Who Wrote the Book? Detecting and Attributing LLM Ghostwriters

Anudeex Shetty[1], **Qiongkai Xu**[1,2], **Olga Ohrimenko**[1], **and Jey Han Lau**[1]
[1]School of Computing and Information Systems, The University of Melbourne, Australia
[2]School of Computing, FSE, Macquarie University, Australia
{anudeex, oohrimenko, laujh}@unimelb.edu.au,
qiongkai.xu@mq.edu.au

Abstract

Out-of-distribution (OOD) generalisation is essential for authorship attribution (AA), especially when involving new authors, yet this aspect remains insufficiently supported by existing AA datasets. In this paper, we introduce GHOSTWRITEBENCH, a dataset for LLM AA task, designed to test generalisation across multiple OOD dimensions, including unseen domains and authors. GHOSTWRITEBENCH comprises long-form texts (avg. 49K words per book) generated by ten frontier LLMs. We find that existing methods degrade substantially in unseen author scenarios — dropping by as much as 70% even under high-resource settings. We propose TRACE, a novel lightweight and interpretable fingerprinting method that captures token-level transition patterns (i.e., word rank and entropy) using a lightweight language model. Experiments show robustness of TRACE in OOD settings and establish it as a strong baseline on GHOSTWRITEBENCH.[1]

1 Introduction

Recently, a large number of AI detection datasets have been proposed (see Table 1). However, several limitations remain. First, most datasets frame detection as binary human vs. AI classification (He et al., 2024; Dugan et al., 2024), with only a few (La Cava and Tagarelli, 2025; Uchendu et al., 2021) supporting the more challenging authorship attribution (AA) task (i.e., multi-class model classification). Second, they rarely evaluate generalisation (e.g., by including text generated by unseen generators or text from other domains).[2] Finally, existing datasets focus on short texts, even though we are beginning to see the emergence of LLM-ghostwritten[3] books (The Authors Guild, 2024; Australian Society of Authors, 2025). In addition to these dataset limitations, existing attribution methods (La Cava and Tagarelli, 2025; Guo et al., 2024; Tyo et al., 2023; Fabien et al., 2020) typically rely on supervised training with known authors and exhibit limited generalisation across domains and unseen models.

We approach these challenges by (i) constructing a new dataset, GHOSTWRITEBENCH, for AA with multiple OOD dimensions and (ii) proposing a novel fingerprinting approach, TRACE, that is lightweight and has better generalisation capabilities. GHOSTWRITEBENCH supports two OOD dimensions: OOD-Domain and OOD-Author, which cover genre shift and unseen LLM author evaluation, respectively. GHOSTWRITEBENCH primarily focuses on long-form texts (i.e., books) generated by current frontier LLMs (on average over 49K words per document) to reflect an emerging form of AI content generation: AI ghostwriting (The Authors Guild, 2024; Australian Society of Authors, 2025). Our experiments show that existing attribution methods struggle under the OOD-Author setting, with strong methods suffering drops of up to 90% in low-resource and 70% in high-resource training scenarios, highlighting limited generalisation in prior work.

Next, we introduce a novel detection technique, **T**ransition-based **R**epresentation for **A**uthorship **C**lassification and **E**valuation (TRACE), for LLM attribution. TRACE first uses a (small) set of training texts generated by an LLM to create its fingerprint. At test time, to determine whether the test instance is generated by the LLM, we first create the test fingerprint and assess its similarity to the

[1]The dataset and code can be found at https://github.com/anudeex/TRACE.

[2]In this paper, we target specifically non-adversarial OOD generalisation (e.g., unseen authors and domains) rather than adversarial robustness (e.g., against obfuscation attacks (Dugan et al., 2024)).

[3]Ghostwriting occurs when a writer (a.k.a *ghostwriter*) produces content on someone's behalf without authorship credit. LLM-ghostwritten books are typically low-quality sham books designed to deceive customers and take sales away from legitimate books.

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 57419–57447
October 24-29, 2026 ©2026 Association for Computational Linguistics

Dataset	Task Type	LLM Generators		OOD		Long Form	Avg. Length	Num. Docs
		Num.	Frontier†	Domain	Author			
TURINGBENCH(2021)	Multi	20	✗	✗	✗	✗	<200	160K
MGTBENCH(2024)	Binary	6	✗	✗	✗	✗	<500	21K
RAID(2024)	Binary	11	✗	✓	✗	✗	<250	500K
M4GT-BENCH(2024)	Multi	6	✗	✓	✗	✗	<500	87K
OPENTURING(2025)	Multi	7	✗	✓	✗	✗	<500	497K
GHOSTWRITEBENCH(Ours)	Multi	10	✓	✓	✓	✓	49K	325

†Frontier LLMs refer to recent state-of-the-art models (e.g., `gpt-5.1`, `gpt-4.1`, etc.).

Table 1: Overview of existing datasets for LLM-generated text attribution. Most prior work focuses on binary classification rather than multi-class authorship attribution (AA) tasks. LLMs included in these benchmarks are largely outdated (i.e., they are small-scale, 7B–10B parameters, models and/or not the latest frontier models), as indicated by the 'Frontier' column. While some studies include out-of-distribution (OOD) evaluation, this is rarely considered for AA tasks. 'Author' OOD denotes open-set evaluation where authors unseen during training are evaluated. 'Long Form' refers to documents longer than 10K words. We report only AI-generated texts for 'Avg. Length' (in words) and 'Num. Docs'.

LLM's fingerprint. To produce the fingerprint for a text (during training or inference), we examine the transitions of token-level patterns (e.g., rank of a word) estimated by another lightweight language model in the word sequence to create a compressed two-dimensional signature (see Figure 1). Our technique does not require white-box access and operates post hoc (and so works for both open- and closed-source LLMs). It requires minimal training data (i.e., only a small set of LLM-generated texts), and is effective for detecting text from new domains or generated by unseen authors (LLM or human).

To summarise, our contributions are:

- We introduce GHOSTWRITEBENCH, a new dataset for LLM authorship attribution that supports long-form book detection and explicitly tests generalisation across two OOD dimensions.
- We propose TRACE, a lightweight and interpretable fingerprinting technique for LLM authorship attribution. It achieves strong performance in-domain, OOD, and in low-resource training scenarios, establishing a strong baseline on GHOSTWRITEBENCH.

2 Related Work

2.1 Authorship Attribution

Authorship attribution (AA) is a long-standing field (Mendenhall, 1887; Mosteller and Wallace, 1963) that has grown with the rise of digital texts and advances in machine learning (Tyo et al., 2023; Stamatatos, 2009; Neal et al., 2017). It is usually formulated as a multi-class classification problem, with each class representing an author. Current SOTA methods rely on neural networks, often using Transformer-based backbones fine-tuned on domain-specific corpora (Tyo et al., 2023; Fabien et al., 2020). Despite their effectiveness, these approaches are resource-intensive, require careful dataset curation, generalise poorly to new domains, and struggle with long documents due to standard Transformer (Vaswani et al., 2017) context limits (e.g., 512 tokens).

2.2 LLM-Generated Text

As LLMs are increasingly used to generate misinformation, provenance and veracity have become pressing (Kumarage et al., 2024; Huang et al., 2025). Detecting LLM-generated text is non-trivial: humans often perform only slightly above chance (Dou et al., 2022; Uchendu et al., 2021). Most prior work (Mitchell et al., 2023; Yang et al., 2024; Wu et al., 2025) focuses on binary (human vs. machine) classification, even though attribution to a specific model is important for accountability. Existing SOTA methods (La Cava and Tagarelli, 2025; Guo et al., 2024) largely mirror traditional AA approaches, inheriting limitations such as poor generalisation and lack of interpretability. Some works have explored cross-domain evaluation (Dugan et al., 2024; He et al., 2024; Wang et al., 2024). However, they consider only a limited set of authors and exclude long-form LLM-generated text. Most assume a closed set of authors, whereas practical attribution often occurs in open-set scenarios (Stolerman et al., 2014; Huang et al., 2025), which is the setting that GHOSTWRITEBENCH assumes.

Model	Total Books	Avg. # Words	Text Quality and Diversity				
			PPL (↓)	S-B (↓)	S-R (↓)	NGD (↑)	CR (↓)
deepseek-chat	42	33.4K	21.62	0.001	8.22	2.29	2.96
glm-4.6	32	28.4K	33.06	0.002	7.69	2.46	2.90
kimi-k2	37	37.1K	27.87	0.002	8.76	2.24	2.81
qwen3-235b	38	67.1K	21.30	0.004	10.04	2.18	2.72
qwen3-max	34	41.2K	24.96	0.003	9.01	2.29	2.73
claude-sonnet	24	47.1K	18.89	0.005	8.38	2.37	2.88
gemini-flash	30	20.5K	24.18	0.003	7.68	2.37	2.88
gemini-pro	27	88.1K	22.10	0.004	9.89	2.06	2.71
gpt-4.1	35	66.1K	40.59	0.002	8.53	2.44	2.58
gpt-5.1	26	69.3K	31.67	0.005	9.04	2.33	2.64
Human	100	88.0K	31.36	0.001	10.25	2.14	2.65

Table 2: GHOSTWRITEBENCH statistics: total books (genre splits in Appendix Table 10), average number of words in book, and text quality and diversity metrics for each LLM (and 100 random human books). Metrics include perplexity (PPL), self-BLEU (S-B), repetition (S-R), n-gram diversity (NGD), and compression ratio (CR) (more details in Appendix A.2). Arrows (↑; ↓) indicate whether higher or lower values are better.

LLMs have also fostered ghostwriting, producing content on someone's behalf without authorship credit, contributing to low-quality or sham books impacting authors (The Authors Guild, 2024; Australian Society of Authors, 2025). Motivated by these developments and methodological limitations, our work focuses on improving the generalisation of attribution techniques for LLM-generated long-form documents such as books.[4]

3 GHOSTWRITEBENCH Dataset

In this paper, we focus on the task of LLM AA, i.e., determining which LLM generated a given document. As shown in Table 1, most existing datasets of LLM-generated text focus on constrained settings, i.e., (i) framing the task as binary detection, (ii) comprising outdated models, (iii) without evaluating OOD generalisation, and (iv) focusing on short texts. Motivated by these limitations, we construct a new dataset, GHOSTWRITEBENCH, covering ten recent frontier LLMs (model details in Appendix Table 9) and comprising books (both fiction and non-fiction) as long-form documents[5] supporting multiple OOD dimensions.

3.1 Book Generation Pipeline

Humans typically write a book through a hierarchical process involving planning, drafting, and revision (Sun et al., 2022; Lee et al., 2025; Hayes and Flower, 2016). We mimic this process by designing a book generation pipeline with incremental steps: outlining, summarisation, and long-text expansion, following prior works (Yao et al., 2019; Wang et al., 2025; Alhussain and Azmi, 2021).

For each book, we first generate a detailed outline conditioned on some constraints (e.g., era and genre); this outline produces the narrative structure, plot, and characters. And then iteratively over multiple turns, we instruct the LLM to produce the next text segment, conditioned on the outline, the previous segment, and the running summary (prompts and more details in Appendix A.1). We provide real, human-authored texts (sourced from Project Gutenberg[6]) as few-shot reference examples to guide the LLM generation to maximise generation quality (constraints such as era and genre are therefore drawn from these Gutenberg books). This pipeline ensures that narrative coherence and discourse structure are preserved, enabling the generation of full-length books.[7] Importantly, all books were generated using the same generation pipeline across all LLMs, ensuring no pipeline-specific bias. While some model-specific

[4] Although ghostwriting can occur in short texts, it mainly affects long-form documents, making books a practical focus.

[5] Our definition of long-form is document with more than 10K words.

[6] https://www.gutenberg.org/.

[7] Book generation statistics reported in Appendix Table 6.

generation biases are inevitable, we try to reduce the likelihood of dataset artefacts through careful data cleaning (Section 3.3). Table 2 provides various statistics for these LLM-generated books; we will return to discuss more on how we evaluate quality in Section 3.3.

3.2 OOD Dimensions

In addition to standard in-distribution (ID) evaluation, the GHOSTWRITEBENCH dataset examines two types of OOD generalisation: (i) **OOD-Domain**, to test generalisation to texts written in unseen genres; and (ii) **OOD-Author**, to test generalisation to texts generated by unseen LLMs. We construct per-author domain splits, where book genres are partitioned into ID and OOD sets for each author.[8] This creates a more challenging setting than using a single split shared across all authors, as commonly used in existing datasets (see Appendix Table 10 for genre splits). We use the genres defined by Momen et al. (2025) for Gutenberg books (see Table 5 for the list of genres[9]). In practice, a Gutenberg book often consists of multiple genres (e.g., LF (Literature & Fiction) + ST (Science & Technology), and many of the LLM-generated books are also multi-genre.

For OOD-Author, we create ten splits by holding out one LLM at a time (we have ten LLMs) and report average performance across these splits. That is, for a given split, the training partition contains books from nine LLMs and the test partition contains books from the held-out LLM to ensure there is no development-set leakage during thresholding.

To understand the impact of training data, within a particular evaluation setting (ID, OOD-Domain, OOD-Author), GHOSTWRITEBENCH also provides two training scenarios: **high-resource** and **low-resource**, where the former has a large number (10–35) and the latter a small number of training books (3–5) text–label pairs per LLM.[10]

3.3 Dataset Analysis

Data Cleaning. We sanitise the LLM-generated books by removing non-ASCII and smart characters, along with common LLM markers (e.g., END, START OF SEGMENT). As observed by Momen et al. (2025), the start and end pages of the books often contain identifier markers (such as author, title, year, and others). Therefore, we remove 1.5K tokens from the beginning and end of the text to prevent potential shortcuts by attribution methods. We also conduct manual inspection on a subset and perform n-gram analysis across the corpus to check for frequent boilerplate text (titles, generation markers, etc.) to filter any additional shortcuts.

Quality Analysis. We evaluate text quality using several metrics in Table 2. We measure perplexity (PPL; Jurafsky and Martin (2026)) of generated texts for different LLMs, a standard measure of language understanding. We then measure several lexical diversity metrics, comprising both inter- and intra-document measures. We compute self-BLEU (S-B; Zhu et al. (2018)), a homogenisation score over pairs of books generated by the LLM (since we lack references), which is commonly used in writing tasks to ensure diversity (Padmakumar and He, 2024). It is common for LLMs to repeat text in long-form generation (Bai et al., 2025); hence, we evaluate self-repetition (S-R; Salkar et al. (2022)), which captures this repetition tendency. Similarly, we measure the n-gram diversity score (NGD; (Meister et al., 2023)), which measures the number of unique n-grams in the texts generated by LLMs. Finally, a non-n-gram-based metric, compression ratio (CR; (Shaib et al., 2025)) evaluates compressibility (higher values suggest redundancy). Across all these diversity metrics, we observe that all LLMs in GHOSTWRITEBENCH have values similar to those of human books, quantitatively indicating comparable text quality.

In addition to quantitative analysis, we conduct a small-scale human evaluation on a subset of books (more details in Appendix A.3) with ten annotators. We assess quality across multiple dimensions, following prior work (Xia et al., 2025; Chhun et al., 2022). Appendix Table 7 reports the average annotator scores across different quality dimensions (see Appendix A.3 for definitions), both at the passage level and overall for each book. Overall, we observe strong scores (> 4; out of 5) for relevance, coherence, fluency, and diversity, complementing the above text quality and diversity analysis. Scores for other subjective dimensions (engagement, empathy, surprise, and complexity) are also generally positive (> 3.3; out of 5). These results indicate that the text in GHOSTWRITEBENCH exhibits strong relevance and is of overall good quality.

[8] We ensure there is no genre overlap between the ID and OOD-Domain partition per author; see Appendix Table 10.

[9] Recall that we use Gutenberg books as few-shot reference examples for the LLM. As such, the genres for the LLM-generated books are also grounded in Gutenberg genres.

[10] To ensure robustness of results, we create three random splits for each subset; see Appendix Table 18.

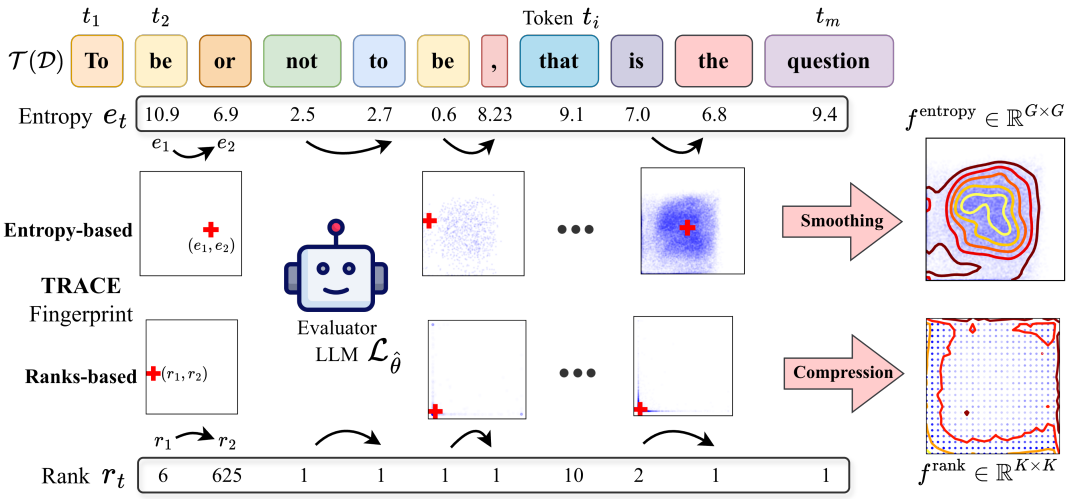


Figure 1: Overview of TRACE fingerprint construction. We model transitions of consecutive token scores (using another lightweight evaluator language model; `GPT-2` used in illustration). The process is shown for both Rank-based (§4.1) and Entropy-based (§4.2) fingerprint variant. We construct a pool of reference fingerprints from an LLM's training texts. At inference time, attribution is a straightforward comparison between the test fingerprint and the LLM reference fingerprint.

4 TRACE Fingerprint

We next describe **T**ransition-based **R**epresentation for **A**uthorship **C**lassification and **E**valuation (TRACE), a novel fingerprinting method for LLM attribution. At training time, we first construct reference fingerprints for each LLM (if an LLM has multiple books, we create one fingerprint for each book). Given a test sample, we then construct its fingerprint and determine whether it matches one of the reference fingerprints.[11]

As different LLMs may have different preferences when it comes to word selection during generation, our idea is to examine token-level transitions (as shown in Figure 1), such as word rank (Section 4.1) and entropy (Section 4.2), as computed by a separate, lightweight *evaluator* language model. Although capturing word preferences via a separate evaluator model is related to existing distributional heuristics (e.g., perplexity; Wu et al. (2025); Huang et al. (2025)), TRACE is fundamentally different in that it models *transitions* between consecutive tokens rather than static distributions, producing more discriminative signatures (as seen in Appendix Figures 10 and 12). To the best of our knowledge, modelling token preference transitions as fingerprints for multi-class LLM attribution has not been explored in prior work.

Our fingerprint TRACE can be applied to both open- and closed-source LLMs, since it does not require white-box access. It is lightweight and adaptable, and produces an interpretable 2D signature (more in Section 4.3). We next discuss these two variants of TRACE: Rank-based and Entropy-based.

4.1 Rank-Based

Formally, given a document (or book in our case) $\mathcal{D}$ represented as a token sequence $\langle t_1, t_2, \ldots, t_m \rangle$ tokenised by a tokenizer $\mathcal{T}$ with vocabulary $\mathcal{V}$. For each token $t_i \in \langle t_1, t_2, \ldots, t_m \rangle$, we compute its rank under an evaluator language model $\mathcal{L}_{\hat{\theta}}$ with context $t_{<i}$ as follows:

$$r_i = \left| \left\{ v \in \mathcal{V} : \mathbb{P}(t = v \mid t_{<i}; \hat{\theta}) \geq \mathbb{P}(t = t_i \mid t_{<i}; \hat{\theta}) \right\} \right|.$$

The raw fingerprint $\hat{f}^{\text{rank}} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ captures the counts of (r_{i-1}, r_i) (left side in Figure 1). This fingerprint matrix is very large and sparse, and so we compress it by binning ranks into equal probability clusters (as detailed in Appendix Algorithm 1), giving the final fingerprint $f^{\text{rank}} \in \mathbb{R}^{K\times K}$ where $K \ll |\mathcal{V}|$.

We apply this fingerprint construction process for each LLM using their training books (if an LLM has multiple books, one fingerprint is created for each book) to create reference fingerprints for each LLM. Then, for a test text, we construct a test fingerprint following the same process above. To determine which LLM generated the test text,

[11] Rather than performing majority voting over reference fingerprints, we retrieve the most similar reference fingerprint.

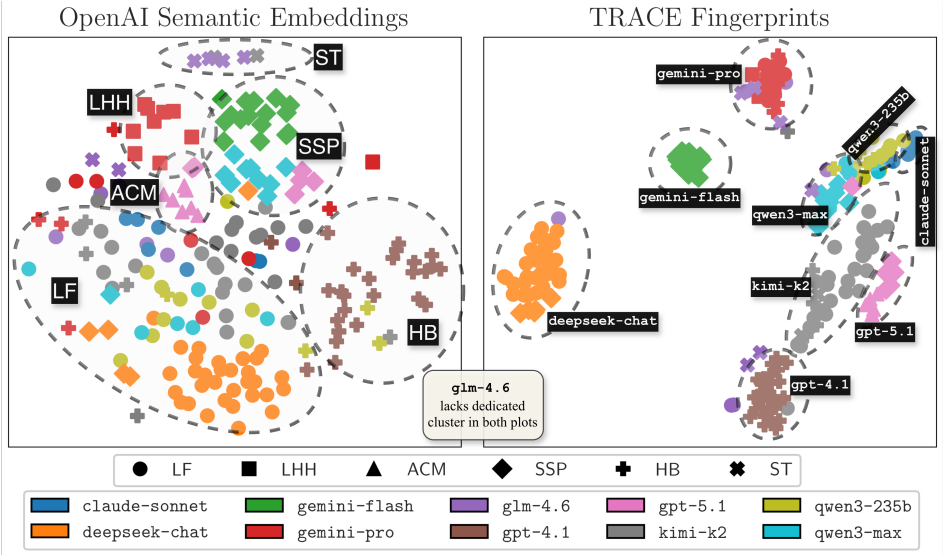


Figure 2: t-SNE visualisation comparing semantic embeddings (`openai/text-embedding-3-large`; left) and our Entropy-based TRACE fingerprints (right) on single-genre books in GHOSTWRITEBENCH. Marker shape denotes genre and colour denotes the generating LLM. The left plot shows content-based clustering by genre, while the right plot presents clusters by LLM across genres, indicating that TRACE captures model-specific stylistic patterns rather than genre-specific content features.

we identify the reference fingerprint that is most similar to the test fingerprint.

To measure similarity between two fingerprints, which can be interpreted as discrete probability distributions, we use the Jensen-Shannon (JS) distance (equation given in Appendix B.1).

4.2 Entropy-Based

Entropy has been used in NLP to capture various properties of natural language (Genzel and Charniak, 2002; Jaeger, 2010). Like word ranks, LLMs may exhibit unique entropy transition patterns and so constitute a potential signal for creating the fingerprint. We provide a qualitative example in Appendix Figure 3, illustrating the generalisation of Entropy-based fingerprints.

Formally, the entropy of a token t_i similar to setup in Rank-based is given by:

$$e_i = -\sum_{v \in \mathcal{V}} \mathbb{P}(t = v \mid t_{<i}; \hat{\theta}) \cdot \log \mathbb{P}(t = v \mid t_{<i}; \hat{\theta}).$$

Unlike word ranks, the resulting fingerprint is continuous rather than discrete (intuitively the fingerprint is like a scatter plot). To smooth it and produce a density map (of dimension G), we use kernel density estimation (detailed in Appendix Algorithm 2) to obtain $f^{\text{entropy}} \in \mathbb{R}^{G \times G}$ where G is the grid size of the density map. For fingerprint matching, in addition to JS distance (as used for ranks), we experiment with another similarity metric, norm-mean, which is computed by comparing the angle between surface normals (equation given in Appendix B.1).

4.3 Qualitative nature of TRACE fingerprints

As illustrated in Figure 1, we construct the fingerprint by modelling transitions between consecutive tokens. Each axis of the resulting 2D matrix represents the rank (or entropy) of consecutive token transitions as estimated by an evaluator language model, and the density in a given region reflects how frequently an LLM produces that transition type. For instance, high density in the bottom-left indicates a preference for high-probability tokens following other high-probability tokens (a more predictable generation style), while density in the top-right reflects a tendency toward more surprising token choices. Different LLMs produce visually distinct fingerprints; for example, GPT models exhibit a more circular structure (for Entropy-based; Appendix Figure 10) than Gemini models, which tend toward a squarer pattern.

We hypothesise that TRACE generalises because it relies on token-level transition patterns computed by an independent evaluator language model, rather than direct supervision on attribution labels (as done in prior methods). This is supported by the preliminary analysis in Figure 2, where the TRACE's signature for both ID and

OOD books from the same LLM cluster together, suggesting that TRACE captures LLM-specific stylistic characteristics rather than content; in contrast, semantic embeddings are genre-clustered. We next study this empirically with experiments on GHOSTWRITEBENCH.

5 Experimental Setup

Baselines. We consider techniques from different areas and categories as baselines. From metric-based methods, we evaluate RANK, ENTROPY, and GLTR (Gehrmann et al., 2019). Although originally designed for binary detection, we follow La Cava and Tagarelli (2025) to adapt them for multi-class classification (by learning a threshold for each LLM).

For model-based (supervised) methods, we consider N-GRAM (Koppel and Schler, 2004), BERT-AA (Fabien et al., 2020), TOPFORMER (Uchendu et al., 2024), and DETECTIVE (Guo et al., 2024), covering both non-neural and neural (including different backbones, representations, and learning paradigms) approaches. Tyo et al. (2023) found very strong performance for N-GRAM and BERT-AA for traditional human AA tasks. TOPFORMER is a ROBERTA-based model fine-tuned with a topological data analysis (TDA) layer specifically designed for the AA task, capturing distinctive writing styles. While DETECTIVE is a SOTA multi-level contrastive learning-based detector for AI-generated text with implicit author representation learning. Finally, while there are recent LLM-based detection methods (Hans et al., 2024; Yu et al., 2024; Verma et al., 2024), most are formulated for binary human-vs-machine detection and are therefore not directly applicable to the LLM attribution setting considered in this work. More details regarding baselines are provided in Appendix D.

Experimental Setting. We use `GPT-2` (smallest version; 124M parameters) (Radford et al., 2019) as the independent evaluator language model to estimate word ranks and entropy for creating the fingerprints. For the Rank-based fingerprint, we use a cluster size (K) of 50. For the Entropy-based fingerprint, we use a grid size (G) of 50. All hyperparameters are selected based on the development set (which is additional two ID genre books per LLM). Additional experimental details in Appendix C.

Evaluation. AA techniques are typically deployed in open-set settings, where the author of a test text may not appear among those seen during training (GHOSTWRITEBENCH captures this with OOD-Author evaluation). In such cases, a reliable technique should be well-calibrated, assigning high confidence to known authors and low confidence to unseen ones (i.e., rejection) (Scheirer et al., 2012; Guo et al., 2017). To achieve this, we calibrate all methods (baselines and our methods) on the development set to determine a confidence threshold that best balances rejection and attribution performance. All thresholds for different configurations can be found in Appendix Table 19. During evaluation, we apply this threshold across all scenarios: ID, OOD-Domain, and OOD-Author for fair comparison (in a practical application, we would not know whether a test text is ID or OOD, and so this thresholding would have to be applied at all times). For texts from known authors, predictions below the threshold are treated as false negatives (missed attributions); for unseen authors, such predictions correspond to correct rejection. AA performance is evaluated using the macro-F_1 metric, which assigns equal weight to all LLMs (Uchendu et al., 2020; Tyo et al., 2023). As discussed in Section 4, we use JS distance and norm-mean similarity metrics (more in Appendix B.1) for comparing TRACE fingerprints.

6 Experimental Results

6.1 Main Results

Table 3 presents the main results on GHOSTWRITEBENCH, comparing the TRACE fingerprint against baselines. We report results separately for low-resource and high-resource authors (recall from Section 3.2 that these settings control the amount of training books available per LLM); combined results can be found in Appendix Table 13. Although many baselines perform well in the ID setting, they experience substantial performance degradation (as high as 90% in low-resource and 70% in high-resource training settings) in our OOD setting, showing that these methods would not generalise well in a real-world application. In the more challenging scenarios: low-resource and the two OOD-Domain and OOD-Author settings, TRACE consistently outperforms both model- and metric-based baselines. A qualitative example is shown in Appendix Figure 3. That said, for some settings with more training data (high-resource), a few base-

Method	Low Resource			High Resource		
	ID	OOD-Domain	OOD-Author	ID	OOD-Domain	OOD-Author
RANK	0.00 ± 0.00	0.00 ± 0.00	0.40 ± 0.19	0.27 ± 0.08	0.25 ± 0.06	0.38 ± 0.21
ENTROPY	0.00 ± 0.00	0.00 ± 0.00	0.20 ± 0.20	0.33 ± 0.06	0.35 ± 0.04	0.10 ± 0.08
GLTR	0.00 ± 0.00	0.00 ± 0.00	0.44 ± 0.17	0.30 ± 0.02	0.31 ± 0.02	0.28 ± 0.03
N-GRAM	0.40 ± 0.16	0.20 ± 0.16	0.36 ± 0.31	$\underline{0.96 \pm 0.01}$	$\mathbf{0.92 \pm 0.04}$	0.28 ± 0.35
BERT-AA	0.44 ± 0.17	0.27 ± 0.19	0.04 ± 0.06	0.96 ± 0.04	$\underline{0.90 \pm 0.05}$	0.29 ± 0.05
TOPFORMER	0.51 ± 0.28	0.24 ± 0.17	0.29 ± 0.17	0.94 ± 0.04	0.88 ± 0.04	0.46 ± 0.27
DETECTIVE	0.53 ± 0.09	$\underline{0.27 \pm 0.19}$	0.40 ± 0.20	0.95 ± 0.03	0.85 ± 0.11	$\underline{0.49 \pm 0.22}$
TRACE (Ours)						
Ranks-JS	0.40 ± 0.24	0.22 ± 0.19	0.18 ± 0.03	0.78 ± 0.06	0.66 ± 0.12	0.20 ± 0.08
Entropy-JS	$\underline{0.60 \pm 0.16}$	0.23 ± 0.12	$\underline{0.49 \pm 0.14}$	0.91 ± 0.06	0.82 ± 0.08	$\mathbf{0.58 \pm 0.20}$
Entropy-Norm	$\mathbf{0.67 \pm 0.19}$	$\mathbf{0.49 \pm 0.11}$	$\mathbf{0.51 \pm 0.17}$	$\mathbf{0.96 \pm 0.02}$	0.82 ± 0.10	0.37 ± 0.03

Table 3: Results for attribution techniques on GHOSTWRITEBENCH under low-resource (left) and high-resource (right) training settings for different OOD dimensions (OOD-Domain and OOD-Author). Reported scores are macro-F_1 (mean $\pm$ std for three random splits; see Appendix Table 18); **bold** denotes the best result and underline the second best. Overall, TRACE has strong performance with several existing methods showing limited generalisation.

Method	OOD-Author	
	LLM	Human
N-GRAM	0.32 ± 0.33	$\underline{0.48 \pm 0.29}$
DETECTIVE	$\underline{0.45 \pm 0.15}$	0.16 ± 0.07
TRACE (Ours)		
Ranks-JS	0.19 ± 0.03	0.11 ± 0.01
Entropy-JS	$\mathbf{0.54 \pm 0.15}$	$\mathbf{1.00 \pm 0.00}$
Entropy-Norm	0.44 ± 0.09	$\mathbf{1.00 \pm 0.00}$

Table 4: Performance on the OOD-Author variant consisting of unseen human authors. TRACE achieves near-perfect performance, demonstrating its ability to distinguish human-written vs. LLM-generated texts.

lines marginally outperform our method. For completeness, we also report top-k accuracy results in Appendix Table 14 and without any thresholding (for ID and OOD-Domain) in Appendix Table 15, and note that the overall trend is the same as the main results (Table 3).

Unseen Human Texts. We also evaluate a variant of the OOD-Author setting, focusing on unseen *human* authors. We randomly sample Gutenberg books by 10 human authors and measure how well our method can identify these unseen human texts, reflecting the common binary classification task of LLM-generated vs. human-written. As shown in Table 4, the Entropy-based variants of TRACE achieve perfect performance in distinguishing human authors from LLM authors. These results indicate that the fingerprints of human authors are *very* different to those of LLM authors (also supported by visualisations in Appendix Figure 11), suggesting that TRACE can also be an effective detector in the more traditional binary classification setup.

Error Analysis. We examine the prediction confusion matrices (see Appendix Figure 9) for the top-performing methods: N-GRAM, DETECTIVE, and the Rank- and Entropy-based TRACE variants. We observe that baselines tend to confuse similar LLMs (i.e., those belonging to the same model family). This pattern is more pronounced in Appendix Table 16, where we assess OOD-Author in a different complementary manner: if the predicted class for the unseen LLM belongs to the same family (e.g., unseen `gpt-4.1` texts classified as `gpt-5.1`), we treat that as a correct prediction. Models such as BERT-AA, TOPFORMER, and DETECTIVE show a performance gain, indicating that they lack the specificity to identify a particular LLM and tend to confuse it with models from the same family.

6.2 Ablation Study

Impact of Evaluator Language Model. We investigate the impact of using different evaluator language models ($\mathcal{L}_{\hat{\theta}}$), which form the main component of TRACE. These models are used to compute scores (rank and entropy) for each token, as defined in Section 4. We evaluate two other 1B models, `OLMo` and `Gemma`,[12] which differ in architecture, type, and size compared to `GPT-2` used in our experiments. We observe similar performance

[12]Exact model versions are `allenai/OLMo-2-0425-1B` and `google/gemma-3-1b-it`.

across different evaluator language models in the majority of settings (see Appendix Table 17). This suggests that TRACE exhibits considerable robustness to the choice of evaluator language model.

Possible Data Contamination. All classifiers, detectors, and evaluators used in our experiments—including TRACE and the baseline methods—are based on GPT-2 or BERT/RoBERTa architectures, which were not pre-trained on the Gutenberg corpus. For the frontier models serving as generators, there may be potential data contamination due to their pre-training data, since snippets from Gutenberg books are provided as few-shot references during GHOSTWRITEBENCH book generation. Importantly, this affects only the generation process and not the evaluation stage. Consistent with this, we observe no significant performance differences in TRACE when using alternative evaluator language models (e.g., Gemma, which likely includes Gutenberg data in their pre-training).

Further ablation studies investigating different components of TRACE (size, order, and similarity metrics), numbers of tokens, and other hyperparameters can be found in Appendix E.

7 Conclusion

We propose GHOSTWRITEBENCH, an authorship attribution dataset for long-form LLM texts. To study OOD generalisation, GHOSTWRITEBENCH examines two OOD settings: OOD-Domain and OOD-Author. Our experiments reveal that existing attribution methods show limited generalisability. We also introduce TRACE, a novel, interpretable fingerprint method for LLM attribution. TRACE uses token-level transition patterns, such as word ranks and entropy, to create text fingerprints. Our experiments demonstrate the effectiveness of TRACE in ID, OOD, and limited training data scenarios in GHOSTWRITEBENCH. Our research provides a new benchmark that explicitly tests for generalisation and establishes a new strong baseline, paving the way for future research directions in LLM text forensics.

Limitations

GHOSTWRITEBENCH is limited to English texts only; we plan to extend this dataset with cross-lingual OOD dimensions in future work. It would be interesting to study the effectiveness of TRACE in an OOD-language setting. Similarly, there are other OOD dimensions, such as time, human authors, and others, which are natural extensions of GHOSTWRITEBENCH. Although the books generated by our pipeline are of good quality, we acknowledge that book generation and related aspects, such as creativity, self-reflection, rewriting, etc., constitute research problems in their own right and fall outside the scope of this work. Adversarial robustness (e.g., paraphrasing or style-transfer attacks) is an orthogonal generalisation axis not studied in this work. That said, since TRACE is not a supervised method, we expect it to be less prone to such obfuscation attacks; a thorough evaluation remains an important direction for future work. Adapting TRACE for fewer (<10K) tokens remains an open problem. For BERT-AA, TOPFORMER, and DETECTIVE, we use chunking to handle long texts, following prior work. Alternatives such as filtering chunks by confidence or using overlapping windows are interesting avenues for future work. We also focus only on a single-authorship setup. A natural next step would be to study multi-authorship (i.e., a piece of text written by multiple authors), which is more common in long-form documents that typically involve collaborative writing.

Ethics Statement

GHOSTWRITEBENCH contains LLM-generated books across various genres. The content of these books may include writing or plot elements that reflect undesirable values. That said, all LLMs used incorporate built-in guardrails, so the likelihood of such issues should be minimal. This is further supported by our small-scale quality annotation, where no annotators identified any issues.

Acknowledgements

We appreciate the valuable feedback from all anonymous reviewers. We thank annotators Liuliu Chen, Ming-Bin Chen, Edoardo De Duro, Matteo Greco, Matteo Guida, Jemima Kang, Demian Inostroza, and Rui Xing for their time and support in data annotation.

This research was supported by The University of Melbourne's Research Computing Services and the Petascale Campus Initiative. A. Shetty was supported by the Commonwealth through an Australian Government Research Training Program Scholarship (DOI: https://doi.org/10.82133/C42F-K220), Computing and Information Systems

PhD Scholarship, and the Avashya Foundation. A. Shetty's conference travel was partially supported by the Engineering and IT Conference Travel Grant. J.H. Lau was supported by the Australian Research Council under Grant LP210200917 and DP240101006.

References

Arwa I. Alhussain and Aqil M. Azmi. 2021. Automatic story generation: A survey of approaches. *ACM Comput. Surv.*, 54(5).

Anthropic. 2025. System card: Claude Sonnet 4.5. Technical report, Anthropic.

Australian Society of Authors. 2025. Artificial intelligence. [Accessed 13-02-2026].

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2025. Longwriter: Unleashing 10,000+ word generation from long context LLMs. In *The Thirteenth International Conference on Learning Representations*.

Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2024. Booookscore: A systematic exploration of book-length summarization in the era of LLMs. In *The Twelfth International Conference on Learning Representations*.

Cyril Chhun, Pierre Colombo, Fabian Suchanek, and Chloé Clavel. 2022. Of human criteria and automatic metrics: A benchmark of the evaluation of story generation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5794–5836.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3416 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *Preprint*, arXiv:2507.06261.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A. Smith, and Yejin Choi. 2022. Is GPT-3 text indistinguishable from human text? scarecrow: A framework for scrutinizing machine text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7250–7274, Dublin, Ireland. Association for Computational Linguistics.

Liam Dugan, Alyssa Hwang, Filip Trhlík, Andrew Zhu, Josh Magnus Ludan, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. 2024. RAID: A shared benchmark for robust evaluation of machine-generated text detectors. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12463–12492, Bangkok, Thailand. Association for Computational Linguistics.

Maël Fabien, Esau Villatoro-Tello, Petr Motlicek, and Shantipriya Parida. 2020. BertAA : BERT fine-tuning for authorship attribution. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 127–137, Indian Institute of Technology Patna, Patna, India. NLP Association of India (NLPAI).

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116.

Dmitriy Genzel and Eugene Charniak. 2002. Entropy rate constancy in text. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 199–206.

GLM Team, Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, Kedong Wang, Lucen Zhong, Mingdao Liu, Rui Lu, Shulin Cao, Xiaohan Zhang, Xuancheng Huang, Yao Wei, and 152 others. 2025. GLM-4.5: Agentic, reasoning, and coding (ARC) foundation models. *Preprint*, arXiv:2508.06471.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Xun Guo, Yongxin He, Shan Zhang, Ting Zhang, Wanquan Feng, Haibin Huang, and Chongyang Ma. 2024. Detective: Detecting ai-generated text via multi-level contrastive learning. *Advances in Neural Information Processing Systems*, 37:88320–88347.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting llms with binoculars: zero-shot

detection of machine-generated text. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

John R Hayes and Linda S Flower. 2016. Identifying the organization of writing processes. In *Cognitive processes in writing*, pages 3–30. Routledge.

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.

Baixiang Huang, Canyu Chen, and Kai Shu. 2025. Authorship attribution in the era of llms: Problems, methodologies, and challenges. *ACM SIGKDD Explorations Newsletter*, 26(2):21–43.

T Florian Jaeger. 2010. Redundancy and reduction: Speakers manage syntactic information density. *Cognitive psychology*, 61(1):23–62.

Kaijie Jiao, Quan Wang, Licheng Zhang, Zikang Guo, and Zhendong Mao. 2025. M-RangeDetector: Enhancing generalization in machine-generated text detection through multi-range attention masks. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8971–8983, Vienna, Austria. Association for Computational Linguistics.

Angel B John and Sumam Mary Idicula. 2026. Authorship attribution of malayalam biblical texts using deep learning. In *2026 2nd International Conference on Advances in Intelligent Computing and Applications (AICAPS)*, pages 1–6.

Daniel Jurafsky and James H. Martin. 2026. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*, 3rd edition. Online manuscript released January 6, 2026.

Kimi Team, Yifan Bai, Yiping Bao, Y. Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, and 181 others. 2026. Kimi k2: Open agentic intelligence. *Preprint*, arXiv:2507.20534.

Moshe Koppel and Jonathan Schler. 2004. Authorship verification as a one-class classification problem. In *Proceedings of the twenty-first international conference on Machine learning*, page 62.

Tharindu Kumarage, Garima Agrawal, Paras Sheth, Raha Moraffah, Aman Chadha, Joshua Garland, and Huan Liu. 2024. A survey of ai-generated text forensic systems: Detection, attribution, and characterization. *arXiv preprint arXiv:2403.01152*.

Lucio La Cava, Davide Costa, and Andrea Tagarelli. 2024. Is contrasting all you need? contrastive learning for the detection and attribution of ai-generated text. In *ECAI 2024*, pages 3179–3186. SAGE Publications.

Lucio La Cava and Andrea Tagarelli. 2025. Openturingbench: An open-model-based benchmark and framework for machine-generated text detection and attribution. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 26666–26682.

Yukyung Lee, Soonwon Ka, Bokyung Son, Pilsung Kang, and Jaewook Kang. 2025. Navigating the path of writing: Outline-guided text generation with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 233–250.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2023. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11:102–121.

T. C. Mendenhall. 1887. The characteristic curves of composition. *Science*, ns-9(214s):237–246.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 24950–24962. PMLR.

Omar Momen, Manuel Schaaf, and Alexander Mehler. 2025. Filling the temporal void: Recovering missing publication years in the Project Gutenberg corpus using LLMs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 17318–17334, Vienna, Austria. Association for Computational Linguistics.

Frederick Mosteller and David L. Wallace. 1963. Inference in an authorship problem. *Journal of the American Statistical Association*, 58(302):275–309.

Tempestt Neal, Kalaivani Sundararajan, Aneez Fatima, Yiming Yan, Yingfei Xiang, and Damon Woodard. 2017. Surveying stylometry techniques and applications. *ACM Comput. Surv.*, 50(6).

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. GPT-4 technical report. *Preprint*, arXiv:2303.08774.

Vishakh Padmakumar and He He. 2024. Does writing with language models reduce content diversity? In *The Twelfth International Conference on Learning Representations*.

Qwen Team. 2026. Pushing qwen3-max-thinking beyond its limits.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Nikita Salkar, Thomas Trikalinos, Byron Wallace, and Ani Nenkova. 2022. Self-repetition in abstractive neural summarizers. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 341–350, Online only. Association for Computational Linguistics.

Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. 2012. Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1757–1772.

Chantal Shaib, Venkata S Govindarajan, Joe Barrow, Jiuding Sun, Alexa Siu, Byron C Wallace, and Ani Nenkova. 2025. Standardizing the measurement of text diversity: A tool and comparative analysis. In *Proceedings of The 14th International Joint Conference on Natural Language Processing and The 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 36–46, Mumbai, India. Association for Computational Linguistics.

Yuhui Shi, Yehan Yang, Qiang Sheng, Hao Mi, Beizhe Hu, Chaoxi Xu, and Juan Cao. 2025. Phantomhunter: Detecting unseen privately-tuned llm-generated text via family-aware learning. *Preprint*, arXiv:2506.15683.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, and 467 others. 2026. OpenAI GPT-5 system card. *Preprint*, arXiv:2601.03267.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.

Ariel Stolerman, Rebekah Overdorf, Sadia Afroz, and Rachel Greenstadt. 2014. Breaking the closed-world assumption in stylometric authorship attribution. In *IFIP International Conference on Digital Forensics*, pages 185–205. Springer.

Xiaofei Sun, Zijun Sun, Yuxian Meng, Jiwei Li, and Chun Fan. 2022. Summarize, outline, and elaborate: Long-text generation via hierarchical supervision from extractive summaries. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6392–6402.

The Authors Guild. 2024. AI Is Driving a New Surge of Sham "Books" on Amazon - The Authors Guild. [Accessed 13-02-2026].

Jacob Tyo, Bhuwan Dhingra, and Zachary C. Lipton. 2023. Valla: Standardizing and benchmarking authorship attribution and verification through empirical evaluation and comparative analysis. 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 649–660, Nusa Dua, Bali. Association for Computational Linguistics.

Adaku Uchendu, Thai Le, and Dongwon Lee. 2024. TOPFORMER: Topology-aware authorship attribution of deepfake texts with diverse writing styles. In *27th European Conference on Artificial Intelligence, ECAI 2024*, pages 1446–1454. IOS Press BV.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, Online. Association for Computational Linguistics.

Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. Turingbench: A benchmark environment for turing test in the age of neural text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2001–2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2024. Ghostbuster: Detecting text ghostwritten by large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1702–1717, Mexico City, Mexico. Association for Computational Linguistics.

Wenqing Wang, Mingqi Gao, Xinyu Hu, and Xiaojun Wan. 2025. Towards a "novel" benchmark: Evaluating literary fiction with large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 21648–21673.

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. 2024. 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.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. 2025. A

survey on llm-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, 51(1):275–338.

Haotian Xia, Hao Peng, Yunjia Qi, Bin Xu, Juanzi Li, Hou Lei, and Xiaozhi Wang. 2025. Storywriter: A multi-agent framework for long story generation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 6559–6563.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Xianjun Yang, Wei Cheng, Yue Wu, Linda Ruth Petzold, William Yang Wang, and Haifeng Chen. 2024. DNA-GPT: Divergent n-gram analysis for training-free detection of GPT-generated text. In *The Twelfth International Conference on Learning Representations*.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: towards better automatic storytelling. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Xiao Yu, Kejiang Chen, Qi Yang, Weiming Zhang, and Nenghai Yu. 2024. 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, Miami, Florida, USA. Association for Computational Linguistics.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 1097–1100.

Appendix

A GHOSTWRITEBENCH Generation

Code	Genre
LF	Literature & Fiction
HB	History & Biographies
BLP	Business, Law & Politics
ST	Science & Technology
ACM	Art, Culture & Media
SSP	Social Sciences & Philosophy
JR	Journals & Reports
LHH	Lifestyle, Health & Hobbies

Table 5: Different genres used in this work, sourced from Project Gutenberg (Momen et al., 2025).

A.1 Book Generation Prompts

As described in Section 3.1, our book generation involves first generating the book outline (see Prompt 16), followed by incremental generation conditioned on the current running summary, outline, and last segment (see Prompts 17 and 18). For summarisation, we adapt the iterative summarisation prompts (see Prompts 14 and 15) from Chang et al. (2024).

A.2 Quantitative Analysis

In addition to the standard language understanding metric, perplexity, we evaluate several text diversity metrics as mentioned in Table 2. Considering we are operating in a reference-free setup, we focus on two prominent types of text diversity metrics, similarity between text pairs from the same model, and the token/type ratio (Shaib et al., 2025):

- **Self-BLEU**: Self-BLEU (S-B) measures similarity of each generated text against the remaining texts using BLEU (Zhu et al., 2018). It is a homogenisation score widely used to evaluate text diversity in writing tasks (Padmakumar and He, 2024).
- **n-Gram Diversity Score**: n-Gram Diversity Score (NGD) (Meister et al., 2023) calculates the ratio of unique n-grams to all n-grams in the dataset. In our analysis, we consider n-grams of length four.
- **Self-repetition**: Self-repetition (S-R) (Salkar et al., 2022) measures the extent to which n-grams (up to 4-grams) are repeated across different outputs of the same LLM.
- **Compression Ratio**: Compression Ratio (CR) (Shaib et al., 2025) defined as the ratio of all LLM-generated texts original file size and to the compressed file size (using gzip), this metric captures redundancy in a manner distinct from n-gram-based measures.
- **Perplexity**: Perplexity (PPL) (Jurafsky and Martin, 2026) is computed as the average negative log-likelihood of the texts using the `GPT-2-Large` language model as a standard measure of language understanding.

LLM	Book Generation			
	Steps	Outline	Segment	Total
`deepseek-chat`	39.90	0.6K	0.8K	33.4K
`glm-4.6`	14.12	1.8K	2.0K	28.4K
`kimi-k2`	15.65	2.4K	2.4K	37.1K
`qwen3-235b`	38.47	2.0K	1.7K	67.1K
`qwen3-max`	27.53	1.4K	1.5K	41.2K
`claude-sonnet`	11.96	1.7K	3.9K	47.1K
`gemini-flash`	8.87	2.7K	2.3K	20.5K
`gemini-pro`	45.85	2.2K	1.9K	88.1K
`gpt-4.1`	34.00	1.2K	1.9K	66.1K
`gpt-5.1`	15.92	4.1K	4.4K	69.3K

Table 6: Book generation statistics across different LLMs. All reported numbers (other than steps) are in words.

A.3 Qualitative Analysis

Following Chhun et al. (2022); Wang et al. (2025), we evaluate LLM-generated books across several dimensions capturing meso-, micro-, and macro-level properties.

- **Relevance (Rel.)**: how well the text aligns with or follows the given instruction. Ensures that the generated text is appropriate and follows the given book outline.
- **Engagement (Eng.)**: measures the degree to which the reader is engaged with the book.
- **Coherence (Coh.)**: measures whether the story makes sense. Ensures the generated text flows logically from beginning to end, with consistent narrative elements and clear progression.
- **Fluency (Flu.)**: measures the smoothness and natural flow of the language.

LLM	Genre	Passage-level					Overall				
		Rel.	Eng.	Coh.	Flu.	Div.	Coh.	Emp.	Sur.	Eng.	Comp.
deepseek-chat	SSP	3.2	2.6	3.4	3.4	4.0	3.0	3.5	4.0	3.0	3.0
glm-4.6	HB,SSP,JR	4.9	3.7	4.3	4.4	4.0	4.0	4.0	3.0	4.0	3.0
kimi-k2	LF	4.8	4.4	4.2	4.2	4.4	4.0	5.0	3.0	5.0	4.0
qwen3-235b	SSP,LF	4.6	3.7	4.7	4.8	4.7	4.0	3.0	3.5	3.5	4.0
qwen3-max	LHH,ST	3.7	3.0	4.4	4.0	3.7	3.0	3.5	4.0	2.5	2.5
claude-sonnet	LF	4.0	3.9	4.2	4.7	4.5	4.0	4.0	3.5	3.5	4.0
gemini-flash	LHH,LF	4.6	3.8	4.2	4.4	4.6	4.0	4.0	3.0	4.0	2.0
gemini-pro	LHH	3.5	3.2	4.0	3.7	4.7	3.5	4.0	4.0	3.0	3.0
gpt-4.1	HB	4.3	3.7	4.3	4.2	4.4	4.0	3.0	3.0	3.0	3.5
gpt-5.1	ACM	4.4	3.6	5.0	4.8	4.8	4.0	4.0	3.0	3.0	4.0
Overall		4.2	3.6	4.3	4.3	4.4	3.8	3.8	3.4	3.5	3.3

Table 7: Human evaluation scores across passage-level and overall narrative quality dimensions for different LLMs. Rel. - Relevance, Eng. - Engagement, Coh. - Coherence, Flu. - Fluency, Div. - Diversity, Emp. - Empathy, Sur. - Surprise, and Comp. - Complexity. 1 is the lowest and 5 is the highest score.

- **Diversity (Div.)**: measures variety in linguistic expression, including rich vocabulary and varied sentence structures.
- **Empathy (Emp.)**: measures how well the model captures the characters' emotions, regardless of whether the reader agrees with them.
- **Surprise (Sur.)**: measures how surprising the ending of the book is.
- **Complexity (Comp.)**: measures how elaborate the book is.

Annotators. The annotators were volunteers from our university representing mixed demographics. All annotators were fluent in English, and we evaluated 10 books, one for each LLM. Annotators were provided with clear instructions for the assigned task. No compensation beyond voluntary participation was offered, and all annotators were fully informed about the nature of the study and their role.

Annotation Task. Given that our generated books are long, we randomly sample five passages capturing the flow of each book. Each passage comprises five questions, followed by five questions at the end of the book, totalling 30 questions per book (results in Table 7). Each question is answered on a 5-point Likert scale. We acknowledge potential confounding factors (such as annotator, genre, and model). The human evaluation is intended primarily to provide confidence that the generated text is reasonable.

Results. Table 7 presents the average annotator scores across different quality dimensions for both at the passage level and overall for each book. We note strong scores (> 4; out of 5) for relevance, coherence, fluency, and diversity, complementing the text quality and diversity analysis from Table 2. For other subjective dimensions (engagement, empathy, surprise, and complexity), the scores are also generally positive (> 3.3). This indicates that the texts in GHOSTWRITEBENCH exhibit good quality qualitatively. Annotators' agreement (Krippendorff's $\alpha = 0.17$) remains low across evaluation tasks. We attribute this to the highly subjective nature of the task and the length of the generations (49K words on average), which can make it difficult for annotators to follow the narrative consistently. However, 70% of ratings were within ± 1 Likert point, with a mean difference of 1.05, indicating that most disagreements were relatively minor.

A.4 Cost Analysis

Appendix Table 8 lists generation costs per LLM. In total, generating GHOSTWRITEBENCH cost approximately $530.

B TRACE Fingerprint

B.1 Similarity Metrics

We formulate the two similarity metrics used to compare two fingerprints, f_A and f_B, below:

LLM	API Cost ($/1M)		Total
	Input	Output	(in $)
deepseek-chat	0.2	0.5	10.6
glm-4.6	0.5	1.8	10.9
kimi-k2	0.5	2.0	22.1
qwen3-235b	0.1	0.1	9.0
qwen3-max	1.2	6.0	60.2
claude-sonnet	3.0	10.0	85.9
gemini-flash	0.3	2.5	6.6
gemini-pro	1.3	10.0	155.0
gpt-4.1	2.0	8.0	81.4
gpt-5.1	1.3	10.0	88.7
Total			530.4

Table 8: GHOSTWRITEBENCH generation cost breakdown. 'Input' and 'Output' denote API cost per 1M tokens. 'Total' is the cost of generating all books for a given LLM (as per the statistics in Table 2). OpenRouter API costs are as of September 2025.

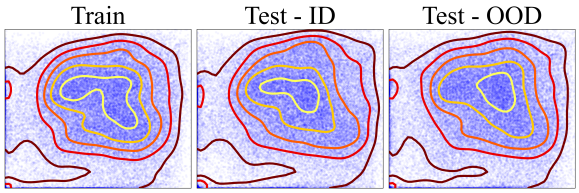


Figure 3: An example where TRACE captures similar gpt-4.1 fingerprints across domains, demonstrating generalisation. Left: a training reference fingerprint; Middle: a test sample from seen domain (HB); Right: a test sample from an unseen domain (SSP,LHH).

JS Distance.

$$\mathrm{JS} = \sqrt{\frac{1}{2} D_{\mathrm{KL}}(f_A \,\|\, M) + \frac{1}{2} D_{\mathrm{KL}}(f_B \,\|\, M)}$$

with

$$M = \frac{f_A + f_B}{2}, \quad D_{\mathrm{KL}}(p \,\|\, q) = \sum_k p_k \log_2 \frac{p_k}{q_k}.$$

Norm-Mean. Compute gradients of fingerprint f:

$$f_x(i,j) \approx \frac{f(i,j+1) - f(i,j-1)}{2\,\mathrm{d}x},$$
$$f_y(i,j) \approx \frac{f(i+1,j) - f(i-1,j)}{2\,\mathrm{d}y}.$$

Using gradients, we compute surface normals:

$$\hat{\mathbf{n}}(i,j) = \frac{(-f_x(i,j), -f_y(i,j), 1)}{\sqrt{f_x(i,j)^2 + f_y(i,j)^2 + 1}}.$$

Mean angular difference between fingerprints f_A and f_B:

$$\text{norm-mean} = \frac{1}{n^2} \sum_{i,j=1}^{n} \arccos\big(\hat{\mathbf{n}}_A(i,j) \cdot \hat{\mathbf{n}}_B(i,j)\big),$$

where $\hat{\mathbf{n}}_A$ and $\hat{\mathbf{n}}_B$ are computed from f_A and f_B.

C Experimental Details (cont.)

We use OpenRouter[13] for all LLM API calls used for GHOSTWRITEBENCH generation, with temperature = 1.0, top-P = 1.0, top-K = 0, and maximum_output_length = 8192 tokens. We provide a model card for all LLMs used in this work in Table 9. All experiments were conducted using a single A100 GPU with CUDA 12.4 and PyTorch 2.10.0.

For BERT-AA and TOPFORMER, as they do not process long-form documents by default, we split each text into 512-token chunks and average the prediction scores to compute the final prediction, following Tyo et al. (2023). For DETECTIVE, since it is clustering-based, we compute the percentage of top-K results across text chunks. As noted earlier, all hyperparameter tuning was performed on the development set, with 5 epochs and 20 epochs of fine-tuning for both BERT-AA and TOPFORMER and DETECTIVE, respectively. In Rank-based fingerprint, we set $\alpha = 1.5$ and use 100K samples for power-law approximation (more in Appendix Algorithm 1). For all TRACE experiments, we use GPT-2 as evaluator language model using context of 1024 and 50 grid size (or cluster for ranks) unless stated otherwise.

[13] https://openrouter.ai/

Model Name	OpenRouter Checkpoint	Context	Max. Out.	Type	Reference
deepseek-chat	deepseek/deepseek-chat-v3-0324	163.8K	163.8K	Open	DeepSeek-AI et al. (2025)
glm-4.6	z-ai/glm-4.6	204.8K	204.8K	Open	GLM Team et al. (2025)
kimi-k2	moonshotai/kimi-k2-thinking	131.1K	131.1K	Open	Kimi Team et al. (2026)
qwen3-235b	qwen/qwen3-235b-a22b-2507	262.1K	262.1K	Open	Yang et al. (2025)
qwen3-max	qwen/qwen3-max	262.1K	32.8K	Closed	Qwen Team (2026)
claude-sonnet	anthropic/claude-sonnet-4.5	1M	64K	Closed	Anthropic (2025)
gemini-flash	google/gemini-2.5-flash	1.05M	65.5K	Closed	Comanici et al. (2025)
gemini-pro	google/gemini-2.5-pro	1.05M	65.5K	Closed	Comanici et al. (2025)
gpt-4.1	openai/gpt-4.1	1.05M	32.8K	Closed	OpenAI et al. (2024)
gpt-5.1	openai/gpt-5.1	400K	128K	Closed	Singh et al. (2026)

Table 9: Details for all the LLMs used in GHOSTWRITEBENCH. 'Context' is the total input context supported and 'Max. Out.' is the maximum number of output tokens supported by the LLM. 'Type' indicates whether model weights are public (Open) or if it is proprietary LLM (Closed).

D Baselines (cont.)

Rank. We compute the average token rank for a given text using the `GPT-2-Medium` model.

Entropy. Similarly, we compute the average token entropy for a given text using the evaluator model, `GPT-2-Medium`.

GLTR. GLTR (Gehrmann et al., 2019) computes the probability of a token falling into four rank ranges (or buckets). Compared with a single average-based metric, it is more representative.

For all three above metric-based techniques, we adapt them to the AA task following La Cava and Tagarelli (2025) by attaching a logistic regression classifier on top of these metric values.

N-GRAM. This is a traditional but effective technique that ensembles three n-gram models: character-based, part-of-speech-based, and summary statistics. Moreover, it can naturally handle long texts, so no chunking is required, unlike the next two methods.

BERT-AA. A multi-class classification layer is added to the pre-trained BERT model (Fabien et al., 2020), which is then fine-tuned on a specialised dataset. However, as a Transformer-based architecture (Vaswani et al., 2017), it has a bottleneck of processing at most 512 tokens at a time (Devlin et al., 2019).

TOPFORMER. **Top**ological Trans**former**-based model (TOPFORMER) (Uchendu et al., 2024) attaches topological data analysis (TDA) layer on top of pre-trained ROBERTA. The motivation is that TDA is known to capture shape and structure (i.e., linguistic structure) enhancing contextual representations of base model, which is fine-tuned similar to BERT-AA.

DETECTIVE. This is a multi-level contrastive learning–based state-of-the-art method for binary AI detection (Guo et al., 2024). However, it internally learns representations for each author. Therefore, these embeddings can also be used for the AA task. We also considered more recent M-RANGEDETECTOR (Jiao et al., 2025), PHANTOMHUNTER (Shi et al., 2025), and other simpler contrastive learning methods (La Cava et al., 2024; John and Idicula, 2026), but their implementations are not publicly available, and we were unable to obtain sufficient details for reproduction.

E Further Ablation Studies

Top-k Accuracy AA Performance. Following prior work in authorship attribution (Huang et al., 2025; Uchendu et al., 2020), we additionally report top-k accuracy ($k = 1, 2, 3, 5$) as a complementary evaluation metric. Since there are only ten authors in our case, top-k accuracy saturates quickly as seen in Table 14. We nevertheless include it for completeness, and observe that it is consistent with the main findings reported using macro-F_1 in Table 13.

OOD-Author Additional Analysis. To further strengthen our OOD-Author evaluation, we created an additional balanced test set containing both unknown and known (randomly sampled from same split) generators, resembling a mixed deployment scenario, reported in Table 11. We also conducted error analysis, evaluating per-LLM

Algorithm 1 Rank-based Fingerprint (f^{rank}) Compression

Require: $|\mathcal{V}|$, $R = \langle r_1, \ldots, r_m \rangle$, α (power-law exponent), C (clusters)

$p(i) \leftarrow$ power-law approx. with α, $\forall i \in \{1 \ldots |\mathcal{V}|\}$
cum $\leftarrow 0$, $k \leftarrow 1$, $\Delta \leftarrow 1/C$
for $i \leftarrow 1$ to $|\mathcal{V}|$ **do**
    cluster$[i] \leftarrow k$
    cum $\leftarrow$ cum $+ p(i)$
    while cum $\geq \Delta$ **do**
        cum $\leftarrow$ cum $- \Delta$
        $k \leftarrow k + 1$
    end while
end for
$K \leftarrow k$ ▷ number of used clusters
$f^{\text{rank}} \leftarrow \mathbf{0}^{K \times K}$
for $i \leftarrow 2$ to m **do**
    $f^{\text{rank}}[\text{cluster}[R_{i-1}], \text{cluster}[R_i]] \mathrel{+}= 1$
end for
return f^{rank} ▷ $\mathbb{R}^{K \times K}$

Algorithm 2 Entropy-based Fingerprint (f^{entropy}) Smoothing

Require: $|\mathcal{V}|$, $E = \langle e_1, \ldots, e_m \rangle$, G (grid size)

$P \leftarrow \{(e_{i-1}, e_i) \mid i = 2, \ldots, m\}$
KDE $\leftarrow$ GaussianKDE(P)
$x_{\text{grid}}, y_{\text{grid}} \leftarrow \text{linspace}(0, \log(|\mathcal{V}|), G)$
$f^{\text{entropy}} \leftarrow \text{KDE}(\text{meshgrid}(x_{\text{grid}}, y_{\text{grid}}))$
return f^{entropy} ▷ $\mathbb{R}^{G \times G}$

Model	Split	Genres	# Books
deepseek-chat	ID	LF	33
	OOD-Domain	LHH, HB	2
		BLP, HB	2
		SSP	5
glm-4.6	ID	HB, SSP, JR	3
		HB, JR	2
		SSP, LF	15
		LF	6
	OOD-Domain	ST	6
kimi-k2	ID	LF	30
	OOD-Domain	HB	5
		ST	2
qwen3-235b	ID	LF	11
		SSP, LF	19
	OOD-Domain	BLP, LHH, HB	1
		BLP, HB	3
		HB	4
qwen3-max	ID	LHH, ST	11
		SSP	11
		SSP, ACM	6
	OOD-Domain	LF	6
claude-sonnet	ID	HB, LF	11
		LF	8
	OOD-Domain	BLP, SSP	5
gemini-flash	ID	HB, ACM	7
		SSP	17
	OOD-Domain	LHH, LF	6
gemini-pro	ID	LHH, HB	5
		HB	6
		LHH	11
	OOD-Domain	LF	5
gpt-4.1	ID	HB	29
	OOD-Domain	LHH, SSP	6
gpt-5.1	ID	SSP, ACM	7
		ACM	7
		SSP	7
	OOD-Domain	ST, LF	2
		LHH, ST	3

Table 10: Detailed genre splits (ID vs. OOD-Domain) for different LLMs in GHOSTWRITEBENCH, complementing Table 2. These genre splits replicate human book distribution from Project Gutenberg.

macro-F_1 OOD-Author performance in Table 12 to check whether performance is dominated by a few LLMs. It shows that the macro-average is usually not driven by a single or two models for TRACE and is more balanced compared to other methods. It further highlights substantial variation in difficulty across LLMs (e.g., `gemini-pro`, the `Qwen` family are generally more challenging). Overall, these results further strengthen and complement the main OOD-Author results from Table 3.

Method	OOD-Author	
	AUROC	AUPR
RANK	0.49	0.62
ENTROPY	0.58	0.66
GLTR	0.51	0.63
N-GRAM	0.90	0.89
BERT-AA	0.83	0.86
TOPFORMER	0.80	0.84
DETECTIVE	0.81	0.84
TRACE (Ours)		
Ranks-JS	0.72	0.74
Entropy-JS	**0.92**	**0.93**
Entropy-Norm	**0.92**	**0.93**

Table 11: OOD-Author performance on balanced test set containing both unknown and known author samples. Reported numbers are average threshold-independent metrics (AUROC and AUPR) over three random splits. Entropy-based TRACE is remains performant as seen in the main results.

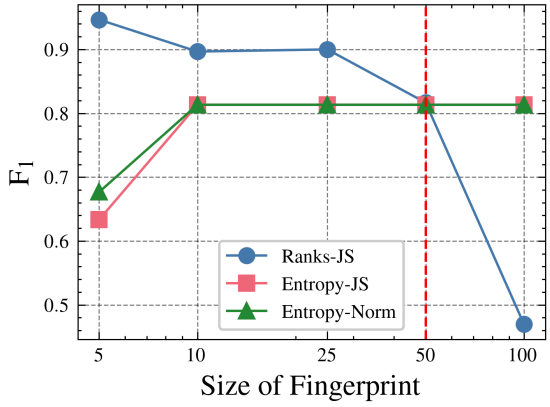


Figure 4: The impact of using different fingerprint sizes (K clusters for Rank-based and G grid-size for Entropy-based) for TRACE. The red dashed line denotes the selected configuration.

Impact of Fingerprint Size and Order. We investigate different fingerprint sizes for Rank-based and Entropy-based TRACE variants. As the cluster-based compression is applied over a large range of ranks, the size and order of the fingerprint can be important factors. From Figure 4, we observe that lower fingerprint sizes (i.e., grid size G from Section 4.2) perform poorly for the Entropy-based variant, whereas larger sizes (i.e., clusters K from Section 4.1) are less effective for the Rank-based variant. A higher number of clusters when compressing ranks may not sufficiently reduce sparsity. A similar trend appears in Appendix Figure 8, where increasing the transition order (e.g., from 2nd order ($r_{i-1} \rightarrow r_i$; default) to 3rd order transitions ($r_{i-2} \rightarrow r_{i-1} \rightarrow r_i$)) leads to increased sparsity and reduced performance.

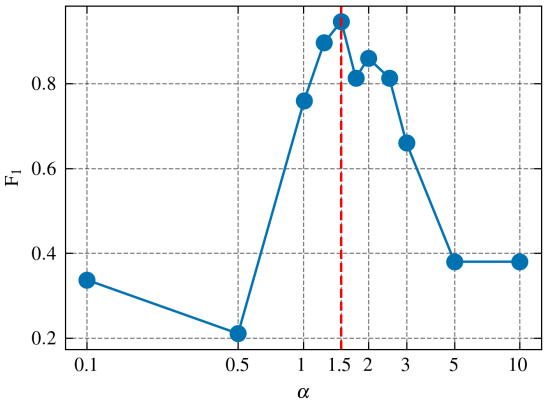


Figure 5: The impact of α in power law approximation for Rank-based TRACE. The red dashed line denotes the chosen $\alpha = 1.5$.

Impact of α in Rank-based Fingerprint. Considering the sparsity in Rank-based fingerprints due to the large vocabulary size, we perform compression clustering by grouping ranks into equal-probability bins, as outlined in Appendix Algorithm 1. We approximate ranks distribution as power-law distribution using the parameter α. We investigate the impact of α in TRACE. As expected, for $\alpha < 1$ or $\alpha > 5$, performance degrades because the distribution no longer resembles a power law. Fortunately, a range of α values can work, as seen in Figure 5; we use $\alpha = 1.5$ unless stated otherwise.

Impact of Number of Tokens. Although this paper studies a long-form document, we investigate how limiting the number of tokens (per book) impacts the performance of TRACE. From Figure 6,

Method	N-GRAM	BERT-AA	DETECT.	TOPFORM.	TRACE (Ours)			Avg.
					Ranks-JS	Entropy-JS	Entropy-Norm	
deepseek-chat	0.31	0.22	0.26	0.00	0.95	0.99	0.99	0.53
glm-4.6	0.28	0.14	0.46	0.25	0.32	0.31	0.28	0.29
kimi-k2	0.33	0.28	0.80	0.84	0.40	0.32	0.31	0.47
qwen3-235b	0.33	0.00	0.22	0.22	0.00	0.22	0.44	0.20
qwen3-max	0.54	0.36	0.32	0.27	0.00	0.17	0.05	0.24
claude-sonnet	0.22	0.07	0.58	0.59	0.19	0.98	0.96	0.51
gemini-flash	0.44	0.00	0.56	0.00	0.00	0.78	0.67	0.35
gemini-pro	0.17	0.21	0.21	0.17	0.00	0.33	0.12	0.17
gpt-4.1	0.27	0.22	0.37	0.43	0.05	0.74	0.23	0.33
gpt-5.1	0.29	0.17	0.78	1.00	0.00	0.51	0.34	0.44

Table 12: OOD-Author per-LLM macro-F_1 scores (averaged over the three splits). 'Avg.' column is row-wise average, highlighting some LLMs are particularly more difficult to differentiate.

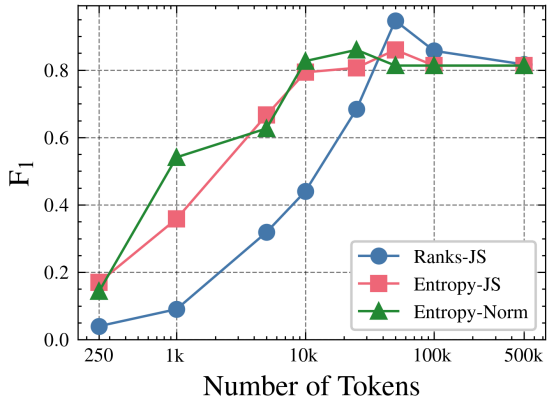


Figure 6: The impact of the number of tokens considered per text in TRACE.

we can see that for Entropy-based fingerprints, the performance converges at 10K tokens, whereas for Rank-based, it converges at 50K. This suggests TRACE works for shorter documents as well; we leave it for future work.

Different Similarity Metrics. We investigate several metrics for comparing two TRACE fingerprints in Figure 7. We observe that JS distance (along with Wasserstein or Earth-Mover distance) is effective for Rank-based fingerprints. This is expected when these fingerprints are viewed as discrete probability functions. In contrast, for Entropy-based fingerprints, the differences are small, and most metrics perform similarly. Future work could explore information-geometric metrics to further improve discrimination.

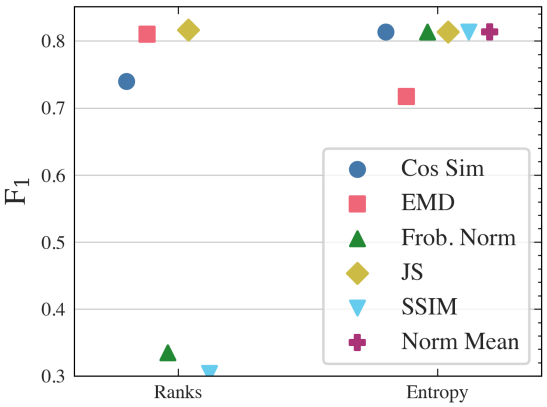


Figure 7: Evaluation of different metrics for comparing TRACE fingerprints.

Method	ID	OOD-Domain	OOD-Author
RANK	0.13 ± 0.04	0.13 ± 0.03	0.39 ± 0.19
ENTROPY	0.16 ± 0.03	0.18 ± 0.02	0.15 ± 0.12
GLTR	0.15 ± 0.01	0.15 ± 0.01	0.36 ± 0.07
N-GRAM	0.68 ± 0.08	0.56 ± 0.10	0.32 ± 0.33
BERT-AA	0.70 ± 0.10	$\underline{0.59 \pm 0.12}$	0.17 ± 0.05
TOPFORMER	0.73 ± 0.12	0.56 ± 0.10	0.38 ± 0.05
DETECTIVE	0.74 ± 0.06	0.56 ± 0.15	0.45 ± 0.15
TRACE			
Ranks-JS	0.59 ± 0.13	0.44 ± 0.11	0.19 ± 0.03
Entropy-JS	$\underline{0.76 \pm 0.08}$	0.52 ± 0.03	$\mathbf{0.54 \pm 0.15}$
Entropy-Norm	$\mathbf{0.81 \pm 0.10}$	$\mathbf{0.66 \pm 0.06}$	$\underline{0.44 \pm 0.09}$

Table 13: Results on GHOSTWRITEBENCH for all authors (including both low-resource and high-resource), complementing Table 3.

Method	ID				OOD-Domain			
	Top-1	Top-2	Top-3	Top-5	Top-1	Top-2	Top-3	Top-5
RANK	0.33 ± 0.04	0.46 ± 0.04	0.59 ± 0.05	0.81 ± 0.05	0.33 ± 0.03	0.51 ± 0.05	0.62 ± 0.04	0.86 ± 0.02
ENTROPY	0.42 ± 0.03	0.52 ± 0.00	0.69 ± 0.08	0.86 ± 0.01	0.44 ± 0.03	0.47 ± 0.01	0.75 ± 0.07	0.86 ± 0.01
GLTR	0.41 ± 0.07	0.55 ± 0.03	0.67 ± 0.09	0.82 ± 0.06	0.43 ± 0.08	0.52 ± 0.02	0.70 ± 0.03	0.85 ± 0.04
N-GRAM	0.91 ± 0.02	0.97 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	$\underline{0.87 \pm 0.04}$	$\mathbf{0.94 \pm 0.02}$	0.95 ± 0.01	0.97 ± 0.00
BERT-AA	0.92 ± 0.05	0.96 ± 0.01	0.97 ± 0.02	0.98 ± 0.03	0.85 ± 0.05	0.92 ± 0.02	0.97 ± 0.04	0.98 ± 0.03
TOPFORMER	0.94 ± 0.02	0.97 ± 0.00	0.97 ± 0.00	0.99 ± 0.01	0.86 ± 0.04	0.96 ± 0.03	0.97 ± 0.02	0.99 ± 0.01
DETECTIVE	$\underline{0.93 \pm 0.02}$	0.96 ± 0.02	$\underline{0.98 \pm 0.02}$	$\underline{0.99 \pm 0.01}$	$\mathbf{0.89 \pm 0.02}$	$\mathbf{0.94 \pm 0.02}$	0.95 ± 0.01	0.96 ± 0.01
TRACE								
Ranks-JS	0.80 ± 0.09	0.95 ± 0.02	0.97 ± 0.00	0.97 ± 0.00	0.69 ± 0.09	0.79 ± 0.04	0.85 ± 0.03	$\underline{0.98 \pm 0.01}$
Entropy-JS	0.90 ± 0.05	$\mathbf{0.98 \pm 0.01}$	$\mathbf{0.98 \pm 0.01}$	$\underline{0.99 \pm 0.01}$	0.76 ± 0.05	0.90 ± 0.03	0.93 ± 0.03	$\mathbf{0.99 \pm 0.01}$
Entropy-Norm	$\mathbf{0.94 \pm 0.03}$	$\underline{0.97 \pm 0.00}$	0.97 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.83 ± 0.04	$\underline{0.93 \pm 0.01}$	$\mathbf{0.97 \pm 0.00}$	$\mathbf{0.99 \pm 0.01}$

Table 14: Top-k accuracy results on GHOSTWRITEBENCH for all authors, complementing Table 3 and Table 13.

Method	Low Resource		High Resource	
	ID	OOD-Domain	ID	OOD-Domain
N-GRAM	0.47 ± 0.09	0.20 ± 0.16	0.95 ± 0.01	$\underline{0.92 \pm 0.04}$
BERT-AA	0.51 ± 0.23	0.24 ± 0.17	0.95 ± 0.04	0.91 ± 0.06
TOPFORMER	$\mathbf{0.69 \pm 0.06}$	0.31 ± 0.08	$\underline{0.96 \pm 0.04}$	0.91 ± 0.05
DETECTIVE	0.58 ± 0.14	0.31 ± 0.08	$\underline{0.96 \pm 0.04}$	$\mathbf{0.93 \pm 0.04}$
TRACE (Ours)				
Ranks-JS	0.39 ± 0.25	0.21 ± 0.19	0.81 ± 0.04	0.73 ± 0.13
Entropy-JS	0.64 ± 0.20	$\underline{0.34 \pm 0.12}$	0.91 ± 0.05	0.81 ± 0.06
Entropy-Norm	$\underline{0.67 \pm 0.19}$	$\mathbf{0.56 \pm 0.19}$	$\mathbf{0.96 \pm 0.02}$	0.85 ± 0.05

Table 15: Without thresholding results for attribution techniques on GHOSTWRITEBENCH, showing ID and OOD-Domain for Low and High resource settings, complementing Table 3. **Bold** denotes the best result and underline the second best.

Method	Low Resource		High Resource	
	Reject	Family	Reject	Family
N-GRAM	0.36 ± 0.31	0.19 ± 0.26	0.28 ± 0.35	0.03 ± 0.04
BERT-AA	0.04 ± 0.06	0.55 ± 0.10	0.29 ± 0.05	0.09 ± 0.08
TOPFORMER	0.29 ± 0.17	0.55 ± 0.10	0.46 ± 0.27	0.15 ± 0.13
DETECTIVE	0.40 ± 0.20	0.49 ± 0.18	0.49 ± 0.22	0.15 ± 0.11
TRACE (Ours)				
Ranks-JS	0.18 ± 0.03	0.53 ± 0.22	0.20 ± 0.08	0.32 ± 0.12
Entropy-JS	0.49 ± 0.14	0.11 ± 0.16	0.58 ± 0.20	0.19 ± 0.07
Entropy-Norm	0.51 ± 0.17	0.11 ± 0.16	0.37 ± 0.03	0.25 ± 0.11

Table 16: Variant of OOD-Author evaluation. 'Family' column: if the predicted class for the unseen LLM belongs to the same family, we treat as a correct prediction (instead of rejecting: 'Reject' column).

Model	Low Resource			High Resource		
	ID	OOD-Domain	OOD-Author	ID	OOD-Domain	OOD-Author
GPT-2	0.40 ± 0.24	0.22 ± 0.19	0.18 ± 0.03	0.78 ± 0.06	0.66 ± 0.12	0.20 ± 0.08
OLMo	0.22 ± 0.14	0.32 ± 0.22	0.11 ± 0.08	0.74 ± 0.05	0.56 ± 0.08	0.18 ± 0.05
Gemma	0.33 ± 0.22	0.27 ± 0.04	0.11 ± 0.16	0.79 ± 0.06	0.63 ± 0.08	0.15 ± 0.03

(a) Rank-based.

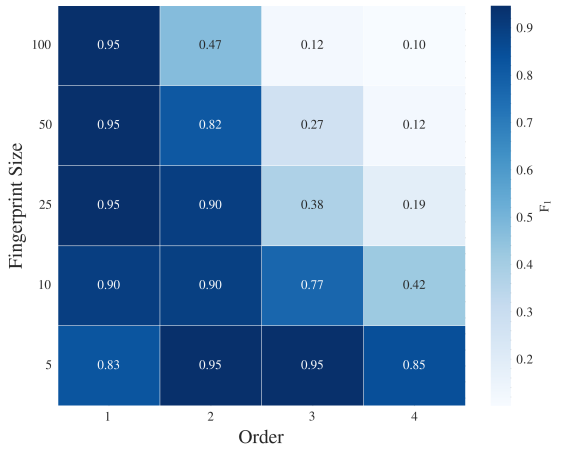
Model	Low Resource			High Resource		
	ID	OOD-Domain	OOD-Author	ID	OOD-Domain	OOD-Author
GPT-2	0.60 ± 0.16	0.23 ± 0.12	0.49 ± 0.14	0.91 ± 0.06	0.82 ± 0.08	0.58 ± 0.20
OLMo	0.71 ± 0.08	0.24 ± 0.11	0.29 ± 0.08	0.95 ± 0.02	0.75 ± 0.12	0.47 ± 0.13
Gemma	0.51 ± 0.08	0.10 ± 0.08	0.42 ± 0.08	0.89 ± 0.06	0.80 ± 0.09	0.54 ± 0.18

(b) Entropy-based (JS Dist.).

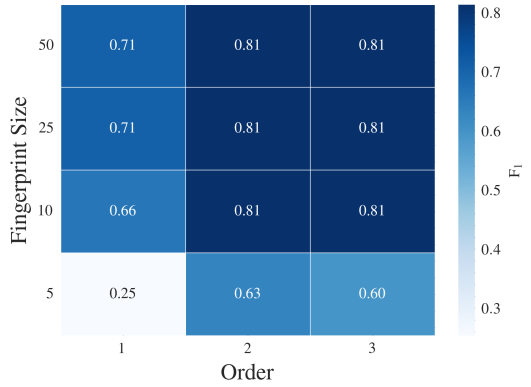
Model	Low Resource			High Resource		
	ID	OOD-Domain	OOD-Author	ID	OOD-Domain	OOD-Author
GPT-2	0.67 ± 0.19	0.49 ± 0.11	0.51 ± 0.17	0.96 ± 0.02	0.82 ± 0.10	0.37 ± 0.03
OLMo	0.70 ± 0.08	0.34 ± 0.10	0.29 ± 0.11	0.92 ± 0.04	0.81 ± 0.09	0.54 ± 0.09
Gemma	0.33 ± 0.09	0.20 ± 0.16	0.67 ± 0.11	0.87 ± 0.13	0.80 ± 0.09	0.69 ± 0.09

(c) Entropy-based (norm-mean).

Table 17: TRACE performance for different evaluator language models. Each sub-table is for a variant of the TRACE fingerprint. Overall, we observe comparable performance using different evaluator language models.



(a) Rank-based



(b) Entropy-Based

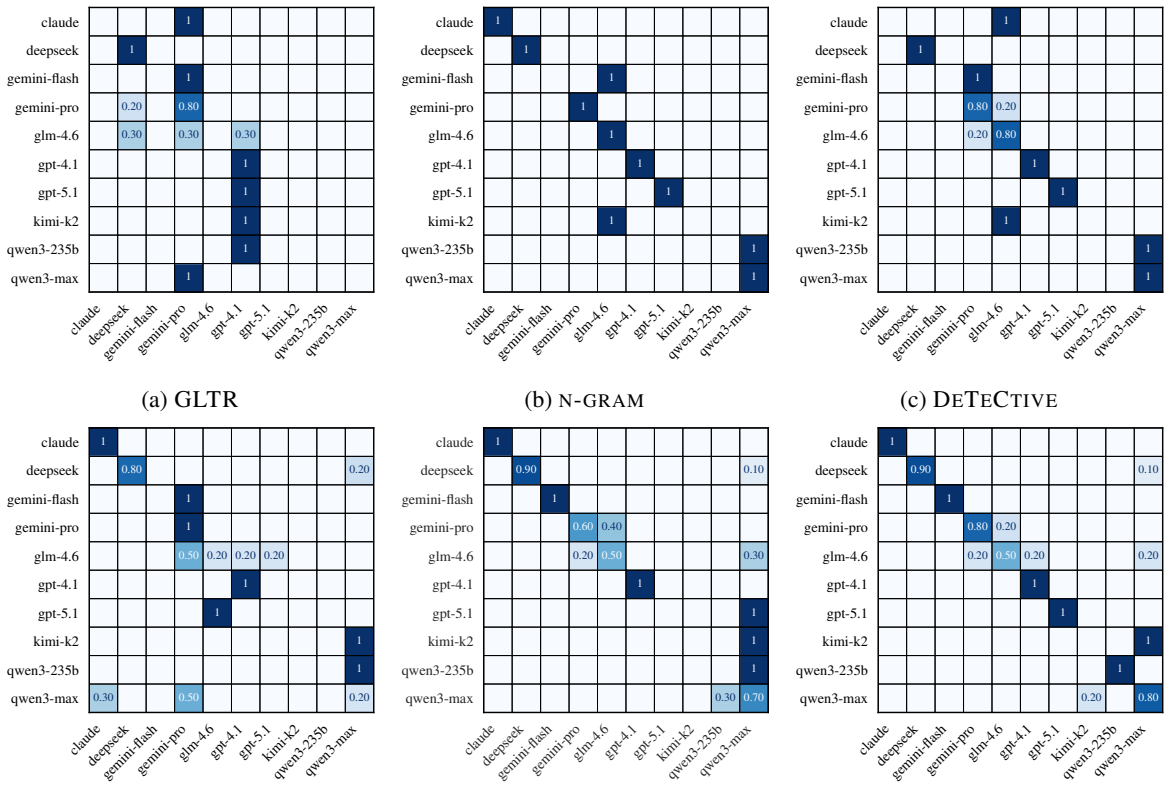
Figure 8: The impact of different orders used in TRACE fingerprints.

Scenario	Split 1	Split 2	Split 3
low-resource	gemini-flash	gemini-flash	gemini-pro
	glm-4.6	claude-sonnet	gemini-flash
	deepseek-chat	kimi-k2	qwen3-235b
	gpt-5.1	gpt-5.1	qwen3-max
	qwen3-235b	qwen3-235b	deepseek-chat
high-resource	gemini-pro	gemini-pro	glm-4.6
	claude-sonnet	deepseek-chat	claude-sonnet
	kimi-k2	gpt-4.1	kimi-k2
	gpt-4.1	glm-4.6	gpt-4.1
	qwen3-max	qwen3-max	gpt-5.1

Table 18: Random LLMs splits for low-resource and high-resource scenarios. These splits are released with the GHOSTWRITEBENCH dataset.

Method	Split 1	Split 2	Split 3
RANK	0.30	0.30	0.35
ENTROPY	0.30	0.25	0.25
GLTR	0.30	0.30	0.25
N-GRAM	0.40	0.75	0.40
BERT-AA	0.50	0.50	0.50
TOPFORMER	0.60	0.60	0.70
DETECTIVE	0.50	0.90	0.80
TRACE			
GPT-2			
Ranks	0.92	0.92	0.92
Entropy-JS	0.95	0.93	0.94
Entropy-Norm	-4.0	-4.0	-4.5
OLMo			
Ranks	0.92	0.92	0.92
Entropy-JS	0.94	0.93	0.95
Entropy-Norm	-3.5	-4.0	-3.5
Gemma			
Ranks	0.92	0.91	0.92
Entropy-JS	0.94	0.94	0.95
Entropy-Norm	-2.75	-2.50	-3.25

Table 19: Threshold (tuned as per development set) for different splits (from Table 18) and attribution methods used in this work. 'Norm' represents the norm-mean metric (as angle) as defined Appendix B.1 and for this lower is better (hence negative values reported). All norm-mean values are multiplied by 100 for presentation.



(a) GLTR (b) N-GRAM (c) DETECTIVE

(d) Ranks-JS (e) Entropy-JS (f) Entropy-Norm

Figure 9: Detailed normalised OOD-Domain confusion matrices for different techniques (in sub-captions) without selecting thresholds.

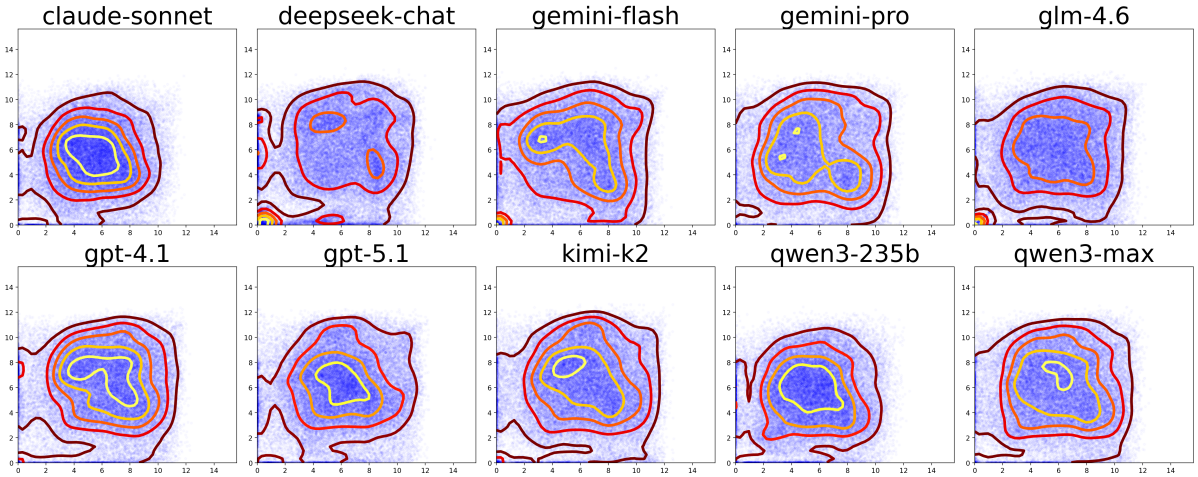


Figure 10: TRACE Entropy-based reference fingerprints learned for different LLMs (in subcaption) in GHOSTWRITEBENCH. Each axis represents the entropy of consecutive token transitions estimated by an independent language model; denser regions indicate more frequent transition types. We note distinct structural patterns for each LLM, suggesting they capture their generation styles.

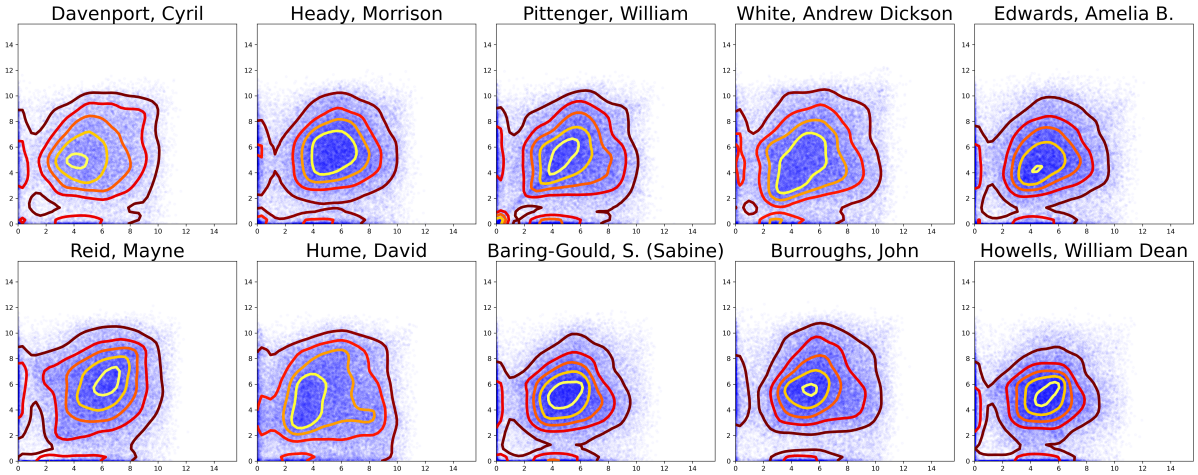


Figure 11: TRACE Entropy-based reference fingerprints (which are very different from LLM counterparts in Figure 10) learned for different human authors (in subcaption; sourced from Project Gutenberg) used for unseen authors ablation study Section 6.1.

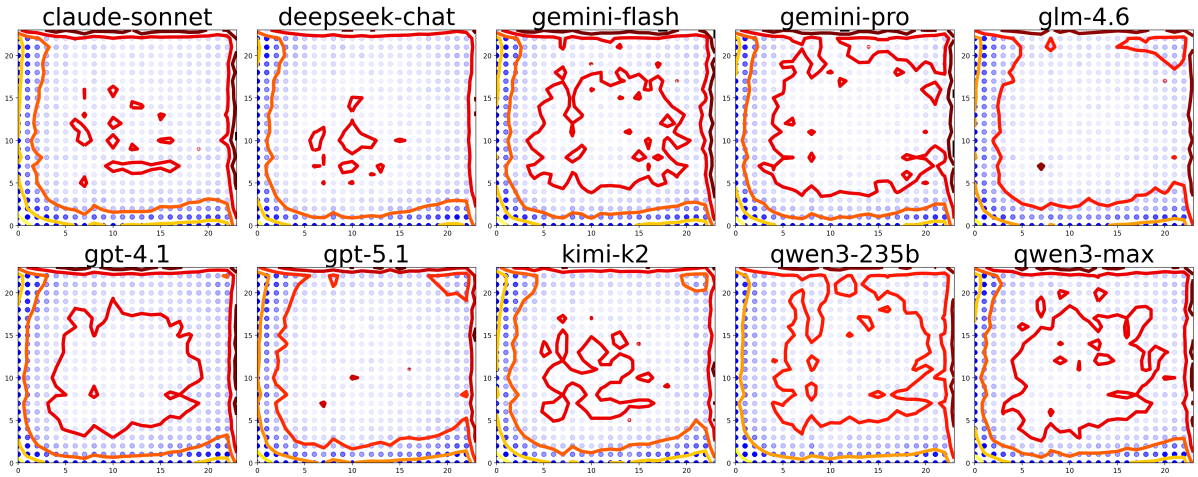


Figure 12: TRACE Rank-based reference fingerprints (in log-scale) learned for different LLMs (in subcaption). Although not as pronounced as Entropy-based (as seen in Figure 10), some structural patterns are there for each LLM. Note: Rank-based fingerprints have a grid structure due to the discrete nature of rank compared to continuous (scatter plot) entropy values.

```
Below is the beginning part of a story:

---

{}

---

We are going over segments of a story sequentially to gradually update one comprehensive summary
of the entire plot. Write a summary for the excerpt provided above, make sure to include vital
information related to key events, backgrounds, settings, characters, their objectives, and
motivations. You must briefly introduce characters, places, and other major elements if they are
being mentioned for the first time in the summary. The story may feature non-linear narratives,
flashbacks, switches between alternate worlds or viewpoints, etc. Therefore, you should organize
the summary so it presents a consistent and chronological narrative. Despite this step-by-step
process of updating the summary, you need to create a summary that seems as though it is written
in one go. The summary could include multiple paragraphs.

Summary:
```

Figure 13: Generate Initial Summary Prompt, adapted from (Chang et al., 2024).

```
Below is a segment from a story:

---

{}

---

Below is a summary of the story up until this point:

---

{}

---

We are going over segments of a story sequentially to gradually update one comprehensive summary
of the entire plot. You are required to update the summary to incorporate any new vital
information in the current excerpt. This information may relate to key events, backgrounds,
settings, characters, their objectives, and motivations. You must briefly introduce characters,
places, and other major elements if they are being mentioned for the first time in the summary.
The story may feature non-linear narratives, flashbacks, switches between alternate worlds or
viewpoints, etc. Therefore, you should organize the summary so it presents a consistent and
chronological narrative. Despite this step-by-step process of updating the summary, you need to
create a summary that seems as though it is written in one go. The updated summary could include
multiple paragraphs.

Updated summary:
```

Figure 14: Generate Intermediate Summary Prompt, adapted from (Chang et al., 2024).

```
Below is a summary of part of a story:

---

{}

---

Currently, this summary contains {} words. Your task is to condense it to less than {} words. The
condensed summary should remain clear, overarching, and fluid while being brief. Whenever
feasible, maintain details about key events, backgrounds, settings, characters, their objectives,
and motivations - but express these elements more succinctly. Make sure to provide a brief
introduction to characters, places, and other major components during their first mention in the
condensed summary. Remove insignificant details that do not add much to the overall story line.
The story may feature non-linear narratives, flashbacks, switches between alternate worlds or
viewpoints, etc. Therefore, you should organize the summary so it presents a consistent and
chronological narrative.

Condensed summary (to be within {} words):
```

Figure 15: Summary Compression Prompt, adapted from (Chang et al., 2024).

```
Generate a detailed outline for a {}-word novel in your own style based on these genres: {} set in
the {} century. Include word counts and major plot points.

Below you are given some snippets of similar novels just for reference:

SNIPPET 1: {}
    .
    .
    .
SNIPPET 6: {}

Please insert the generated text between <START>...<END> tags. Any other message or metadata
outside these tags.
```

Figure 16: Book Outline Generation Prompt.

```
GOAL: Write a {} words novel in your own style based on these genres: {}. The novel is set in the
{} century. We are generating the complete novel sequentially and in parts.

TASK: Begin writing the novel by generating the initial segment (~{} words). Expand the narrative
comprehensively, including all relevant details, while strictly adhering to the word limit
specified in the outline. Do not rush through any milestone; it is acceptable to split the content
across multiple parts if needed to respect the word limits. Please enclose the generated text
strictly between <START> and <END> tags. Any other message or metadata outside these tags.

Please read the following text carefully:
Follow the given novel outline strictly (including word counts and major plot points).

OUTLINE:
{}


Below you are given some snippets of similar novels for reference:
SNIPPET 1: {}
    .
    .
    .
SNIPPET 6: {}
```

Figure 17: Book Generation Start Prompt.

```
GOAL: Write a {} words novel in your own style based on these genres: {}. The novel is set in the
{} century. We are generating the complete novel sequentially and in parts.

TASK: You are continuing the novel generation process. Generate the next segment (~{} words) of
the novel. Expand the narrative comprehensively, including all relevant details, while strictly
adhering to the word limit specified in the outline. Do not rush through any milestone; it is
acceptable to split the content across multiple parts if needed to respect the word limits. Please
enclose the generated text strictly between <START> and <END> tags. If novel is complete, add "END
OF NOVEL" at the end of the generated text. Any other message or metadata outside these tags.

INSTRUCTIONS:
1. Follow the novel outline strictly, including word counts and major plot points.
2. Maintain full continuity with the previous segment and the overall summary generated so far.
3. Use the previous segments and summary as reference for style, tone, and story consistency.
4. Draw inspiration from the provided snippet texts for style, but do not copy directly.

OUTLINE:
{}

ENDING OF PREVIOUS SEGMENT:
{}

SUMMARY OF NOVEL TILL NOW:
{}

REFERENCE SNIPPETS:
SNIPPET 1: {}
    .
    .
    .
SNIPPET 6: {}
```

Figure 18: Book Generation Incremental Prompt.