. This strategy is not limited to Fast-DetectGPT (Bao et al.), which we adopt as a representative case, but is applicable to a broader class of zero-shot methods. Our experiments demonstrate that this approach consistently improves performance across a range of detectors—including Likelihood, Rank, LogRank, Fast-DetectGPT, and BiScope—when using Sheared-LLaMA as the scoring model in detecting text generated by developments of LLaMA-2. For supervised detectors, we propose a practical training strategy aimed at enhancing generalization: including training data from the initial version of an LLM family can significantly improve performance on its subsequent updates.

Overall, EvoBench serves as a complementary benchmark that focuses on the evolving nature of LLMs, offering a more accurate framework for evaluating the real-world applicability of detection methods. While it does not claim to be a one-size-fits-all solution, we envision EvoBench evolving alongside detection methods and other benchmarks, helping guide future developments in this area. In particular, we suggest that extracting shared features from evolving LLMs could enhance the generalization of detection methods. With EvoBench, the community can more effectively evaluate and refine detection methods, contributing to more robust and practical detection systems that mitigate LLM misuse in real-world applications.

2 Related Work

2.1 Evolving LLMs

Updates of Pre-trained LLMs. The advent of LLMs, such as ChatGPT (OpenAI, 2022a) and GPT-4 (OpenAI, 2023), has marked a paradigm shift in text generation. However, these pre-trained models are not static (Tao et al., 2024; Touvron et al., 2023a; Gunasekar et al., 2023; Biderman et al., 2023); rather, they undergo continuous evolution, with models like ChatGPT frequently updating their parameters to enhance performance and adapt to new application scenarios (Zheng et al., 2024; Touvron et al., 2023a).

Developments on Pre-trained LLMs. The rise of open-source LLMs has facilitated their widespread usage in diverse domains. Developers can flexibly tailor LLMs to specific tasks through fine-tuning (Touvron et al., 2023a,b; Zhang et al., 2023), pruning (Sun et al.; Liang et al., 2021; Xia et al.), and quantization (Du et al., 2024; Chen et al., 2024a; Dettmers et al., 2023) techniques that modify the model's parameters or structure, ultimately influencing output characteristics. In this process, developers could use domain-specific datasets to enhance the model's understanding of specialized fields and their contextual knowledge (Kerner, 2024). Furthermore, depending on practical application scenarios, users may select models with varying parameter scales to strike an optimal balance between performance, speed, and resource consumption (Kim et al., 2024; Zhao et al., 2023).

2.2 LLM-generated Text Detection

Supervised Methods. Supervised methods (Liu et al., 2019; Chen et al., 2024b) are usually trained to differentiate between texts generated by LLMs and texts created by humans. For example, RADAR (Hu et al., 2023) introduces the idea of adversarial learning to train a detector that can resist paraphrase attacks. Text Fluoroscopy (Yu et al., 2024a) extracts discriminative features from the intermediate layers of the language model and utilizes them to train a binary classifier, which enhances the generalization of supervised detectors across texts from different semantic domains.

However, as LLMs continue to evolve, frequent adaptation mechanisms—including updates, fine-tuning, pruning, and other optimization strategies—introduce changes in their generated text, making it difficult to guarantee the effectiveness of existing detection methods on newer model versions. This creates a significant gap between academic detection models and real-world applications, ultimately limiting the practical utility of these methods.

Zero-shot Methods. Existing zero-shot methods primarily rely on statistical features extracted using pre-trained large language models (Bao et al.; Mitchell et al., 2023). These features include likelihood (Gehrmann et al., 2019), probability curvature (Mitchell et al., 2023), divergence between multiple completions of a truncated passage (Yang et al.), and conditional probability curvature (Bao et al.). Zero-shot detection methods are immune to domain-specific degradation, demonstrating superior generalization in detection tasks (Gehrmann et al., 2019; Mitchell et al., 2023). However, the effectiveness of zero-shot detection heavily depends on the alignment between the pre-trained LLM used for detection and the generation model that produced the text to be detected (Bao et al.; Mitchell et al., 2023). While current methods achieve state-of-the-art results on existing benchmarks (Guo et al., 2023; Bao et al.; Wang et al., 2024b,a; Kwan et al., 2024), the evolving nature of LLMs introduces significant shifts in this alignment. As models continue to evolve, the pre-trained LLMs used for detection may become increasingly misaligned with the updated generation models, leading to a decline in detection performance. This explains why many methods, despite ranking highly on leaderboards, often fail when deployed in real-world scenarios.

3 EvoBench

3.1 Definition of Evolving LLMs

EvoBench primarily considers two evolving pathways for LLMs: (1) **updates**, which are released by publishers, and (2) **developments**, which involve optimization made by developers. These two evolving pathways present significant challenges to current detection methods.

Updates. We focus on two types of updates that occur over time: model (Brown et al., 2020; Tao et al., 2024) and version updates (Brown et al., 2020). Model updates typically involve significant changes to the model's architecture. For example, the transition from GPT-4 to GPT-4o represents a major architectural shift (Achiam et al., 2023). In contrast, version updates occur more frequently, with shorter cycles and less perception, such as the regular releases of new versions of GPT-4 every 2-3 months, such as updates in May, August, and November 2024.

Developments. In EvoBench, model developments include fine-tuning (Hu et al., 2022; Mangrulkar et al., 2022), pruning (Liang et al., 2021), often used for domain adaptation and optimizing efficiency for real-world applications. Unlike updates, these actions are nearly imperceptible to the detector (Bhattacharjee et al., 2023). For example, developers may fine-tune LLMs using private datasets, making the detection more challenging.

3.2 Dataset Collection

3.2.1 Data Collection Principle

EvoBench, incorporating the two evolving pathways of LLMs, extends existing benchmarks (Mitchell et al., 2023) widely used in current detection methods, including DetectGPT (Mitchell et al., 2023), Fast-DetectGPT (Bao et al.), and ImBD (Chen et al., 2024b) to assess three key dimensions of generalization: (1) evolving LLMs, (2) domain, and (3) generation paradigms. Figure 3 illustrates our data collection pipeline to cover these three key dimensions.

Generalization to LLM Evolution. To comprehensively assess evolving, we consider 7 widely used LLM families: GPT-4o, GPT-4, Claude-Haiku, Claude-Sonnet, Qwen, LLaMA3, and LLaMA2, encompassing a total of 29 evolving versions. For updates, we focus on families of GPT-4o, GPT-4, Claude, Gemini, Qwen, and LLaMA3, tracking how detection methods perform across updates of these models. For developments, we focus on the LLaMA2 family, incorporating fine-tuning and pruning techniques. Detailed specifications of all LLMs are provided in the Appendix C.

Generalization to Domains. EvoBench includes five datasets spanning diverse tasks: *XSum* (Narayan et al., 2018) for (1)news articles, *WritingPrompts* (Fan et al., 2018) for (2) creative story writing, and *PubMed* (Jin et al., 2019) for (3) biomedical research question answering. Additionally, we also include two datasets: *SocialMedia* (Kula and Gregor, 2024) for (4) datasets on social media and *PeerRead* (Kang et al., 2018) for

(5) peer-reviewed academic writing.

Generalization to Generation Paradigms. EvoBench incorporates three distinct generation paradigms: (1) continuation-based generation, including *XSum*, *Writing*, and *PeerRead*; (2) question-answering generation, including *PubMed*; (3) paraphrased generation, including *SocialMedia*. In the *SocialMedia* dataset, we specifically focus on paraphrastic rewriting of text, which presents a particularly challenging scenario for detection. By integrating these diverse paradigms, EvoBench offers a more comprehensive framework to assess the robustness of detection methods in various text generation paradigms.

In addition, EvoBench accounts for variation in text length. *PubMed* and *SocialMedia* primarily consist of shorter texts, while *PeerRead* includes medium-length samples, and *XSum* and *Writing* represent long-form text generation. This diversity in length further enhances the benchmark's ability to test the generalization capabilities of detection methods under varied textual conditions.

3.2.2 Dataset Collection Process

We detail our systematic dataset construction process following the three-dimensional generalization framework. For each dataset, we first carefully selected 150 human-authored texts as reference samples. To collect AI-generated texts, we employed consistent prompting strategies across all evaluated models. Specifically, for continuation-based generation tasks (*Xsum*, *Writing*, *PeerRead*), we provided the prefix of human-written texts as context. For *PubMed*, we maintained the original question-answer format, while for *SocialMedia*, we implemented a paraphrasing approach where models were tasked with preserving semantic meaning while using different words. Details of prompts are shown in Appendix E.

3.3 Dataset Statistics

EvoBench consists of 7 LLM families, encompassing a total of 29 evolving versions. For each evolving LLM, we collect samples across 5 distinct datasets, with each dataset containing a balanced distribution of 150 human-authored texts and 150 machine-generated texts. In total, EvoBench comprises $29 \times 5 \times 150 = 21750$ machine-generated samples, resulting in a comprehensive evaluation benchmark of $21,750$ pairs of text samples.

3.4 Evaluation Metrics

To quantify the generalization ability of detection methods to evolving LLMs, we introduce the EMG Φ_E (Evolving Model Generalization) metric, inspired by the coefficient of variation. It evaluates the consistency and trend of detection performance across evolving LLMs.

For an LLM family with m evolving versions, the widely evaluated version is selected as the $base$ model, and the remaining $m-1$ versions as evolving models. The performance (measured by AUROC Φ_A) change $\Delta\Phi_A^i$ between the i-th models and the $base$ model is :

$$\Delta\Phi_A^i = \Phi_A^i - \Phi_A^{base}, \tag{1}$$

where Φ_A^i and Φ_A^{base} is the Φ_A of the i-th evolving model and $base$ model, respectively.

The average performance change μ_E of $\Delta\Phi_A$, representing the overall performance change, is:

$$\mu_E = \frac{1}{m-1}\sum_{i=1}^{m-1}\left(\Delta\Phi_A^i\right). \tag{2}$$

The volatility σ_E of $\Delta\Phi_A$ using the standard deviation is:

$$\sigma_E = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m-1}\left(\Delta\Phi_A^i - \mu_E\right)^2}. \tag{3}$$

The proposed Φ_E is defined as follows:

$$\Phi_E = \frac{\mu_E}{\sigma_E + \lambda} \times \gamma, \tag{4}$$

where λ is a regularization term to prevent fluctuations, and γ is a scaling factor. In this study, we set λ and γ as 1 and 100, respectively.

3.5 Detection Methods

We evaluate a range of supervised and zero-shot detection methods to assess their generalization ability against evolving LLMs. Additionally, we conducted repeated generation processes to ensure that output diversity does not significantly impact detection performance in Appendix B.

3.5.1 Existing Methods

For supervised detectors, we tested the GPT-2 detectors developed by OpenAI (Liu et al., 2019), the RADAR detector (Hu et al., 2023), Text Fluoroscopy (Yu et al., 2024a), and ImBD (Imitate Before Detect) (Chen et al., 2024b). For zero-shot detectors, we included Likelihood (average log probabilities) (Gehrmann et al., 2019), LRR (a hybrid method combining log probability and log-rank) (Su et al., 2023), LogRank (mean of the log ranks sorted in descending order of probabilities) (Solaiman et al., 2019), Entropy (average token-level entropy from the predictive distribution)(Ippolito et al., 2020), DNA-GPT (Yang et al.), DetectGPT (Mitchell et al., 2023), and its enhanced variants NPR (Su et al., 2023) and Fast-DetectGPT (Bao et al.).

3.5.2 Optimizing Strategies

To address the challenges mentioned above, we introduce a baseline method for EvoBench and explore it from two perspectives.

Enhancing Zero-shot Detector via Extracting Shared Features. To enhance the detection generalization of the zero-shot method, we propose to prune the scoring model p_θ^{pruned} to extract the shared features from the developments of LLMs. Take Fast-DetectGPT as an example, we define our detection metric $d(x, p_\theta^{\text{pruned}})$ based on conditional probability curvature:

$$d(x, p_\theta^{\text{pruned}}) = \frac{\log p_\theta^{\text{pruned}}(x|x) - \tilde{\mu}}{\tilde{\sigma}}, \qquad (5)$$

where $\tilde{\mu}$ represents the expected log probability under the pruned model:

$$\tilde{\mu} = \mathbb{E}_{\tilde{x}\sim p_\theta^{\text{pruned}}(\tilde{x}|x)}[\log p_\theta^{\text{pruned}}(\tilde{x}|x)], \qquad (6)$$

and $\tilde{\sigma}^2$ captures the variance of these log probabilities:

$$\tilde{\sigma}^2 = \mathbb{E}_{\tilde{x}\sim p_\theta^{\text{pruned}}(\tilde{x}|x)}\left[\left(\log p_\theta^{\text{pruned}}(\tilde{x}|x) - \tilde{\mu}\right)^2\right]. \qquad (7)$$

In addition, this optimization strategy is not limited to Fast-DetectGPT; rather, it is designed for the broader class of zero-shot detectors. These detectors (Gehrmann et al., 2019) use a scoring model to extract statistical features from human- and AI-generated text, and then perform detection based on these features.

Enhancing Supervised Detectors via Optimizing Training Data. To improve the generalization of supervised detectors across evolving LLMs, we propose an incremental training strategy that focuses on optimizing the training data. Specifically, for each LLM family, we include only the initial version in the training set and evaluate the detector's performance on subsequent, unseen versions. This approach reduces the need for frequent retraining while enhancing robustness to model updates.

4 Exprimental Results

4.1 Setup

Detection Methods. To mimic real-world scenarios, we set up a black-box environment in which the detector is assumed to be unaware of the source model of the text to be detected under evaluation. We tested 14 detection methods, including 5 supervised detectors and 9 zero-shot detectors. Among them, RADAR (Hu et al., 2023) and Fast-DetectGPT (Bao et al.) are well-known methods in their respective categories. Details of experimental settings can be found in the Appendix D and Appendix F.

4.2 Main Results

To begin with, we select two leading detection methods, Fast-DetectGPT and RADAR detector, and evaluate them on our EvoBench, the results are shown in Table 1.

First, a trend is that texts generated by **more advanced LLMs are harder to detect in the same LLM family.** For instance, comparing Claude-3-Haiku[4] and Claude-3-Opus [5], the latter demonstrates more advanced capabilities. When detecting these two LLMs, Fast-DetectGPT and RADAR showed decreased detection AUROC of 3.58% and 2.77%, respectively, as shown in Table 1. Similarly, comparing detecting *SocialMedia* datasets generated by GPT-4o-mini with GPT-4o, two detectors showed a decline in detection AUROC of 3.22% and 6.69%, respectively.

Our results also reveal an intriguing pattern: **existing detection methods often over-optimize for specific LLM versions**, rather than maintaining robust generalization capabilities across different evolving LLMs. Specifically, detectors with high AUROC on widely used LLM families often show performance degradation with newer versions of the same family. For example, Fast-DetectGPT performed excellently on the GPT-4o-05-13 version, achieving an average detection accuracy of 0.8003 on EvoBench, but dropped to 0.7422 when facing the GPT-4o-latest version. Similar patterns were observed in other widely used LLM families, such as GPT-4 and Claude-Sonnet. In contrast, when evaluating less widely used LLMs like LLaMA3, detectors tend to demonstrate better generalization capabilities across evolving LLMs. For example, the detection performance of the RADAR remained

[4]claude-3-haiku-20240307

[5]claude-3-opus-20240229

LLMs	Version Time/ Version Name	Fast-DetectGPT						RADAR					
		Xsum	Writing	PubMed	SocialMedia	PeerRead	Avg.	Xsum	Writing	PubMed	SocialMedia	PeerRead	Avg.
Updates													
GPT-4o	2024-05-13	0.90	0.97	0.73	0.58	0.83	0.80	0.99	0.83	0.83	0.60	0.68	0.79
	2024-08-06	0.87 (−2.75%)	0.94 (−2.41%)	0.70 (−3.54%)	0.59 (+1.76%)	0.77 (−5.89%)	0.77 (−2.57%)	0.99 (−0.11%)	0.86 (+3.27%)	0.82 (−0.53%)	0.62 (+2.12%)	0.69 (+0.75%)	0.80 (+1.10%)
	2024-11-20	0.72 (−17.6%)	0.90 (−6.90%)	0.70 (−3.77%)	0.57 (−0.09%)	0.78 (−4.25%)	0.73 (−6.54%)	0.99 (−0.87%)	0.72 (−11.0%)	0.79 (−4.08%)	0.62 (+2.02%)	0.66 (−1.47%)	0.75 (−3.09%)
	Latest	0.70 (−20.0%)	0.91 (−6.13%)	0.70 (−3.36%)	0.58 (+0.64%)	0.82 (−0.14%)	0.74 (−5.81%)	0.99 (−0.79%)	0.75 (−7.49%)	0.80 (−2.83%)	0.64 (+4.46%)	0.67 (−0.53%)	0.77 (−1.44%)
GPT-4o-mini	2024-07-18	0.91 (+0.91%)	0.97 (+0.12%)	0.73 (−0.42%)	0.61 (+3.02%)	0.80 (−2.14%)	0.80 (+0.30%)	1.00 (+0.33%)	0.87 (+3.82%)	0.82 (−0.68%)	0.66 (+6.69%)	0.64 (−3.96%)	0.80 (+1.24%)
GPT-4	2023-06-13	0.93	0.98	0.76	0.49	0.75	0.78	0.99	0.87	0.89	0.66	0.75	0.83
	2023-11-06	0.92 (−0.85%)	0.93 (−4.76%)	0.73 (−2.94%)	0.55 (+5.43%)	0.83 (+7.71%)	0.79 (+0.92%)	0.99 (−0.01%)	0.79 (−8.49%)	0.80 (−8.81%)	0.66 (+0.24%)	0.63 (−12.3%)	0.77 (−5.88%)
	2024-01-25	0.89 (−3.22%)	0.94 (−3.41%)	0.70 (−5.75%)	0.54 (+4.23%)	0.82 (+7.34%)	0.78 (−0.16%)	0.99 (−0.23%)	0.80 (−7.23%)	0.84 (−5.09%)	0.65 (−0.46%)	0.66 (−8.68%)	0.79 (−4.34%)
	2024-04-09	0.83 (−9.30%)	0.91 (−6.56%)	0.72 (−4.39%)	0.58 (+8.27%)	0.79 (+4.08%)	0.77 (−1.58%)	0.99 (−0.36%)	0.79 (−8.45%)	0.82 (−7.07%)	0.60 (−5.92%)	0.60 (−14.8%)	0.76 (−7.32%)
Claude-Sonnet	2024-02-29	0.95	0.98	0.81	0.58	0.93	0.85	0.97	0.86	0.84	0.74	0.61	0.80
	2024-06-20	0.97 (+2.33%)	0.99 (+0.80%)	0.77 (−4.32%)	0.66 (+7.69%)	0.90 (−2.90%)	0.86 (+0.72%)	0.99 (+1.61%)	0.81 (−5.21%)	0.82 (−2.44%)	0.72 (−1.05%)	0.55 (−5.57%)	0.78 (−2.53%)
	2024-10-22	0.90 (−4.43%)	0.93 (−5.39%)	0.68 (−13.9%)	0.51 (−7.80%)	0.78 (−15.3%)	0.76 (−9.37%)	0.95 (−1.52%)	0.72 (−13.4%)	0.74 (−9.85%)	0.67 (−6.74%)	0.55 (−6.38%)	0.73 (−7.59%)
Claude-Haiku	2024-03-07	1.00	1.00	0.86	0.75	0.94	0.91	1.00	0.93	0.84	0.77	0.67	0.84
	2024-10-22	0.84 (−15.5%)	0.92 (−7.56%)	0.64 (−21.7%)	0.39 (−35.9%)	0.46 (−47.4%)	0.65 (−25.6%)	1.00 (−0.07%)	0.86 (−7.50%)	0.73 (−10.9%)	0.70 (−6.38%)	0.82 (+14.7%)	0.82 (−2.04%)
Claude-Opus	2024-02-29	0.97 (−2.49%)	0.96 (−3.89%)	0.82 (−3.80%)	0.72 (−3.27%)	0.89 (−4.46%)	0.87 (−3.58%)	0.99 (−0.33%)	0.87 (−6.01%)	0.82 (−2.23%)	0.77 (+0.29%)	0.62 (−5.57%)	0.81 (−2.77%)
Qwen	Qwen1.5-7B	0.92	0.99	0.66	0.49	0.91	0.80	0.93	0.90	0.72	0.51	0.48	0.71
	Qwen2-7B	0.99 (+6.86%)	1.00 (+0.29%)	0.74 (+7.35%)	0.46 (−2.75%)	0.89 (−1.37%)	0.82 (+2.08%)	0.89 (−4.13%)	0.87 (−3.42%)	0.82 (+9.46%)	0.52 (+1.35%)	0.54 (+6.14%)	0.73 (+1.88%)
	Qwen2.5-7B	0.99 (+6.51%)	1.00 (+0.10%)	0.79 (+12.3%)	0.47 (−2.11%)	0.90 (−0.78%)	0.83 (+3.22%)	0.90 (−2.60%)	0.91 (+0.66%)	0.79 (+6.85%)	0.52 (+1.31%)	0.53 (+4.95%)	0.73 (+2.23%)
LLaMA3	LLaMA-3.1-8B	0.99	0.98	0.85	0.69	0.97	0.90	1.00	0.89	0.78	0.72	0.63	0.80
	LLaMA-3.1-70B	1.00 (+0.31%)	1.00 (+1.50%)	0.81 (−3.72%)	0.67 (−1.73%)	0.96 (−0.96%)	0.89 (−0.92%)	1.00 (−0.19%)	0.88 (−1.32%)	0.76 (−1.57%)	0.74 (+1.29%)	0.62 (−1.22%)	0.80 (−0.60%)
	LLaMA-3.2-1B	0.96 (−3.46%)	0.99 (+1.09%)	0.92 (+7.29%)	0.86 (+17.5%)	0.96 (−0.30%)	0.94 (+4.43%)	0.96 (−3.51%)	0.94 (+4.39%)	0.82 (+3.74%)	0.75 (+2.80%)	0.71 (+7.52%)	0.83 (+2.99%)
	LLaMA-3.2-3B	0.99 (−0.13%)	0.99 (+0.31%)	0.91 (+6.56%)	0.77 (+8.32%)	0.96 (−0.30%)	0.93 (+2.95%)	0.99 (−0.44%)	0.92 (+2.71%)	0.82 (+3.70%)	0.77 (+4.23%)	0.66 (+2.80%)	0.83 (+2.60%)
	LLaMA-3.3-70B	1.00 (+0.32%)	1.00 (+1.61%)	0.79 (−5.53%)	0.69 (+0.15%)	0.90 (−6.38%)	0.88 (−1.97%)	1.00 (−0.32%)	0.89 (−0.74%)	0.79 (+0.94%)	0.73 (+0.76%)	0.65 (+2.29%)	0.81 (+0.59%)
Developments													
Fine-tuning	LLaMA-2-7B-chat-hf	0.97	0.99	0.91	0.90	0.97	0.95	0.92	0.81	0.70	0.64	0.64	0.74
	Vicuna-1.5-7B	1.00 (+2.36%)	1.00 (+0.52%)	0.80 (−10.4%)	0.88 (−1.58%)	0.94 (−2.54%)	0.92 (−2.34%)	0.97 (+5.00%)	0.93 (+12.0%)	0.81 (+11.0%)	0.64 (−0.00%)	0.65 (+1.00%)	0.80 (+5.80%)
	Wizardmath-7B	0.92 (−5.17%)	0.98 (−1.00%)	0.74 (−17.0%)	0.85 (−4.73%)	0.84 (−12.8%)	0.87 (−8.14%)	0.75 (−17.0%)	0.76 (−5.00%)	0.70 (−0.00%)	0.55 (−9.00%)	0.68 (+4.00%)	0.69 (−5.40%)
Pruning	Sheared-llama1.3B	0.77 (−20.2%)	0.91 (−8.04%)	0.56 (−34.5%)	0.76 (−13.6%)	0.88 (−8.93%)	0.78 (−17.0%)	0.77 (−15.0%)	0.66 (−15.0%)	0.59 (−11.0%)	0.54 (−10.0%)	0.73 (+9.00%)	0.66 (−8.40%)
	Sheared-LLaMA1.3B-pruned	0.50 (−47.2%)	0.83 (−16.0%)	0.39 (−51.5%)	0.53 (−36.6%)	0.78 (−18.9%)	0.61 (−34.0%)	0.92 (−0.00%)	0.79 (−2.00%)	0.70 (−0.00%)	0.65 (+1.00%)	0.87 (+23.0%)	0.79 (+4.40%)
	Sheared-LLaMA2.7B-pruned	0.60 (−37.2%)	0.89 (−10.0%)	0.44 (−46.5%)	0.59 (−30.6%)	0.79 (−17.9%)	0.66 (−28.4%)	0.84 (−8.00%)	0.76 (−5.00%)	0.59 (−11.0%)	0.55 (−9.00%)	0.78 (+14.0%)	0.70 (−3.80%)

Table 1: The detection performance (measured in AUROC) of two leading detection methods, Fast-DetectGPT and RADAR, on EvoBench. For each LLM family, we set the LLMs that are widely evaluated in text generation detection benchmarks as anchor points. 'Latest' refers to the LLM currently used in the web version of GPT-4o.

stable across different versions of the LLaMA3 family, with AUROC scores across five datasets ranging from 0.7989 to 0.8348.

This suggests that the essence of **generalization lies in the overfitting problem**. Current detectors are often optimized to achieve superior performance on existing benchmarks, which leads to overfitting to a specific LLM version rather than being truly effective. This also reveals the vulnerability of current detection methods across evolving LLMs. From a practical perspective, a key priority for future research lies in developing more adaptable detection frameworks that can simultaneously preserve detection performance while demonstrating robust generalization across dataset domains and evolving LLMs.

4.2.1 Detection Generalization across Evolving LLMs

To visually demonstrate the generalization ability of detection methods across evolving LLMs, we evaluate the 14 detection methods using the EMG metric, with results shown in Table 2.

Table 2 demonstrates that no method can maintain stable performance across all evolving LLMs. Specifically, detectors like ImBD and Text-Fluoroscopy perform well on complex models like GPT-4o and GPT-4. Meanwhile, detectors like entropy and RADAR exhibit better generalizations of newer model families, including Qwen, LLaMA3, and the developments of LLaMA2. This variation in performance across different detectors suggests that improving generalization might require adopting strategies that combine the strengths of different approaches, which may help improve the

Methods	Updates						Developments		Avg.
	GPT-4o	GPT-4	Claude-Sonnet	Claude-haiku	Qwen	LLaMA3	Fine-tuning	Pruning	
Supervised Detectors									
RoBERTa-base	−2.252	−4.593	−8.325	−15.95	+5.862	+1.457	−2.378	+6.871	−2.413
RoBERTa-large	−1.878	−5.577	−10.06	−14.63	+4.646	+3.448	−0.725	+12.54	−1.530
RADAR	−0.537	−5.775	−4.934	−2.394	+2.053	+1.372	+11.96	+7.302	+1.131
Text-Fluoroscopy	−0.493	−0.655	+0.902	−0.123	−5.241	+1.895	−11.03	−15.77	−3.815
Imitate Before Detect	+0.440	+2.397	−1.572	−4.445	+0.023	−0.054	+8.942	−9.842	−0.513
Zero-Shot Detectors									
Likelihood	−0.873	−0.047	−3.810	−8.280	−1.174	+0.439	−6.991	−32.99	−6.717
Rank	−2.127	+1.071	−0.178	−12.09	+4.249	+1.023	−9.661	−7.109	−3.103
LogRank	−1.316	−0.106	−4.272	−9.443	−0.608	+0.837	−6.851	−30.14	−6.488
Entropy	−3.202	+0.416	+0.602	−3.411	+4.138	+2.145	+5.150	+25.12	+3.869
LRR	−2.511	−0.075	−5.864	−14.26	+2.629	+1.911	−6.964	−17.45	−5.325
NPR	−2.846	+7.436	−2.237	−15.26	+7.132	+0.866	−8.063	−31.18	−5.520
DNA-GPT	−3.805	+1.249	−4.621	−12.26	−1.332	+1.337	−6.355	−36.11	−7.738
DetectGPT	−4.095	+10.04	+2.258	−11.09	+2.357	−0.336	−5.442	−20.18	−3.312
Fast-DetectGPT	−3.558	−0.271	−4.115	−13.16	+2.632	+1.094	−5.095	−27.94	−6.302

Table 2: EMG performance of 14 detection methods on EvoBench. Red indicates a negative EMG, signifying a decrease in AUROC when facing evolving LLMs, while a larger EMG value reflects a better generalization of the detection method.

Developments of LLaMA-2	Detection Methods	LLaMA-2-7b (baseline)	Sheared-LLaMA-1.3B	Sheared-LLaMA-2.7B	Wanda-LLaMA
Fine-tuning	Likelihood	**0.8616**	0.8515	0.8529	0.8593
	Rank	0.6288	**0.6781**	0.6469	0.6163
	LogRank	**0.8762**	0.8551	0.8632	0.8714
	Entropy	**0.3908**	0.3362	0.3524	0.3881
	Fast-detectGPT	0.9350	0.9152	**0.9376**	0.9339
	Average	**0.7384**	0.7272	0.7306	0.7338
Pruning	Likelihood	0.3921	**0.7948**	0.5378	0.4633
	Rank	0.6583	**0.7950**	0.7179	0.6113
	LogRank	0.4273	**0.8209**	0.5806	0.5046
	Entropy	**0.7343**	0.4614	0.5407	0.6871
	Fast-detectGPT	0.6847	**0.8913**	0.8066	0.7518
	Average	0.5793	0.7526	0.6367	0.6036
Fine-tuning and Pruning	Likelihood	0.6269	**0.8232**	0.6954	0.6613
	Rank	0.6436	**0.7366**	0.6824	0.6138
	LogRank	0.6518	**0.8380**	0.7219	0.6880
	Entropy	**0.5625**	0.3988	0.4465	0.5376
	Fast-detectGPT	0.8099	**0.9033**	0.8721	0.8429
	Average	0.6589	**0.7400**	0.6837	0.6687

Table 3: Average Detection performance across LLaMA-2 developments, including fine-tuning and pruning versions, using five detection methods with different scoring models.

adaptability of the detector and reduce sensitivity to specific LLM evolution trends.

4.2.2 Performance of Optimizing Strategies

Extracting Shared Features To improve the generalization of current detection methods across evolving LLMs, we have preliminarily explored two possible approaches, with results presented in Table 3.

For zero-shot detection methods, we propose a general optimization strategy called extracting shared features, which is not limited to Fast-DetectGPT but is designed for the broader class of zero-shot detectors. Specifically, we prune the scoring model to preserve the discriminative features that are shared across different generative models, aiming to improve generalization against model developments such as fine-tuning and pruning. To validate the effectiveness of this strategy more broadly, we conduct experiments across multiple detectors, including Likelihood, Rank, LogRank, and Entropy. Additionally, we examine how different pruning strategies for the scoring model impact detection performance. We use LLaMA-2-7b as the baseline and evaluate three pruned versions: Sheared-LLaMA-1.3B, Sheared-LLaMA-2.7B, and Wanda-LLaMA.

LLMs	Version Time/ Version Name	Text Fluoroscopy	
		Original Dataset	Adding Initial Version
GPT-4o	GPT-4o-mini	0.8373	**0.8464**
	2024-08-06	0.8240	**0.8327**
	2024-11-20	0.7910	**0.8035**
	Latest	0.7986	**0.8057**
GPT-4	2023-11-06	0.8117	**0.8633**
	2024-01-25	0.8061	**0.8549**
	2024-04-09	0.8131	**0.8584**
Claude-3.5-Sonnet	2024-06-20	0.8699	**0.8720**
	2024-10-22	0.8218	**0.8401**

Table 4: Average Detection Performance across updates of three LLM families with and without the initial version in the training dataset under Text Fluoroscopy evaluation.

We assess the average detection AUROC of each configuration under two types of model development—fine-tuning and pruning. The results, shown in the table below, demonstrate that using Sheared-LLaMA-1.3B as the scoring model consistently leads to higher detection performance across all detectors. This suggests that the pruned model retains key shared features, making it a strong general-purpose scoring model for detecting AI-generated text across a variety of model variants.

Including the Initial Version For supervised detectors, we introduce a training set-based generalization strategy that can be applied beyond a single method. Rather than tailoring our approach to a specific architecture, we propose a practical and broadly applicable training strategy, including the initial version of an LLM family in the training data. This enables the detector to generalize more effectively to subsequent, unseen updates of the same LLM family, without requiring frequent retraining.

To validate this strategy, we extended our experiments to include both Text Fluoroscopy (Yu et al., 2024a) and the Roberta-based detector (Liu et al., 2019). Details of experiment settings are shown in Appendix D. In each case, we incorporated the initial version of an LLM family into the training dataset and evaluated performance on later versions of that family. For example, for the GPT-4o family, we selected GPT-4o (2024-05-13) as the initial version and added its data to the training set. We then tested the detector's generalization on multiple subsequent versions: GPT-4o-mini (2024-07-18), GPT-4o (2024-08-06), GPT-4o (2024-11-20), and GPT-4o (latest). This setup allowed us to evaluate the model's ability to generalize across updates directly.

As shown in the Table 4, both supervised detectors demonstrated improved generalization after including the initial LLM version in training. Notably, for the GPT-4 family, adding data from GPT-4 (2023-06-13) led to substantial gains. For the GPT-4 family, after only adding data from the initial version (GPT-4-2023-06-13), the performance on GPT-4-2023-11-06 increased from 0.8117 to 0.8633, and on GPT-4-2024-01-25, it improved from 0.8061 to 0.8549. Therefore, when developing detectors, it is essential to consider model updates. To address this, it is crucial to at least include the initial version of each LLM family in the training data. This can improve performance on future unseen updates of the same LLM family without the need for frequent retraining.

5 Conclusion

In this paper, we introduce EvoBench, a novel benchmark for evaluating the generalization of LLM-generated text detection methods across evolving LLMs. EvoBench defines two key dimensions of LLM evolution: (1) updates made by LLM publishers over time and (2) developments carried out by developers, ensuring a comprehensive understanding of how evolving LLMs impact detection performance. To quantify this evolving dimension generalization, we propose the EMG (Evolving Model Generalization) metric.

We evaluate 14 widely used detection methods using EvoBench, revealing their vulnerabilities when facing evolving LLMs. In response, to improve the generalization of zero-shot methods across developing LLMs, we propose two strategies: for zero-shot methods, we suggest pruning the scoring model to extract shared features across LLM developments; for supervised methods, we recommend augmenting training data with data from LLM updates. Our benchmark represents a key step towards bringing detection methods into real-world scenarios for evolving LLMs. We also envision continuously updating this benchmark to cover a broader range of evolving LLMs, enabling more comprehensive evaluations across various domains and evolving LLMs.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China under Grant 62472398 and U2336206, and by the Shanghai Artificial Intelligence Laboratory.

Limitations

Although we have thoroughly considered the three key dimensions of generalization toward real-world scenarios, certain limitations remain that warrant further exploration in future research. First, it is important to emphasize that EvoBench is not intended to replace existing benchmarks for evaluating the generalization of LLM-generated text detection methods but rather to complement them. Therefore, EvoBench primarily focuses on three key dimensions in real-world scenarios: domains, generation strategies, and evolving models. We have not covered other aspects of generalization, such as language. A potential direction for future research is to extend EvoBench to incorporate additional dimensions of generalization.

Second, robustness has not been addressed in this study, as our focus was mainly on detection generalization. However, EvoBench could serve as a foundation for inspiring current robustness research, as robustness also involves tasks such as rewriting text using different LLMs.

References

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

Alim Al Ayub Ahmed, Ayman Aljabouh, Praveen Kumar Donepudi, and Myung Suh Choi. 2021. Detecting fake news using machine learning: A systematic literature review. *arXiv preprint arXiv:2102.04458*.

AI Anthropic. 2024. Claude 3.5 sonnet model card addendum. *Claude-3.5 Model Card*, 3(6).

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations*.

Amrita Bhattacharjee, Tharindu Kumarage, Raha Moraffah, and Huan Liu. 2023. ConDA: Contrastive domain adaptation for AI-generated text detection. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 598–610, Nusa Dua, Bali. Association for Computational Linguistics.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Canyu Chen and Kai Shu. 2023. Combating misinformation in the age of llms: Opportunities and challenges. *arXiv preprint arXiv: 2311.05656*.

Hong Chen, Chengtao Lv, Liang Ding, Haotong Qin, Xiabin Zhou, Yifu Ding, Xuebo Liu, Min Zhang, Jinyang Guo, Xianglong Liu, and Dacheng Tao. 2024a. DB-LLM: Accurate dual-binarization for efficient LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8719–8730, Bangkok, Thailand. Association for Computational Linguistics.

Jiaqi Chen, Xiaoye Zhu, Tianyang Liu, Ying Chen, Xinhui Chen, Yiwen Yuan, Chak Tou Leong, Zuchao Li, Tang Long, Lei Zhang, et al. 2024b. Imitate before detect: Aligning machine stylistic preference for machine-revised text detection. *arXiv preprint arXiv:2412.10432*.

Jon Christian. 2023. Cnet secretly used ai on articles that didn't disclose that fact, staff say. *Futurusm, January*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115.

DaYou Du, Yijia Zhang, Shijie Cao, Jiaqi Guo, Ting Cao, Xiaowen Chu, and Ningyi Xu. 2024. BitDistiller: Unleashing the potential of sub-4-bit LLMs via self-distillation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 102–116, Bangkok, Thailand. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. In *Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics (ACL).

Josh A Goldstein, Girish Sastry, Micah Musser, Renee DiResta, Matthew Gentzel, and Katerina Sedova. 2023. Generative language models and automated influence operations: Emerging threats and potential mitigations. *arXiv preprint arXiv:2301.04246*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Hanxi Guo, Siyuan Cheng, Xiaolong Jin, Zhuo Zhang, Kaiyuan Zhang, Guanhong Tao, Guangyu Shen, and Xiangyu Zhang. 2024. Biscope: Ai-generated text detection by checking memorization of preceding tokens. *Advances in Neural Information Processing Systems*, 37:104065–104090.

Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2024. Mgtbench: Benchmarking machine-generated text detection. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 2251–2265.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. 2023. Radar: Robust ai-text detection via adversarial learning. *Advances in Neural Information Processing Systems*, 36:15077–15095.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018. A dataset of peer reviews (PeerRead): Collection, insights and NLP applications. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1647–1661, New Orleans, Louisiana. Association for Computational Linguistics.

Davinder Kaur, Suleyman Uslu, Kaley J Rittichier, and Arjan Durresi. 2022. Trustworthy artificial intelligence: a review. *ACM Computing Surveys (CSUR)*, 55(2):1–38.

Tobias Kerner. 2024. Domain-specific pretraining of language models: A comparative study in the medical field. *arXiv preprint arXiv:2407.14076*.

Sanghoon Kim, Dahyun Kim, Chanjun Park, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, Changbae Ahn, Seonghoon Yang, Sukyung Lee, Hyunbyung Park, Gyoungjin Gim, Mikyoung Cha, Hwalsuk Lee, and Sunghun Kim. 2024. SOLAR 10.7B: Scaling large language models with simple yet effective depth up-scaling. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 23–35, Mexico City, Mexico. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Sebastian Kula and Michal Gregor. 2024. Multilingual models for check-worthy social media posts detection. *arXiv preprint arXiv:2408.06737*.

Wai-Chung Kwan, Xingshan Zeng, Yufei Wang, Yusen Sun, Liangyou Li, Yuxin Jiang, Lifeng Shang, Qun Liu, and Kam-Fai Wong. 2024. M4LE: A multi-ability multi-range multi-task multi-domain long-context evaluation benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15568–15592, Bangkok, Thailand. Association for Computational Linguistics.

Jooyoung Lee, Thai Le, Jinghui Chen, and Dongwon Lee. 2023. Do language models plagiarize? In *Proceedings of the ACM Web Conference 2023*, pages 3637–3647.

Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. 2021. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, 461:370–403.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Muneer M Alshater. 2022. Exploring the role of artificial intelligence in enhancing academic performance: A case study of chatgpt. *Available at SSRN*.

Dominik Macko, Robert Moro, Adaku Uchendu, Jason Lucas, Michiharu Yamashita, Matúš Pikuliak, Ivan Srba, Thai Le, Dongwon Lee, Jakub Simko, et al. 2023. Multitude: Large-scale multilingual machine-generated text detection benchmark. In *Proceedings* *of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9960–9987.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft.

Alex Mitchell. 2022. Professor catches student cheating with chatgptl: 'i feel abject terror'. Retrieved from https://nypost.com/2022/12/26/students-using-chatgpt-to-cheat-professor\-warns/. Accessed on February 17, 2023.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning*, pages 24950–24962. PMLR.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

OpenAI. 2022a. ChatGPT. https://chat.openai.com/.

OpenAI. 2022b. Introducing chatgpt.

OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*.

Mary Louise Kelly Patrick Wood. 2023. 'everybody is cheating': Why this teacher has adopted an open chatgpt policy. Retrieved from https://www.npr.org/2023/01/26/1151499213/chatgpt-ai-education-cheating-classroom\-wharton-school. Accessed on January 26, 2023.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Jinyan Su, Terry Zhuo, Di Wang, and Preslav Nakov. 2023. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12395–12412.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. 2024. A survey on self-evolution of large language models. *arXiv preprint arXiv:2404.14387*.

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

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

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, Alham Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024a. M4GT-bench: Evaluation benchmark for black-box machine-generated text detection. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3964–3992, Bangkok, Thailand. Association for Computational Linguistics.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, et al. 2024b. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1369–1407.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations*.

Xianjun Yang, Wei Cheng, Yue Wu, Linda Ruth Petzold, William Yang Wang, and Haifeng Chen. Dna-gpt: Divergent n-gram analysis for training-free detection of gpt-generated text. In *The Twelfth International Conference on Learning Representations*.

Xiao Yu, Kejiang Chen, Qi Yang, Weiming Zhang, and Nenghai Yu. 2024a. Text fluoroscopy: Detecting llm-generated text through intrinsic features. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15838–15846.

Xiao Yu, Yuang Qi, Kejiang Chen, Guoqiang Chen, Xi Yang, Pengyuan Zhu, Xiuwei Shang, Weiming Zhang, and Nenghai Yu. 2024b. Dpic: Decoupling prompt and intrinsic characteristics for llm generated text detection. *Advances in Neural Information Processing Systems*, 37:16194–16212.

Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. 2022. Wordcraft: story writing with large language models. In *27th International Conference on Intelligent User Interfaces*, pages 841–852.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*.

A Examples of Texts Generated by Different LLMs

In this section, we provided examples of responses from different updates of the same LLMs to the same question, offering a visual representation of how changes in versions lead to variations in the model's output in Table 5 and Table 6.

B Additional Experiment

Additionally, to eliminate the influence of output diversity on detection performance, we first generated 150 texts from each model for every dataset, resulting in a total of 300 texts. Additionally, we conducted repeated experiments with several models to ensure that output diversity does not significantly impact changes in detection difficulty. By doing so, we ensure that the variations in detection difficulty primarily stem from model evolution itself, rather than biases introduced by the diversity of the generated texts. The results of this experimental process are presented in Table 7, where we highlight the potential range of biases caused by the diversity of generated texts. This helps further to understand the relationship between model evolution and detection performance.

C Details of Evolving LLMs

The details of evolving LLMs are shown in Table 8.

D Details of Optimizing Strategies

In this section, we provide the details of two optimizing strategies. For the zero-shot detection method, we chose to optimize the Fast-DetectGPT detector. We use the pruned model 'princeton-nlp/Sheared-LLaMA-2.7B-Pruned' to replace the scoring model 'meta-llama/Llama-2-7b-hf' model as the scoring model. For the supervised detector, we chose to optimize the Text Fluoroscopy. Following Text Fluoroscopy, we used the first 200 entries of the open-source Human-ChatGPT Comparison Corpus (HC3) (Guo et al., 2023) dataset collected by previous researchers as a training set. The ratio for splitting the training and validation is $8:1$. We use a specific LLM of the LLM families to regenerate machine text in the training dataset and then use the new dataset and the original dataset to retrain the detector. Specifically, we regenerated this part of the training set using GPT-4-0613 and tested the results on the GPT-4 family. For the GPT-4o family, we used GPT-4o-2024-05-13, and for the

Table 5: An example of content generated by different versions of the GPT-4o and GPT-4o-mini

Does airway surgery lower serum lipid levels in obstructive sleep apnea patients?	
Human Answer	Surgery may improve blood lipid levels in patients with OSA, especially in patients with preoperative dyslipidemia, potentially yielding a major benefit in metabolism and cardiovascular sequelae. Prospective studies should examine this ...
GPT-4o 2024-05-13	Airway surgery for obstructive sleep apnea (OSA) has been shown to reduce serum lipid levels in some patients. The improvement is likely due to better oxygenation and reduced inflammation following successful surgery, ...
GPT-4o 2024-08-06	Airway surgery can potentially lower serum lipid levels in obstructive sleep apnea (OSA) patients. Surgical interventions that alleviate OSA symptoms may improve metabolic profiles, including lipid levels, due to reduced hypoxia and ...
GPT-4o 2024-11-20	Yes, airway surgery in obstructive sleep apnea (OSA) patients may lower serum lipid levels. Studies suggest that by improving oxygenation, reducing inflammation, and addressing the metabolic impacts of OSA, surgeries like ...
GPT-4o Latest	Yes, airway surgery for obstructive sleep apnea (OSA) can lower serum lipid levels. By improving airway patency and reducing hypoxia, surgery may enhance lipid metabolism and decrease systemic inflammation, which are linked ...
GPT-4o-mini 2024-07-18	Airway surgery in obstructive sleep apnea patients may lead to improvements in overall health and potentially lower serum lipid levels. However, the extent of lipid level changes can vary among individuals and ...

Table 6: An example of content generated by different versions of the Claude-Haiku and Calude-Opus

Neck pain treatment with acupuncture: does the number of needles matter?	
Human Answer	For this pathology, the number of needles, 5 or 11, seems not to be an important variable in determining the therapeutic effect when the time of stimulation is the same in the two groups.
Claude-Haiku 2024-03-07	Acupuncture for neck pain can be effective, but the number of needles used may not be the most critical factor. The placement and selection of acupuncture points tailored to the individual's condition are more important ...
Claude-Haiku 2024-10-22	Research suggests that the number of acupuncture needles may not significantly impact neck pain treatment. Effectiveness depends more on precise needle placement, practitioner skill, and individual patient response. Further studies are needed to definitively determine ...
Claude-Opus 2024-02-29	The number of needles used in acupuncture for neck pain treatment may impact its effectiveness. However, factors such as needle placement, depth of insertion, and stimulation technique are likely more important than the specific number ...

Claude-Haiku family, we used Claude-3-sonnet-20240229. we use `gte-Qwen1.5-7B-instruct`[6] as the encoder and the classifier consists of three fully connected layers with Tanh function. The dimensions of the intermediate layers in the classifier are 1024 and 512, respectively. The batch size is set to 16, and Adam (Kingma and Ba, 2014) optimizer is employed with an initial learning rate of $3e-3$.

[6]https://huggingface.co/Alibaba-NLP/gte-Qwen1.5-7B-instruct

E Data Collection.

In this section, we describe the process of data collection. Most of these datasets were generated following the dataset construction methodology

	Fast-DetectGPT	RADAR	Text-Fluoroscopy	Imitate Before Detect	RoBERTa-base	RoBERTa-large	Likelihood	Rank	LogRank	Entropy	DetectGPT	LRR	NPR	DNA-GPT
GPT-3.5-turbo-2024-01-25	0.8301	0.8280	0.8904	0.8992	0.6621	0.6723	0.7701	0.6432	0.7663	0.4163	0.4910	0.7098	0.6072	0.8160
	0.8420	0.8319	0.8917	0.9011	0.6629	0.6696	0.7822	0.6511	0.7787	0.4118	0.5008	0.7120	0.6276	0.8266
	(+**1.19%**)	(+**0.39%**)	(+**0.13%**)	(+**0.19%**)	(+**0.08%**)	(−**0.26%**)	(+**1.21%**)	(+**0.78%**)	(+**1.24%**)	(−**0.44%**)	(+**0.98%**)	(+**0.22%**)	(+**2.05%**)	(+**1.06%**)
GPT-4o-mini-2024-07-18	0.8053	0.7954	0.8800	0.8831	0.5539	0.5799	0.7238	0.6474	0.7133	0.4617	0.5374	0.6341	0.6045	0.7255
	0.8033	0.7976	0.8777	0.8886	0.5566	0.5827	0.7264	0.6494	0.7154	0.4543	0.5417	0.6356	0.6101	0.7291
	(−**0.20%**)	(+**0.22%**)	(−**0.23%**)	(+**0.55%**)	(+**0.27%**)	(+**0.28%**)	(+**0.26%**)	(+**0.20%**)	(+**0.21%**)	(−**0.74%**)	(+**0.43%**)	(+**0.15%**)	(+**0.56%**)	(+**0.36%**)
GPT-4-turbo-2024-04-09	0.7644	0.7459	0.8535	0.8534	0.5053	0.5146	0.6959	0.6315	0.6856	0.4526	0.5229	0.6182	0.5982	0.7114
	0.7661	0.7595	0.8504	0.8559	0.5034	0.5099	0.6953	0.6278	0.6848	0.4533	0.5208	0.6109	0.5925	0.6952
	(+**0.17%**)	(+**1.36%**)	(−**0.31%**)	(+**0.25%**)	(−**0.18%**)	(−**0.47%**)	(−**0.06%**)	(−**0.37%**)	(−**0.08%**)	(+**0.07%**)	(−**0.20%**)	(−**0.72%**)	(−**0.57%**)	(−**1.63%**)

Table 7: We selected several LLMs and produced LLM-generated text many times, and then conducted inspections to evaluate the impact of diversity on the detector.

used in DetectGPT and Fast-detectGPT, which are widely recognized in the literature on AI-generated text detection. This setting minimizes the influence of factors, other than model evolution, on the detection performance.

To assess the domain generalization capability of detection methods, we included the following datasets: Writing, PubMed, Community, and PeerRead. We also incorporated three different generation paradigms: continuation, question-answering, and paraphrasing. Specifically, the prompts used for generating each dataset are as follows:

- **Xsum**: "Please write an article with about 150 words starting exactly with: <prefix>"
- **Writing**: "Please write an article with about 150 words starting exactly with: <prefix>"
- **PubMed**: "Please answer the question in about 50 words: <question>"
- **SocialMedia**: "Generate text similar to the input social media text but using different words and sentence composition: <Full text>"
- **PeerRead**: "Please write a peer review with about 150 words starting exactly with: <prefix>"

F Experimental Setup Details

In this section, we provide a detailed description of the experimental setup. Specifically, for zero-shot, we follow Fast-DetectGPT (Bao et al.) and use GPT-J-6B (Wang and Komatsuzaki, 2021) and GPT-Neo-2.7B (Black et al., 2021) as the scoring models for zero-shot methods. For supervised detectors, we used the pre-trained detectors provided by the authors. All experiments are conducted on a workstation equipped with 4 NVIDIA A100 GPUs.

For the valuation metric, we measure the detection performance using two metrics: AUROC(the area under the receiver operating characteristic) and EMG (evolving model generalization). A higher AUROC value indicates better detection quality, while a higher EMG metric indicates better generalization ability across evolving LLMs.

G The explanation of EMG

In this section, we provide an explanation of EMG. A positive EMG value indicates that the detection performance improves as the LLM evolves, while negative values reflect a decline in generalization. The magnitude of the EMG value also reflects the degree of improvement or decline in generalization; smaller values indicate worse stability. EMG reflects the direction and degree of fluctuations. However, it does not directly indicate the actual detection performance regarding AUROC.

For example, as shown in Table 2, in the GPT-4o family, the EMG values for RADAR and Text-Fluoroscopy are -0.537% and -0.493%, respectively. However, as shown in Table 1, the AUROC values for RADAR range from 0.8416 to 0.8694, while those for Text-Fluoroscopy fluctuate between 0.75426 to 0.79618. Although the EMG values are similar, the AUROC differs significantly, suggesting that both methods have comparable generalization abilities in the face of evolving LLMs, but their actual detection performance varies. Therefore, we recommend evaluating both AUROC and EMG together.

Table 8: Details of evolving LLMs.

Evolving LLMs	**Version Time**	**Source**
GPT-4o	2024-05-13	gpt-4o-2024-05-13
GPT-4o	2024-08-06	gpt-4o-2024-08-06
GPT-4o	2024-11-20	gpt-4o-2024-11-20
GPT-4o	Latest	chatgpt-4o-latest
GPT-4o-mini	2024-07-18	gpt-4o-mini-2024-07-18
GPT-4	2023-06-13	gpt-4-0613
GPT-4	2023-11-06	gpt-4-1106-preview
GPT-4	2024-01-25	gpt-4-0125-preview
GPT-4	2024-04-09	gpt-4-turbo-2024-04-09
Claude-Sonnet	2024-02-29	claude-3-sonnet-20240229
Claude-Sonnet	2024-06-20	claude-3-5-sonnet-20240620
Claude-Sonnet	2024-10-22	claude-3-5-sonnet-20241022
Claude-Haiku	2024-03-07	claude-3-haiku-20240307
Claude-Haiku	2024-10-22	claude-3-5-haiku-20241022
Claude-Opus	2024-02-29	claude-3-opus-20240229
Qwen	Qwen1.5-7B	Qwen/Qwen1.5-7B-Chat
Qwen	Qwen2-7B	Qwen2-7B-Instruct
Qwen	Qwen2.5-7B	Qwen/Qwen2.5-7B-Instruct
LlaMa3	Llama-3.1-8B	meta-llama/Meta-Llama-3.1-8B-Instruct
LlaMa3	Llama-3.1-70B	meta-llama/Meta-Llama-3.1-70B-Instruct
LlaMa3	Llama-3.2-1B	meta-llama/Meta-Llama-3.2-1B-Instruct
LlaMa3	Llama-3.2-3B	meta-llama/Meta-Llama-3.2-3B-Instruct
LlaMa3	Llama-3.3-70B	meta-llama/Meta-Llama-3.3-70B-Instruct
Fine-tuning	LLaMA-2-7B-chat-hf	meta-llama/Llama-2-7b-chat-hf
Fine-tuning	Vicuna-1.5-7B	lmsys/vicuna-7b-v1.5
Fine-tuning	Wizardmath-7B	WizardLMTeam/WizardMath-7B-V1.0
Pruning	Sheared-LLaMA1.3B	princeton-nlp/Sheared-LLaMA-1.3B
Pruning	Sheared-LLaMA1.3B-pruned	princeton-nlp/Sheared-LLaMA-1.3B-Pruned
Pruning	Sheared-LLaMA2.7B-pruned	princeton-nlp/Sheared-LLaMA-2.7B
Pruning	Sheared-LLaMA2.7B	princeton-nlp/Sheared-LLaMA-2.7B-Pruned