EVERY STEP COUNTS: DECODING TRAJECTORIES AS AUTHORSHIP FINGERPRINTS OF dLLMS

Qi Li Runpeng Yu Haiquan Lu Xinchao Wang[†]
National University of Singapore
liqi@u.nus.edu xinchao@nus.edu.sg

ABSTRACT

Discrete Diffusion Large Language Models (dLLMs) have recently emerged as a competitive paradigm for non-autoregressive language modeling. Their distinctive decoding mechanism enables faster inference speed and strong performance in code generation and mathematical tasks. In this work, we show that the decoding mechanism of dLLMs not only enhances model utility but also can be used as a powerful tool for model attribution. Specifically, by analyzing the decoding trajectory associated with a given response, we can effectively identify its source model, thereby helping to mitigate risks of harmful content caused by model misuse. A key challenge in this problem lies in the diversity of attribution scenarios, including distinguishing between different models as well as between different checkpoints or backups of the same model. To ensure broad applicability, we identify two fundamental problems: *what information to extract from the decoding trajectory, and how to utilize it effectively.* We first observe that relying directly on per-step model confidence yields poor performance. This is mainly due to the bidirectional decoding nature of dLLMs: each newly decoded token influences the confidence of other decoded tokens, making model confidence highly redundant and washing out structural signal regarding decoding order or dependencies. To overcome this, we propose a novel information extraction scheme called the *Directed Decoding Map (DDM)*, which captures structural relationships between decoding steps and better reveals model-specific behaviors. Furthermore, to make full use of the extracted structural information during attribution, we propose *Gaussian-Trajectory Attribution (GTA)*, where we fit a cell-wise Gaussian distribution at each decoding position for each target model, and define the log-likelihood of a trajectory under different distributions as the attribution score: if a trajectory exhibits higher log-likelihood under the distribution of a specific model, it is more likely to have been generated by that model. Extensive experiments under different settings validate the utility of our methods. Code is available here.

1 INTRODUCTION

Authorship attribution has long been an important problem in natural language processing, with wide applications in criminal investigations (Chaski, 2005; Koppel et al., 2008; Grant, 2020), tracking terrorist threat (Cafiero & Camps, 2023; Budryk, 2019), and social media protection (Hazell, 2023; Barbon Jr et al., 2017; Sinnott & Wang, 2021). In the era of large language models (LLMs), the value of attribution becomes even more prominent, as it promotes responsibility assignment and thus supports efforts to combat misinformation and fraudulent reviews generated by LLMs (McGovern et al., 2024; Li et al., 2023; Antoun et al., 2023; Lin et al., 2024). Typically, this task is approached as a binary classification problem (Fagni et al., 2021; Jawahar et al., 2020; Mitchell et al., 2023; Lin et al., 2024). Prior studies have explored distinguishing human-written text from LLM-generated text (Ji et al., 2024; Clark et al., 2021), while others have focused on detecting and attributing outputs from different LLMs (Li et al., 2023; McGovern et al., 2024). The latter, also known as Model Attribution (MA) (Antoun et al., 2023), is considerably more challenging due to the similarities in model architectures and training corpus across LLMs.

In this work, we provide the first study of the MA problem in a new class of large language models, namely discrete diffusion large language models (dLLMs) (Nie et al., 2025; Ye et al., 2025). Unlike autoregressive generation (OpenAI , 2024; Gemini Team, 2025; DeepSeek-AI, 2025), dLLMs treat generation as an iterative decoding process over discrete token sequences (Nie et al., 2025; Yu et al., 2025). At each decoding step, any position of the token sequence may be unmasked from a mask token into a concrete token. This decoding process naturally forms a trajectory and introduces dependencies across decoding steps. It also captures, in a finer-grained manner, the distinctive characteristics that different models exhibit when responding to the same text prompt. We show that such unique properties of dLLM decoding trajectories offer a novel basis for addressing the model attribution problem.

Specifically, to fully exploit the fine-grained structural information embedded in the decoding process, we identify two fundamental problems: *how to effectively extract information from the decoding trajectory, and how to utilize such information for model attribution.* To solve these problems, we firstly introduce the *Directed Decoding Map (DDM)* for information extraction. The core motivation behind DDM is that different models exhibit stable differences in how newly decoded tokens interact with previously decoded tokens during generation. By encoding whether a newly decoded token induces positive, negative, or mixed effects on the confidence of previously decoded tokens, as well as the direction of confidence changes for each decoded tokens, DDM transforms complex probabilistic dynamics into a structured representation. This representation reliably captures model-specific decoding characteristics and thereby provides a reliable basis for model attribution. Building on this, we propose a novel model attribution method named *Gaussian-Trajectory Attribution (GTA)*. In GTA, each model is queried with a local dataset to obtain its own collection of DDMs. Based on these DDMs, we fit cell-wise Gaussian distributions separately for each model. This procedure preserves the structural information encoded in DDMs and produces a compact probabilistic fingerprint of each model's decoding behavior. To attribute a target response, we compute the log-likelihood of its DDM under all the constructed model-specific distributions and assign it to the model with the highest likelihood.

We conduct extensive experiments under various settings, including distinguishing between different models as well as between different checkpoints or backups of the same model. We also investigate the influence of different token length and decoding strategies in dLLMs, along with ablation studies on both DDM and GTA. The results consistently demonstrate the strong capability of our method for attributing dLLMs. We summarize our contributions as follows:

- We present the first exploration of model attribution for dLLMs and introduce the use of their distinctive decoding trajectories for lightweight yet reliable attribution. To this end, we design an information extraction scheme named *Directed Decoding Map (DDM)*, which reliably captures the structural and dependency information embedded in the decoding trajectory for a better attribution process.
- We propose a model attribution method named *Gaussian-Trajectory Attribution (GTA)*, which builds compact probabilistic fingerprints for each model via cell-wise Gaussian fitting and attributes a target response to the model with the highest likelihood.
- Extensive experiments across different dLLMs and various attribution settings clearly demonstrate the superiority of DDM and GTA. For example, even in the highly restrictive case where two models are fine-tuned from the same checkpoint using identical configurations, the attribution AUC remains above 81%.

2 Related Works

2.1 Discrete Diffusion Large Language Models (dLLMs)

Discrete Diffusion Large Language Models (dLLMs) (Nie et al., 2025; Ye et al., 2025; Yu et al., 2025) recently emerges as a promising paradigm for non-autoregressive (non-AR) language modeling. In tasks such as code generation (DeepMind, 2025; Inception Labs, 2025), planning (Ye et al., 2025), and Sudoku (Ye et al., 2025), dLLMs have been widely shown to achieve better performance than AR models. In contrast to AR generation, dLLMs treat generation as an iterative decoding process over discrete token sequences (Nie et al., 2025; Ye et al., 2025). This paradigm removes the left-to-right constraint, allowing parallel and structurally controllable generation with bidirectional attention. As a result, dLLMs can continuously update their contextual understanding during generation, enabling adaptive comprehension beyond the static, one-pass nature of AR models.

Due to the bidirectional nature of dLLMs, their iterative decoding process can be naturally viewed as a trajectory with strong structural information and contextual dependency. The use of historical information from the decoding process has also inspired some current efforts. For instance, DIJA (Wen et al., 2025) and PAD (Zhang et al., 2025) reformulates conventional jailbreak prompts into an interleaved mask-text format, compelling the model to generate unsafe outputs while maintaining contextual consistency. Another work (Xie et al., 2025) shows that dLLMs are more vulnerable to manipulation at the middle of the response than at the initial tokens and proposes MOSA, a reinforcement learning alignment strategy, which requires the model's generated middle tokens to align with a set of predefined safe tokens. In this work, we make the first attempt to use the history trajectory for model attribution. Rather than modifying it as in prior work, we regard the trajectory as a holistic signal and extract from it a representation that highlights model-specific information.

2.2 Authorship Attribution

Authorship attribution is a long-standing problem that was initially applied to distinguishing between human authors, with practical applications such as criminal investigations (Chaski, 2005; Koppel et al., 2008; Grant, 2020) and counterterrorism efforts (Cafiero & Camps, 2023; Budryk, 2019). With the rise of large language models (LLMs), and inspired by the Turing Test (Turing, 2007; Biever, 2023; Mei et al., 2024), research has expanded to distinguishing human-written from machine-generated text (Lin et al., 2024). Furthermore, model attribution (MA) further extends the concept of authorship attribution to the machine–machine setting (Li et al., 2023; Lin et al., 2024; Antoun et al., 2023), where the objective is to identify which specific model, or even which version of a model, was responsible for generating a given text.

Existing model attribution methods can be broadly grouped into three categories: statistical feature-based methods (Sharma et al., 2018; Sari et al., 2017; Shrestha et al., 2017; Proisl et al., 2018), which rely on measures such as perplexity, n-grams, and entropy and are lightweight but often limited in performance; classifier-based methods (Solorio et al., 2011; Shao et al., 2019), which train discriminative models on texts from different sources and achieve higher accuracy but at the cost of efficiency and robustness to adversarial mimicking; and watermark-based methods (Kirchenbauer et al., 2023; Boenisch, 2021), which take a pre-hoc approach by embedding watermarks in the generation process and later verifying attribution by detecting these watermarks. In this work, we design a lightweight and reliable statistical feature-based method, for the first time leveraging the unique decoding trajectory of dLLMs to achieve effective model attribution.

3 Methods

In this section, we first introduce our proposed information extraction scheme. The core intuition is that relying solely on first-order signals such as model confidence is insufficient to capture the structural information and cross-step dependencies embedded in the decoding trajectories of dLLMs. To address this, we propose a second-order representation, the *Directed Decoding Map (DDM)* in Section 3.1, which explicitly encodes the interdependencies among tokens and steps during decoding into a structured representation. Building on this, we further present our attribution method, *Gaussian-Trajectory Attribution (GTA)* in Section 3.2, which fully leverages the information in DDMs by fitting cell-wise Gaussian distributions over the extracted trajectories, thereby obtaining a compact probabilistic fingerprint of each model's decoding behavior and enabling lightweight yet reliable model attribution.

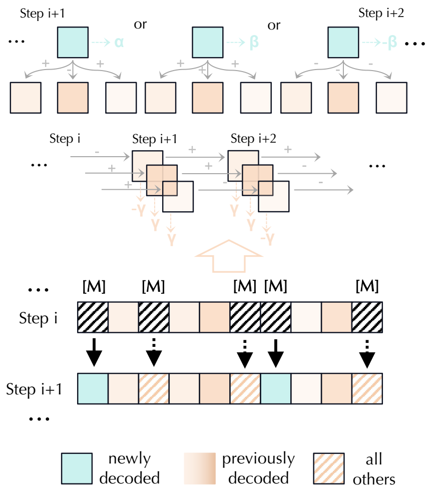
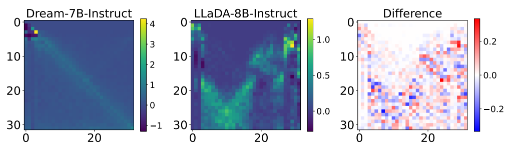


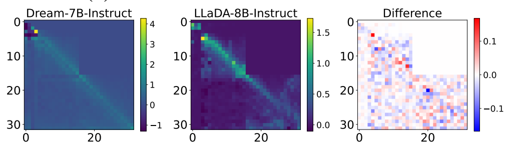
Figure 1: The DDM construction pipeline. [M] is the mask token.

3.1 Directed Decoding Map Construction

Consider a dLLM decoding process of T steps producing a sequence of L tokens. We define $c_i(j)$ as the confidence of position j at step i (if j is not yet decoded at step i, then $c_i(j) = \varnothing$). The confidence change of a position in two consecutive steps is defined as $\Delta c(j) = c_{i+1}(j) - c_i(j)$.



(a) Low-confidence with 32 tokens.



(b) Semi-supervised with 2 blocks and 32 tokens.

Figure 2: DDMs under two decoding strategies: *low-confidence decoding* and *semi-supervised decoding*. The models are instruction-tuned on GSM8K (Cobbe et al., 2021).

Let $U_i = \{j \mid c_i(j) \neq \varnothing\}$ be the set of positions already decoded in step i, and $N_{i+1} = U_{i+1} \setminus U_i$ denotes the set of token positions newly decoded at step $i+1$. We define the Directed Decoding Map entry at the i-th row, j-th position as $E_i(j)$. Each position falls into one of three categories: newly decoded positions $E_i(n)$, previously decoded positions $E_i(p)$, and all other positions $E_i(o)$. As a starting point, the values in the first row $E_1(j)$ are set to 0.

As illustrated in Figure 1, for each newly decoded position $n \in N_{i+1}$, the change it induces on the confidence of previously decoded positions $p \in U_i$ may exhibit three possible patterns: (i) all $\Delta c(p)$ are non-negative (confidence consistently increases), (ii) all $\Delta c(p)$ are non-positive (confidence consistently decreases), or (iii) a mixture of increases and decreases occurs. We define two distinct effect values $\alpha, \beta \in \mathbb{R}^+,\ \alpha \neq \beta$, and assign the effect value for token $n \in N_{i+1}$ as

$$E_{i+1}(n) = \begin{cases} \alpha, & \exists p, p' \in U_i : \Delta c(p) > 0,\ \Delta c(p') < 0, \\ \beta, & \forall p \in U_i : \Delta c(p) \geq 0, \\ -\beta, & \forall p \in U_i : \Delta c(p) \leq 0. \end{cases} \tag{1}$$

This design captures whether the effect of a new token is mixed, purely positive, or purely negative. For each previously decoded token p, once it has been decoded, subsequent steps may either reinforce its confidence or diminish it. Hence, for $p \in U_i$, we assign

$$E_{i+1}(p) = \begin{cases} \gamma, & \Delta c(p) > 0, \\ -\gamma, & \Delta c(p) < 0, \end{cases} \tag{2}$$

where $\gamma \in \mathbf{R}^+, \gamma \neq \alpha \neq \beta$ is another effect value. This rule explicitly encodes whether the stability of an already decoded token is being strengthened or weakened at step $i+1$. Finally, positions that remain masked carry no effect signal in this step. Thus, for any $o \notin U_{i+1}$, we set $E_i(o) = 0$. The final DDM is represented as a matrix $E \in \mathbb{R}^{T \times L}$, where each entry $E_i(j)$ reflects not only the decoding state but also the directionality of token-level influence across steps. Figure 2 shows DDMs for attribution between two models under two decoding strategies (Nie et al., 2025). For a fair comparison, both models (LLaDA-7B-Instruct (Nie et al., 2025) and Dream-8B-Instruct (Ye et al., 2025)) are instruction-tuned on GSM8K (Cobbe et al., 2021) with identical configurations, and the token number is set to 32. The effect values α, β, γ are set to 10, 0.5, 2, respectively (we report an ablation study on the effect values in Section 4, which shows that the actual effect value does not substantially affect the performance). As shown, even for different models under the same decoding strategy, DDMs can successfully capture structural differences between them, providing a reliable signal for attribution.

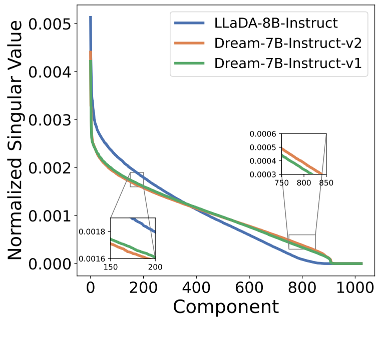


Figure 3: A SVD analysis on the structural information of DDMs.

3.2 Gaussian-Trajectory Attribution

While the Directed Decoding Map (DDM) provides a structured representation of inter-step dependency, a central challenge remains: *how to effectively preserve and utilize these signals for attribution across models*. To better illustrate the importance of preserving the structural information in DDMs, we perform a SVD analysis (Golub & Van Loan, 2013) (more details are given in Appendix A.3). Specifically, DDMs are flattened, concatenated into a matrix, mean-centered across features, and then decomposed via SVD. The resulting singular value spectrum characterizes how the variance of DDMs distributes across principal components. In Figure 3, the two versions of Dream-7B-Instruct are tuned from the same checkpoint under identical training configurations. As can be observed, different models exhibit almost identical spectra on the leading components, indicating that they share similar task-level and model-agnostic structures. However, clear discrepancies emerge in the middle and tail components, which correspond to low-energy yet highly discriminative directions. This observation highlights that *attribution signals are primarily encoded in the fine-grained structural patterns rather than in the dominant modes.* Consequently, applying dimensionality-reduction methods or highly noisy methods that retain only the leading components would inevitably discard these critical differences and significantly weaken attribution performance. To this end, we propose *Gaussian-Trajectory Attribution (GTA)*. By fitting Gaussian distributions to the cell-wise values of the DDM, GTA helps to obtain a compact probabilistic fingerprint that captures both local and global variation, enabling reliable model attribution.

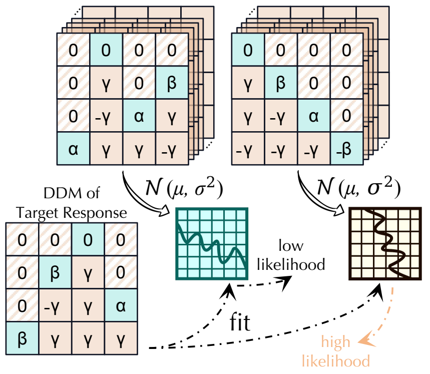


Figure 4: The GTA construction and attribution pipeline.

Let a decoding trajectory be represented as a DDM matrix $E \in \mathbb{R}^{T\times L}$, where T is the number of decoding steps and L is the output token length. As illustrated in Figure 4, for each target model M, we query it with a local dataset to obtain N trajectories $\{E^{(n)}\}_{n=1}^{N}$. These trajectories are then used to fit an independent Gaussian distribution for each cell (t, l) by computing the empirical mean and variance across samples:

$$\mu_M(t,l) = \frac{1}{N}\sum_{n=1}^{N} E_{t,l}^{(n)}, \qquad \sigma_M^2(t,l) = \frac{1}{N}\sum_{n=1}^{N}\left(E_{t,l}^{(n)} - \mu_M(t,l)\right)^2. \tag{3}$$

After the cell-wise Gaussian construction, given a DDM of target response $E^* \in \mathbb{R}^{T\times L}$, its log-likelihood under model M can be formalized as:

$$\ell_M(E^*) = -\tfrac{1}{2}\sum_{t=1}^{T}\sum_{j=1}^{L}\left[\frac{\left(E_{t,l}^{*} - \mu_M(t,l)\right)^2}{\sigma_M(t,l)^2} + \log\bigl(2\pi(\sigma_M(t,l)^2)\bigr)\right]. \tag{4}$$

For a candidate set of target models $\{M_1, \ldots, M_K\}$, we compute the log-likelihood scores $\ell_{M_k}(E^*)$ for each M_k. The attribution decision is then made by comparing likelihoods:

$$\hat{M}(E^*) = \arg\max_{M_k} \ \ell_{M_k}(E^*), \tag{5}$$

i.e., we attribute the trajectory to the model under which it achieves the highest likelihood. GTA is lightweight yet powerful: (i) it preserves the structural information encoded by the DDM without collapsing it into coarse statistics, (ii) it produces compact probabilistic fingerprints that are easily comparable across models, and (iii) it enables fine-grained attribution even among models with similar architectures or training corpora.

4 Experiments

4.1 Experimental Setup

Models and Datasets. In our experiments, we consider LLaDA-8B-Instruct (Nie et al., 2025) and Dream-7B-Instruct (Ye et al., 2025), along with multiple variants constructed under different attribution setups. For dLLMs, direct attribution across distinct models is relatively straightforward due to their inherently different decoding strategies. To establish a fairer and more challenging comparison, we additionally instruction-tune both models on identical datasets with the same training configuration (details are given in Appendix A.4). For tuning and evaluation, we use GSM8K (Cobbe et al.,

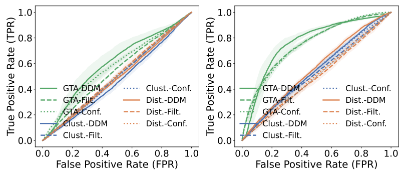
2021) for math reasoning and CodeAlpaca-20K (Chaudhary, 2023) for code generation, following the LLaDA training pipeline (Nie et al., 2025). Each dataset is split 7:3, with the larger part for training and the smaller for the local dataset. We adopt two decoding strategy: *Low-confidence* decoding and *Semi-supervised* decoding (Nie et al., 2025; Ye et al., 2025), and ensure that each model under attribution uses the same strategy. We use ⟨#tokens, block size⟩ to denote the combination of token length and block size. We consider five combinations: ⟨32, 32⟩ and ⟨64, 64⟩ for Low-confidence decoding and ⟨32, 16⟩, ⟨64, 32⟩, and ⟨64, 16⟩ for Semi-supervised decoding.

Table 1: Attribution Results under the three different setups. Semi-supervised decoding (Nie et al., 2025) is used as the decoding strategy here, where the token length is set to be 32 and the block size is 16. Within each method, the best-performing information representation scheme is underlined, while the column-wise best results are highlighted in green.

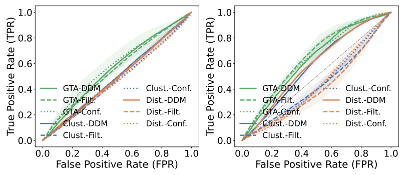
Scenario	Method	Information	GSM8K				CodeAlpaca-20K			
			AUC	TPR@5%FPR	TPR@1%FPR	Acc.	AUC	TPR@5%FPR	TPR@1%FPR	Acc.
CMA	Perplexity		69.52	16.73	11.11	72.08	69.66	23.16	11.06	65.37
	Clustering	confidence	70.68	16.41	4.10	67.22	70.11	33.94	16.30	67.40
		filtered confidence	69.68	21.01	3.52	64.83	63.33	12.75	2.07	61.58
		DDM	97.34	94.38	90.37	95.47	97.83	92.88	71.53	94.12
	Distance	confidence	99.38	98.22	87.87	96.94	89.81	33.54	13.50	84.53
		filtered confidence	99.85	99.91	98.66	98.91	89.59	42.73	8.44	82.60
		DDM	99.92	99.96	98.93	99.04	97.47	84.98	64.19	92.35
	GTA	confidence	99.43	99.38	99.33	99.44	96.45	86.07	35.04	91.85
		filtered confidence	99.66	99.02	95.67	97.99	95.66	84.22	65.11	90.35
		DDM	99.95 (↑ 30.43)	99.96	99.85	99.84	98.94 (↑ 35.61)	98.22	92.03	97.15
IRA	Perplexity		51.58	1.69	0.45	56.09	41.72	2.16	0.57	50.02
	Clustering	confidence	52.23	6.24	1.52	52.70	50.60	4.62	0.67	51.06
		filtered confidence	60.22	7.23	1.92	59.10	51.84	12.59	6.64	51.59
		DDM	61.15	7.00	1.38	59.48	53.82	6.71	2.16	52.93
	Distance	confidence	68.31	20.03	5.40	62.85	54.98	8.27	2.29	54.22
		filtered confidence	68.91	22.84	7.00	64.27	55.18	6.27	1.07	54.64
		DDM	79.49	33.05	19.18	72.26	58.73	10.72	1.24	57.01
	GTA	confidence	76.79	22.30	3.79	69.13	58.31	6.34	2.39	57.28
		filtered confidence	80.35	31.76	5.17	73.26	54.98	8.27	2.29	54.22
		DDM	81.75 (↑ 30.37)	38.22	3.84	74.00	65.05 (↑ 23.33)	14.35	3.72	60.88
CCA	Perplexity		49.32	6.33	1.20	52.03	40.89	4.09	0.78	50.54
	Clustering	confidence	49.90	3.08	0.58	52.12	51.06	4.70	0.83	51.29
		filtered confidence	51.03	6.47	1.61	52.05	51.03	4.57	0.70	51.14
		DDM	56.33	7.63	1.56	54.57	53.79	7.51	1.94	53.49
	Distance	confidence	61.79	5.71	1.83	59.03	59.36	5.83	1.41	60.84
		filtered confidence	57.34	6.74	1.56	56.47	54.68	9.53	2.39	54.46
		DDM	65.84	8.61	1.43	61.24	59.47	11.15	3.05	61.14
	GTA	confidence	64.50	13.60	3.70	60.79	53.86	5.83	1.04	53.19
		filtered confidence	59.53	8.83	1.87	58.14	54.13	6.03	1.02	53.36
		DDM	66.91 (↑ 17.59)	14.50	4.68	64.92	62.64 (↑ 21.75)	6.38	1.31	61.78

Attribution Scenario. We consider three challenging yet important attribution scenarios: **(i)** across different models (e.g., LLaDA-8B-Instruct vs. Dream-7B-Instruct), referred to as Cross-Model Attribution (CMA); **(ii)** between independent runs of the same model initialized from the same checkpoint and tuned using identical training configuration, referred to as Independent-Run Attribution (IRA); and **(iii)** across checkpoints of the same training trajectory at different epochs, referred to as Cross-Checkpoint Attribution (CCA).

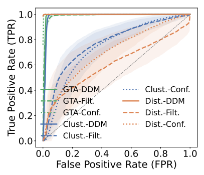
Baseline Methods. We setup several baseline methods that can be fairly compared in our setting. For information extraction, *Confidence* directly uses the model's predicted probabilities over all positions at each decoding step, without distinguishing between decoded and masked tokens. *Filtered Confidence*, in contrast, leverages the decoded tokens and mask out unfinished positions, thereby retaining only the confidence scores of tokens that have been actually generated. For attribution method, *Clustering* applies unsupervised clustering (in our case, we use DBSCAN (Schubert et al., 2017) with Euclidean distance, an epsilon of 0.8, and a minimum of 20 points to form a cluster) over the trajectory features of responses from different models, and then uses cluster proximity to assign attributing labels. *Distance* computes the Euclidean distances between each target response and the average representations of each model, and uses the relative distance margins as the attribution score. *Perplexity (PPL)* (Alon & Kamfonas, 2023; Ankner et al., 2024) further aggregates



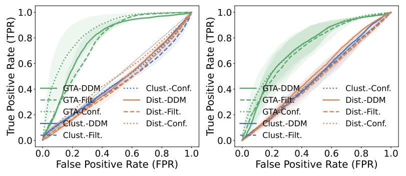
(a) CCA-GSM8K



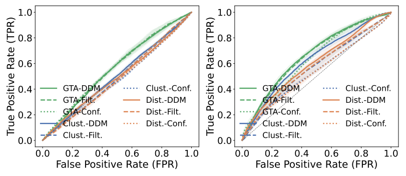
(b) CCA-CA20K



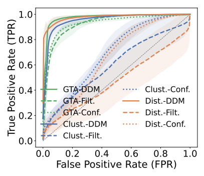
(c) CMA-GSM8K



(d) IRA-GSM8K



(e) IRA-CA20K



(f) CMA-CA20K

Figure 5: ROC curves of different attribution method–information extraction scheme combinations. *CA20K* abbreviates CodeAlpaca-20K. *Conf.*, *Filt.*, and *Clust.* stand for confidence, filtered confidence, and clustering. Attribution methods are distinguished by color, and extraction schemes by line style. Results are averaged over token lengths and decoding strategies. In subfigures (a)–(d), results on LLaDA are presented on the left, while those on Dream are on the right.

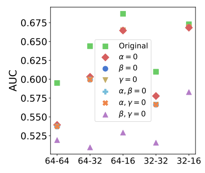
the predicted token probabilities into a score by computing the exponential of the average negative log-likelihood over the response.

Evaluation Metric. We adopt AUC score, TPR@Low% FPR, and Accuracy (Acc.) as evaluation metrics. AUC provides a holistic view of performance across different thresholds and serves as a key measure of attribution performance. TPR@Low% FPR highlights performance under stringent false-positive control, which is particularly important in attribution tasks where incorrect attributions can incur high costs. Accuracy offers an intuitive measure of overall correctness and complements of each method.

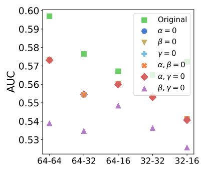
4.2 Main Results

We first provide a global overview of the performance trends across different combinations of attribution methods and information extraction schemes. The results are summarized in Table 1, where we follow prior work by formulating the problem as a binary classification task, i.e., attributing between two models. The token length is set to 32 and the block size to 16. Several key observations can be drawn from the results: **(i).** Among the three attribution scenarios, CMA proves to be the easiest, while CCA is the most challenging. This aligns with intuition: differences in decoding behavior across different models (CMA) are generally more pronounced than those between checkpoints or backups of the same model. Moreover, in CCA, the compared models can be regarded as one model being further fine-tuned from the other, whereas in IRA the compared models share the same initialization and undergo a fine-tuning process with same configuration. Consequently, the inter-model gap in CCA is usually smaller than that in IRA. **(ii).** Across different methods, DDM consistently demonstrates superior performance, yielding roughly a 10% AUC improvement over others under almost all settings. **(iii).** GTA achieves consistently stronger attribution performance than alternative methods, with the combination of GTA and DDM (i.e., our attribution method) delivering the best results. In all settings, this method improves AUC by 20–30% compared to PPL.

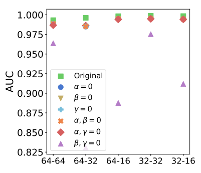
In addition, to provide a more intuitive comparison, we report the ROC curves of different methods under various settings. Results are given in Figure 5, where *CA20K* denotes the abbreviation of CodeAlpaca-20K. *Conf.*, *Filt.*, and *Clust.* denote confidence, filtered confidence, and clustering, respectively. Each attribution method is represented by a distinct color, while each information extraction scheme is indicated by a specific line style. Within subfigures (a), (b), (d) and (e), the left panel shows the results under LLaDA, while the right panel corresponds to Dream. The shaded regions indicate the variance across five token length and decoding strategy settings we consider (see Section 4.1), As can be observed, all three information extraction schemes under GTA outperform the baselines, and DDM consistently enables more stable attribution. Specifically, in the more challenging CCA and IRA settings, nearly all baselines perform close to random guessing. In contrast,



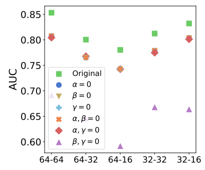
(a) CCA-GSM8K



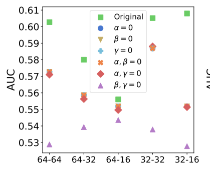
(b) CCA-CA20K



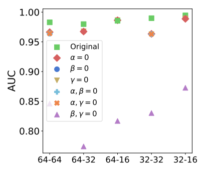
(c) CMA-GSM8K



(d) IRA-GSM8K



(e) IRA-CA20K



(f) CMA-CA20K

Figure 6: This experiment reveals two key aspects: (i) the impact of zeroing out specific effect values, and (ii) the effect of decoding strategy and token length on performance. In subfigures (a)–(d), results on LLaDA are presented on the left, while those on Dream are on the right.

for the relatively easier CMA setting, some baselines achieve moderate AUC but remain suboptimal; only a few methods combined with DDM, such as Dist.-DDM on GSM8K and Dist.-DDM / Clust.-DDM on CodeAlpaca-20K, attain strong performance. These results highlight the robustness and effectiveness of DDM and GTA across diverse scenarios.

4.3 Ablation Study

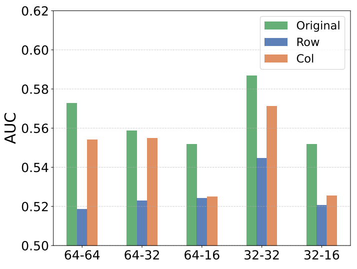


Figure 7: Maintaining the structural integrity of DDMs is critical for reliable attribution.

Structural Information of DDM. We conduct an experiment to investigate the importance of the structural information extracted by DDM from decoding trajectories. Specifically, we zero out each of the three effect values individually and in pairs, resulting in six different variants. As shown in Figure 6, we evaluate these variants along with the original DDM (denoted as Original) under five combinations of token length and decoding strategy, with different colors representing different variants.

Several insights can be drawn: (i) among the three effect values, γ plays the most critical role, followed by β. In particular, when both β and γ are zeroed out, the performance drops drastically. In other cases, performance still degrades but the decline is less severe and relatively consistent. γ corresponds to the trajectory of the confidence variations of previously decoded tokens $E_i(p)$ at each decoding step, while β represents the trajectory of the one-way influence exerted by the newly decoded token $E_i(n)$ on $E_i(p)$.

The substantial contribution of these two components indicates that DDM's effectiveness relies heavily on modeling both the historical semantic accumulation of tokens (captured by γ) and the interaction between newly decoded and previously decoded tokens (captured by β). In other words, DDM goes beyond local confidence signals by leveraging structural and dependancy information from the decoding trajectory, which is key to its superior attribution performance.

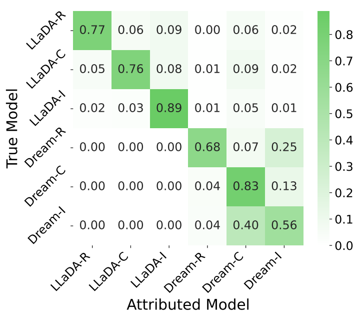


Figure 8: Attribution across multiple models conducted within a single attribution process.

Influence of Decoding Strategy and Token Length. Besides the influence of effect values, Figure 6 also illustrates the impact of different decoding strategies and token lengths. Focusing on the results of the original DDM (green square), we observe that for CMA (Figures 6c, 6f), these factors have little effect on performance. In contrast, for CCA (Figures 6a, 6b) and IRA (Figures 6d, 6e), performance varies slightly across settings and no consistent pattern emerges, with AUC changes generally within 0.1. Considering that token length and decoding strategies can be flexibly chosen in practice, and that our experiments show their influence on performance to be small, the feasibility of applying DDM in attribution is thus well supported.

Structure Preservation of GTA. A primary motivation behind the design of GTA is to better preserve the structural information in DDMs. To assess the necessity and impact of our design, we conduct a comparative study in Figure 7, where the cell-wise Gaussian distribution in GTA is replaced with token-wise (column-based, denoted as Col) and step-wise (row-based, denoted as Row) variants. The results show that when structural information is disrupted, performance drops significantly, particularly in step-wise Gaussian (Row), where the interdependence among tokens is severely undermined, leading to a partial failure of attribution. This further highlights the utility of GTA's design.

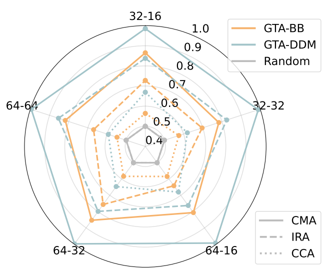


Figure 9: The performance of GTA under black-box setting.

Attribution across Multiple Models. Although model attribution is typically formulated as a binary classification task, performing attribution across multiple models simultaneously provides a stronger validation of a method's scalability. In Figure 8, we conduct attribution over six model variants used in this work. We use GSM8K as the dataset in this experiment, with the token length and decoding strategy set to 64 and 32, respectively. Here, **-R* denotes the reference model, **-C* the model compared with **-R* in CCA, and **-I* the model compared with **-R* in IRA. The diagonal entries indicate cases where the predicted model matches the ground-truth model, i.e., attribution is correct. We observe that in most cases, the combination of GTA and DDM consistently achieve reliable attribution, and the limited deviations that arise are predominantly restricted to models within the same family.

Influence of Effect Values. Besides the default value of α, β and γ (10, 0.5, 2), we also experiment with other values to demonstrate the robustness of our method to such variations. As shown in Table 2, we vary the three effect values ($\alpha \in [5, 15]$ step 1; $\beta \in [0.1, 1.0]$ step 0.1; $\gamma \in [1.5, 2.5]$ step 0.1) and report the AUC std (%) with token length 64 on GSM8K under low-confidence decoding. The std values are mostly below 0.1% and at most around 1%, indicating that DDM and GTA is robust to changes in effect values.

Table 2: Performance variations under different effect values.

std (%)	∇CMA	∇IRA	∇CCA
$\alpha_{5\sim15}$	0.08	1.67	1.49
$\beta_{0.1\sim1.0}$	0.02	0.09	0.08
$\gamma_{1.5\sim2.5}$	0.03	0.86	0.60

GTA in Black-box Scenario. Previous confidence-based methods, including DDM, are operated in the *gray-box* setting, where the adversary lacks access to model parameters or intermediate features and can only rely on model outputs. We further push this boundary by evaluating GTA in the most strict scenario, namely the full *black-box* setting, where the adversary can only observe the final decoded tokens at each step without any confidence information. In this case, GTA constructs distributions directly from the model's decoding history. Results are reported in Figure 9, where GTA-BB refers to GTA under black-box setting. As shown, while the performance degrades, it remains considerable. This demonstrates that for dLLMs, the attribution problem is still solvable even under the strictest setting, with their unique decoding mechanism playing a key role in enabling this.

5 Conclusion

In this work, we take the first step toward exploring the problem of model attribution in dLLMs. As a new class of large language models, dLLMs exhibit distinctive decoding characteristics. Building on the decoding trajectories of dLLMs, we propose Directed Decoding Map (DDM), an information extraction scheme that captures the interdependencies among tokens across decoding steps. The strong structural properties of DDMs provide a clear reflection of model-specific behaviors. Furthermore, we introduce Gaussian Trajectory Attribution (GTA), which constructs cell-wise Gaussian distributions over DDMs, thereby preserving fine-grained structural information and producing a compact probabilistic fingerprint of each model. Extensive experiments across diverse settings demonstrate the efficacy of our method.

References

Gabriel Alon and Michael Kamfonas. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*, 2023.

Zachary Ankner, Cody Blakeney, Kartik Sreenivasan, Max Marion, Matthew L Leavitt, and Mansheej Paul. Perplexed by perplexity: Perplexity-based data pruning with small reference models. *arXiv preprint arXiv:2405.20541*, 2024.

Wissam Antoun, Benoît Sagot, and Djamé Seddah. From text to source: Results in detecting large language model-generated content. *arXiv preprint arXiv:2309.13322*, 2023.

Sylvio Barbon Jr, Rodrigo Augusto Igawa, and Bruno Bogaz Zarpelão. Authorship verification applied to detection of compromised accounts on online social networks: A continuous approach. *Multimedia Tools and Applications*, 76(3):3213–3233, 2017.

Celeste Biever. Chatgpt broke the turing test-the race is on for new ways to assess ai. *Nature*, 619 (7971):686–689, 2023.

Franziska Boenisch. A systematic review on model watermarking for neural networks. *Frontiers in big Data*, 4:729663, 2021.

Zack Budryk. Exclusive: Fbi document warns conspiracy theories are a new domestic terrorism threat. *Yahoo News*, August 2019.

Florian Cafiero and Jean-Baptiste Camps. Who could be behind qanon? authorship attribution with supervised machine-learning. *Digital Scholarship in the Humanities*, 38(4):1418–1430, 2023.

Carole E Chaski. Who's at the keyboard? authorship attribution in digital evidence investigations. *International journal of digital evidence*, 4(1):1–13, 2005.

Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca, 2023.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A Smith. All that's' human'is not gold: Evaluating human evaluation of generated text. *arXiv preprint arXiv:2107.00061*, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepMind. Gemini diffusion. https://deepmind.google/models/gemini-diffusion/, 2025. Accessed: 2025-06-16.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, and Maurizio Tesconi. Tweepfake: About detecting deepfake tweets. *Plos one*, 16(5):e0251415, 2021.

Gemini Team. Gemini: A family of highly capable multimodal models, 2025. URL https://arxiv.org/abs/2312.11805.

Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

Tim Grant. Text messaging forensics: Txt 4n6: idiolect-free authorship analysis? In *The Routledge handbook of forensic linguistics*, pp. 558–575. Routledge, 2020.

Julian Hazell. Spear phishing with large language models. *arXiv preprint arXiv:2305.06972*, 2023.

Inception Labs. Mercury. https://www.inceptionlabs.ai/introducing-mercury, 2025. Accessed: 2025-06-16.

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks VS Lakshmanan. Automatic detection of machine generated text: A critical survey. *arXiv preprint arXiv:2011.01314*, 2020.

Jiazhou Ji, Ruizhe Li, Shujun Li, Jie Guo, Weidong Qiu, Zheng Huang, Chiyu Chen, Xiaoyu Jiang, and Xinru Lu. Detecting machine-generated texts: Not just" ai vs humans" and explainability is complicated. *arXiv preprint arXiv:2406.18259*, 2024.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *International Conference on Machine Learning*, pp. 17061–17084. PMLR, 2023.

Moshe Koppel, Jonathan Schler, and Eran Messeri. Authorship attribution in law enforcement scenarios. *NATO Security Through Science Series D-Information and Communication Security*, 15: 111, 2008.

Linyang Li, Pengyu Wang, Ke Ren, Tianxiang Sun, and Xipeng Qiu. Origin tracing and detecting of llms. *arXiv preprint arXiv:2304.14072*, 2023.

Li Lin, Neeraj Gupta, Yue Zhang, Hainan Ren, Chun-Hao Liu, Feng Ding, Xin Wang, Xin Li, Luisa Verdoliva, and Shu Hu. Detecting multimedia generated by large ai models: A survey. *arXiv preprint arXiv:2402.00045*, 2024.

Hope McGovern, Rickard Stureborg, Yoshi Suhara, and Dimitris Alikaniotis. Your large language models are leaving fingerprints. *arXiv preprint arXiv:2405.14057*, 2024.

Qiaozhu Mei, Yutong Xie, Walter Yuan, and Matthew O Jackson. A turing test of whether ai chatbots are behaviorally similar to humans. *Proceedings of the National Academy of Sciences*, 121(9): e2313925121, 2024.

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

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

OpenAI . Gpt-4o system card, 2024. URL `https://arxiv.org/abs/2410.21276`.

Thomas Proisl, Stefan Evert, Fotis Jannidis, Christof Schöch, Leonard Konle, and Steffen Pielström. Delta vs. n-gram tracing: Evaluating the robustness of authorship attribution methods. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

Yunita Sari, Andreas Vlachos, and RM Stevenson. Continuous n-gram representations for authorship attribution. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, volume 2. ACL, 2017.

Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3):1–21, 2017.

Jialin Shao, Adaku Uchendu, and Dongwon Lee. A reverse turing test for detecting machine-made texts. In *Proceedings of the 10th ACM Conference on Web Science*, pp. 275–279, 2019.

Abhay Sharma, Ananya Nandan, and Reetika Ralhan. An investigation of supervised learning methods for authorship attribution in short hinglish texts using char & word n-grams. *arXiv preprint arXiv:1812.10281*, 2018.

Prasha Shrestha, Sebastian Sierra, Fabio A González, Manuel Montes, Paolo Rosso, and Thamar Solorio. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: Volume 2, short papers*, pp. 669–674, 2017.

Richard Sinnott and Zijian Wang. Linking user accounts across social media platforms. In *Proceedings of the 2021 IEEE/ACM 8th International Conference on Big Data Computing, Applications and Technologies*, pp. 18–27, 2021.

Thamar Solorio, Sangita Pillay, Sindhu Raghavan, and Manuel Montes. Modality specific meta features for authorship attribution in web forum posts. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pp. 156–164, 2011.

Alan M Turing. Computing machinery and intelligence. In *Parsing the Turing test: Philosophical and methodological issues in the quest for the thinking computer*, pp. 23–65. Springer, 2007.

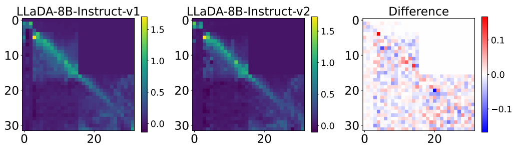
Zichen Wen, Jiashu Qu, Dongrui Liu, Zhiyuan Liu, Ruixi Wu, Yicun Yang, Xiangqi Jin, Haoyun Xu, Xuyang Liu, Weijia Li, Chaochao Lu, Jing Shao, Conghui He, and Linfeng Zhang. The devil behind the mask: An emergent safety vulnerability of diffusion llms, 2025.

Zhixin Xie, Xurui Song, and Jun Luo. Where to start alignment? diffusion large language model may demand a distinct position. *arXiv preprint arXiv:2508.12398*, 2025.

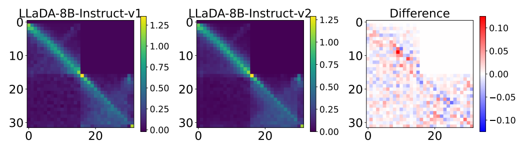
Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models, 2025. URL https://arxiv.org/abs/2508.15487.

Runpeng Yu, Qi Li, and Xinchao Wang. Discrete diffusion in large language and multimodal models: A survey. *arXiv preprint arXiv:2506.13759*, 2025.

Yuanhe Zhang, Fangzhou Xie, Zhenhong Zhou, Zherui Li, Hao Chen, Kun Wang, and Yufei Guo. Jailbreaking large language diffusion models: Revealing hidden safety flaws in diffusion-based text generation. *arXiv preprint arXiv:2507.19227*, 2025.

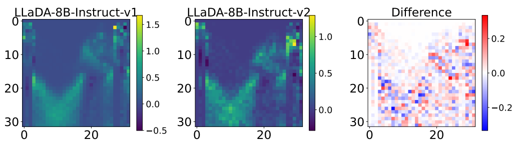


(a) GSM8K, ⟨32, 16⟩.

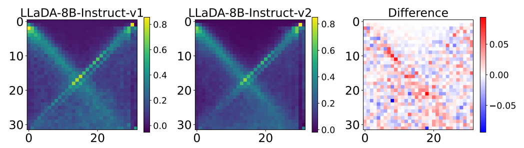


(b) CodeAlpaca-20K, ⟨32, 16⟩.

Figure 10: DDM comparison of LLaDA-8B-Instruct instruction-tuned under two different datasets. ⟨#tokens, block size⟩ is set to ⟨32, 16⟩.



(a) GSM8K, ⟨32, 32⟩.



(b) CodeAlpaca-20K, ⟨32, 32⟩.

Figure 11: DDM comparison of LLaDA-8B-Instruct instruction-tuned under two different datasets. ⟨#tokens, block size⟩ is set to ⟨32, 32⟩.

A Appendix

A.1 More DDM visualizations.

In Figures 10 11 12 13, we further provide several visualizations of DDMs under IRA setting, with the models to be attributed set to LLaDA-8B-Intruct and datasets used are GSM8K and CodeAlpaca-20K. All the five ⟨#tokens, block size⟩ settings are provided.

A.2 More results of Multiple Model Attribution

In Figure 17, we further provide several results under multiple models attribution. The dataset used here is GSM8K and ⟨#tokens, block size⟩ is set to ⟨32, 16⟩, ⟨32, 32⟩, ⟨64, 16⟩, ⟨64, 64⟩, respectively.

A.3 Another SVD Anylasis Under Semi-supervised Decoding.

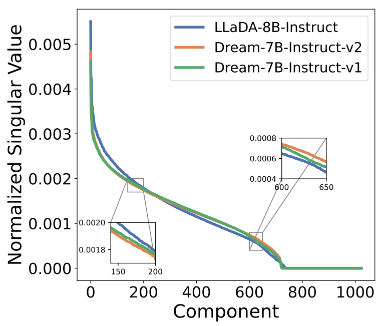
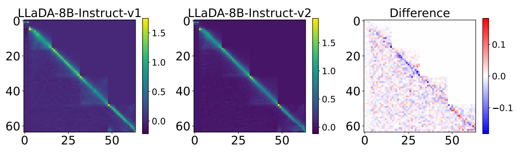


Figure 18: A SVD analysis on the structural information of DDMs.

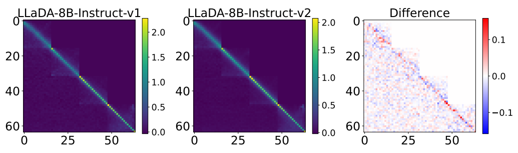
In the main text, we analyzed the structural information of DDM using SVD under low-confidence decoding (⟨#tokens, block size⟩ = ⟨32, 32⟩). Here, we provide another example under semi-supervised decoding with ⟨#tokens, block size⟩ = ⟨32, 16⟩, as shown in Figure 18. The same phenomenon is observed: different models share almost identical spectra in the leading components, reflecting similar task-level, model-agnostic structures, while clear discrepancies appear in the middle and tail components, which capture low-energy yet highly discriminative directions. This suggests that *attribution signals lie in fine-grained structural patterns rather than dominant modes*, and thus dimensionality reduction or noisy methods that retain only leading components would discard critical differences and weaken attribution.

A.4 Training Configurations and Prompts used in Our Work

In Figures 14 and 15, we provide prompt examples for the two datasets used in our work (GSM8K and CodeAlpaca-20K), where GSM8K is for mathematical reasoning task and CodeAlpaca-20K is for code generation task. In Table 3, we report the detailed training configurations of the models used in our paper.

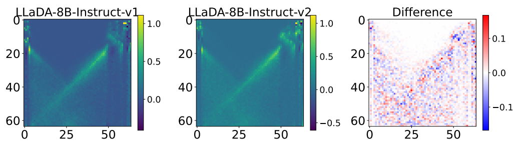


(a) GSM8K, ⟨64, 16⟩.

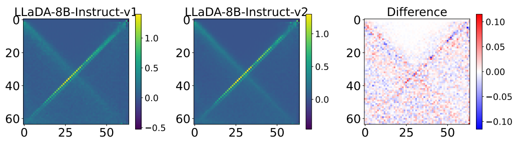


(b) CodeAlpaca-20K, ⟨64, 16⟩.

Figure 12: DDM comparison of LLaDA-8B-Instruct instruction-tuned under two different datasets. ⟨#tokens, block size⟩ is set to ⟨64, 16⟩.



(a) GSM8K, ⟨64, 64⟩.



(b) CodeAlpaca-20K, ⟨64, 64⟩.

Figure 13: DDM comparison of LLaDA-8B-Instruct instruction-tuned under two different datasets. ⟨#tokens, block size⟩ is set to ⟨64, 64⟩.

Setting	Value
Training Arguments	
Epochs	20
Batch size	1
Gradient accumulation	4
Logging steps	2
Max seq. length	4096
Save steps	100
Learning rate	1e-5
Weight decay	0.1
Max grad norm	1.0
Deepspeed Config	
Zero stage	2
Gradient accumulation	4
Gradient clipping	1.0
Zero3 init flag	False
Processes	8

Table 3: Training configuration.

```
User:
Respond in the following format:
<reasoning>
Your reasoning here
</reasoning>
<answer>
...
</answer>
Stefan goes to a restaurant to eat dinner with his family. They
order an appetizer that costs $10 and 4 entrees that are $20
each. If they tip 20% of the total for the waiter, what is the
total amount of money that they spend at the restaurant?

Assistant:
```

Figure 14: Prompt for GSM8K.

```
User:
Create a for loop that goes through every element of
list_of_words and prints 'success' if an element is equal to
"example" and prints 'failure' in any other case.

Assistant:
```

Figure 15: Prompt for CodeAlpaca-20K.

Figure 16: Training configuration (left) and prompt examples (right).

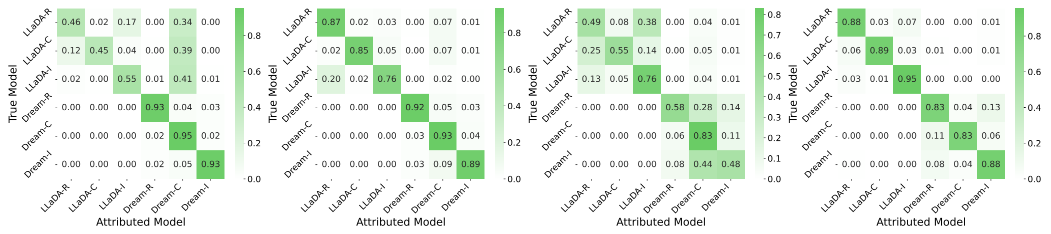


Figure 17: Attribution for multiple models under GSM8K. ⟨#tokens, block size⟩ is set to ⟨32, 16⟩, ⟨32, 32⟩, ⟨64, 16⟩, ⟨64, 64⟩, respectively.