

TDGNet: Hallucination Detection in Diffusion Language Models via Temporal Dynamic Graphs

Arshia Hemmat¹, Philip Torr¹, Yongqiang Chen^{2,3}, Junchi Yu^{1*}

¹University of Oxford,

²Carnegie Mellon University,

³Mohamed bin Zayed University of Artificial Intelligence

Abstract

Diffusion language models (D-LLMs) offer parallel denoising and bidirectional context, but hallucination detection for D-LLMs remains underexplored. Detectors developed for auto-regressive LLMs can be adapted to diffusion generation and remain competitive baselines, but they rely on single-pass cues and therefore leave the denoising trajectory unused, where factuality evidence is distributed across the denoising trajectory and may appear, drift, or be self-corrected over time. We introduce TDGNet, a temporal dynamic graph framework that formulates hallucination detection as learning over evolving token-level attention graphs. At each denoising step, we sparsify the attention graph and update per-token memories via message passing, then apply temporal attention to aggregate trajectory-wide evidence for final prediction. Experiments across four D-LLMs spanning dense and Mixture-of-Experts architectures (7B–16B parameters) and multiple diffusion paradigms show consistent AUROC improvements over output-based, latent-based, and static-graph baselines, with single-pass inference and modest overhead. These results highlight the importance of temporal reasoning on attention graphs for robust hallucination detection in diffusion language models.

1 Introduction

The auto-regressive large language models (AR-LLMs) have become the *de facto* paradigm across a wide range of applications, from natural language understanding (Brown et al., 2020; Ouyang et al., 2022; Achiam et al., 2023) to general-purpose task execution (Vaswani et al., 2017; Brown et al., 2020; Touvron et al., 2023). Recently, diffusion-based large language models (D-LLMs) (Nie et al., 2025; Ye et al., 2025; Sahoo et al., 2024) have emerged as a compelling alternative to their auto-regressive counterparts. In contrast to AR-LLMs,

which generate text sequentially via next-token prediction, D-LLMs leverage bi-directional attention and iteratively denoise the entire output sequence in generation. This formulation enables more flexible generation dynamics (Li et al., 2022; Mireshghallah et al., 2022) and can significantly improve inference efficiency through parallel decoding mechanisms (Zhang et al., 2025; Ghazvininejad et al., 2019). Recent representative models, such as LLaDA-8B (Nie et al., 2025) and Dream-7B (Ye et al., 2025), have demonstrated that D-LLMs can be successfully scaled to the billion-parameter scale with comparable performance to popular AR-LLMs of similar size across a wide range of diverse tasks (Nie et al., 2025; Ye et al., 2025).

As the capabilities of D-LLMs increase significantly, it becomes increasingly critical to ensure the generated content from D-LLMs is factually correct. However, the hallucination detection problem remains substantially less explored than that in AR-LLMs. Existing methods designed for AR models are fundamentally ill-suited for the diffusion paradigm due to two architectural mismatches. First, the *probabilistic intractability* of diffusion generation makes reliable single-pass confidence scores costly to obtain of the responses without expensive trajectory marginalization (Kuhn et al., 2023; Sahoo et al., 2024). Second, D-LLMs exhibit *temporal oscillation*, where predictions fluctuate between factual and hallucinated states during the iterative denoising process (Wang et al., 2026). Such a phenomenon demonstrates that hallucinations manifest as dynamic events in the generation process of D-LLMs, such as self-correction, late-stage drift, or persistence (Figure 1, illustrative), rather than static errors in the model responses. Thus, effective hallucination detection in D-LLMs stands to benefit from modelling the temporal evolution of the generation trajectory rather than inspecting isolated snapshots of the output (Chang et al., 2026).

* Corresponding author: junchi.yu@eng.ox.ac.uk
Project page: arshiahemmat.github.io/tdgnet

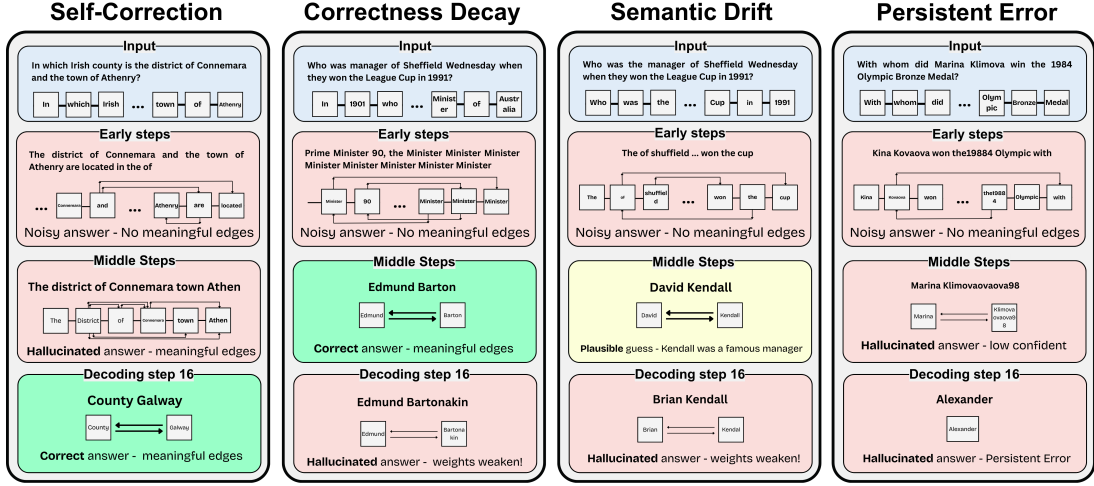


Figure 1: **Why Temporal Modelling is Essential.** Four diffusion dynamics that confound static detectors (case studies in Appendix F): **Self-Correction** (early noise resolves into fact), **Correctness Decay** and **Semantic Drift** (correct or plausible states degrade), and **Persistent Error** (errors lock in early). TDGNet aggregates trajectory-wide signals to distinguish these patterns. These four patterns are illustrative rather than exhaustive or mutually exclusive: a single trajectory may exhibit several of them, for instance drift followed by late self-correction, and they are intended to convey why a single snapshot is insufficient rather than to define a taxonomy.

Building on the observation that hallucinations in D-LLMs unfold over the denoising trajectory (Section 4.4), we focus on how such dynamics should be represented for effective detection. While one may attempt to model the generation process as a sequence of intermediate token states, this abstraction fails to capture the underlying structure in the intermediate contexts. Unlike AR-LLMs, D-LLMs rely on bi-directional attention to iteratively refine global context, which introduces evolving correlations among tokens during the generation process. This evolving pattern of token correlations suggests a representation that explicitly models both relational structure and temporal evolution.

In this work, we propose TDGNet, a temporal graph-based framework for hallucination detection in diffusion-based LLMs that explicitly models the evolving relational structure of the denoising process. The motivation is that hallucinations in D-LLMs arise from changes in token interactions over time, which cannot be reliably captured by sequence-based or snapshot-based representations. Specifically, TDGNet formulates hallucination detection as a temporal graph learning problem, where tokens are treated as nodes and their interactions evolve across denoising steps. TDGNet maintains a persistent memory state (Rossi et al., 2020) for each token that is updated through temporal message passing. This design enables the TDGNet to aggregate subtle signs of hallucina-

tion from the token dependency across the denoising generation process of D-LLMs. Moreover, TDGNet supports both sequence-level and token-level hallucination detection, enabling identification of hallucination at multiple granularities.

We evaluate our method on four open-source D-LLMs spanning dense and Mixture-of-Experts architectures (LLaDA-8B, Dream-7B, LLaDA 1.5, and LLaDA 2.1-mini) (Nie et al., 2025; Ye et al., 2025; Zhu et al., 2025; Bie et al., 2026), across multiple-choice, open-ended, and contextual QA benchmarks, using AUROC as the primary metric for hallucination detection. Our temporal graph detector consistently outperforms strong output-based, latent-based, and attention/graph-based baselines, on *all* datasets and on *both* models, and remains robust under different denoising schedules and graph-construction choices (Tables 1, 2, and 5). Our main contributions are:

- We analyse hallucination dynamics in D-LLMs and show that factuality signals are weak at any single denoising step and instead accumulate across the denoising trajectory (Section 4.4).
- We propose **TDGNet**, a temporal graph learning framework that tracks token interactions with persistent memories, enabling both response-level detection and token-level localisation in a unified model (Section 3).

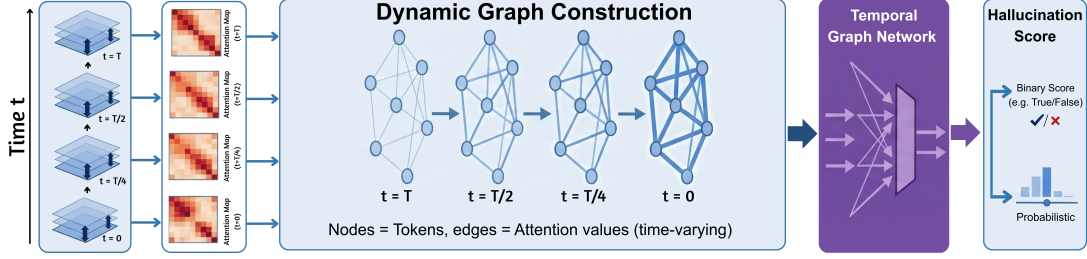


Figure 2: **Overview of the TDGNet framework.** The model extracts attention maps across diffusion denoising steps to construct a temporal dynamic graph, which is then processed by a Temporal Graph Network to predict hallucination probability.

- Across four D-LLMs spanning dense and MoE architectures (7B–16B parameters) on QA benchmarks, TDGNet achieves consistent AUROC improvements over output-based, latent-based, and graph-based baselines, supported by ablations and cross-architecture evaluations confirming that both attention-graph structure and temporal modelling are essential for strong performance (Section 4.4).

2 Related Work

Hallucination detection for LLMs is commonly grouped into (i) output-based approaches that use uncertainty or self-consistency signals (Chang et al., 2026; Kuhn et al., 2023; Lin et al., 2023; Kossen et al., 2024; Manakul et al., 2023), and (ii) latent-based approaches that probe internal representations to separate truthful from hallucinated generations (Chang et al., 2026; Azaria and Mitchell, 2023; Chen et al., 2024; Burns et al., 2022; Park et al., 2025; Orgad et al., 2025; Du et al., 2024; Su et al., 2024b). Retrieval- or tool-based factuality checking is also effective but relies on external knowledge sources, falling outside our internal-signal detection setting (Min et al., 2023; Chern et al., 2023).

Diffusion language models generate text by iteratively denoising a corrupted sequence, enabling parallel refinement-based decoding rather than strict left-to-right next-token prediction (Li et al., 2022; Sahoo et al., 2024; Lou et al., 2023; Zhang et al., 2025). Discrete diffusion objectives and reparameterised decoding schemes have made this practical for language modelling and infilling (Lou et al., 2023; Zheng et al., 2023). However, hallucination detection for D-LLMs remains underexplored: existing AR-LLM detectors do not transfer directly because D-LLMs generate via a *multi-step denois-*

ing process and may not expose the same token-logit signals as AR decoding (Chang et al., 2026).

Graph neural networks (GNNs) learn on relational structures by passing messages along edges (Gilmer et al., 2017; Battaglia et al., 2018; Kipf and Welling, 2016; Hamilton et al., 2017; Veličković et al., 2017; Xu et al., 2018). Temporal/dynamic GNNs extend this with time encodings and persistent node memories, modelling sequences of graph snapshots or continuous-time event streams (Pareja et al., 2020; Sankar et al., 2020; Trivedi et al., 2019; Kumar et al., 2019; Xu et al., 2020; Rossi et al., 2020). In transformer-based models, self-attention naturally induces a weighted directed token graph whose edge weights and node states evolve across iterative decoding steps (Vaswani et al., 2017), making temporal graph learning a natural abstraction for modelling these dependency structures.

3 Method

3.1 Problem Formulation

We consider a Diffusion Language Model (D-LLM) that generates a response sequence $\mathbf{y}_0 = (y^1, \dots, y^n)$ conditioning on an input prompt $\mathbf{x}$. Unlike sequential auto-regressive generation, D-LLMs generate text via a global, iterative refinement process (Li et al., 2022). The generation commences from a fully masked sequence $\mathbf{y}_T$ (where all positions are initialised to a special mask token) (Sahoo et al., 2024). Over T discrete timesteps, the model progressively denoises this sequence to produce a *decoding trajectory* of intermediate states $\mathcal{T} = \{\mathbf{y}_T, \mathbf{y}_{T-1}, \dots, \mathbf{y}_0\}$ (Nie et al., 2025). Each transition is governed by a learned conditional distribution $p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{x})$, where the model utilises bidirectional attention to refine the semantic structure of the entire sequence simultaneously during the iterative decoding process (Gong et al., 2024; Zhang et al., 2025).

Table 1: **Hallucination Detection Performance (AUROC) on LLaDA-8B and Dream-7B.** We compare TDGNet against output-, latent-, and graph-based baselines across four QA benchmarks. TDGNet consistently achieves state-of-the-art performance, improving average AUROC by 3 points on LLaDA-8B and 4 points on Dream-7B over the strongest baseline per model. These results confirm that modeling the denoising trajectory yields superior robustness compared to static output statistics or isolated snapshots. (SS: Single Sampling).

Model	Metric	SS	Math	CSQA	HotpotQA	TriviaQA	Ave
LLaDA-8B-Instruct	Output-Based						
	Semantic Entropy	✗	0.68	<u>0.64</u>	<u>0.61</u>	<u>0.66</u>	<u>0.65</u>
	Lexical Similarity	✗	0.51	0.53	0.54	0.54	0.53
	LN-Entropy	✗	0.70	0.59	0.55	0.55	0.60
	Perplexity	✗	0.67	0.60	0.51	0.54	0.58
	Latent-Based						
	EigenScore	✗	0.56	0.54	0.56	0.59	0.56
	TSV	✓	<u>0.72</u>	0.61	0.55	0.50	0.60
	CCS	✓	0.59	0.56	0.60	0.54	0.57
	TDGNet (Ours)	✓	0.72	0.65	0.64	0.72	0.68
Dream-7B-Instruct	Output-Based						
	Semantic Entropy	✗	0.59	0.50	0.68	0.69	0.62
	Lexical Similarity	✗	<u>0.71</u>	0.68	<u>0.71</u>	0.67	<u>0.69</u>
	LN-Entropy	✗	0.57	0.59	0.52	0.53	0.55
	Perplexity	✗	0.52	0.57	0.51	0.54	0.54
	Latent-Based						
	EigenScore	✗	0.68	0.55	0.63	<u>0.70</u>	0.64
	TSV	✓	0.68	<u>0.71</u>	0.43	0.50	0.58
	CCS	✓	0.59	0.56	0.64	0.60	0.60
	TDGNet (Ours)	✓	0.72	0.72	0.74	0.74	0.73

In this context, we frame hallucination detection as evaluating the factual validity of the generated content. Let $h \in \{0, 1\}$ be the ground-truth label where $h = 1$ indicates a hallucination. The goal is to learn a binary classifier f from a hypothesis space $\mathcal{H}$ that minimises the classification risk for the response $\mathbf{y}_0$, subject to the constraint that $\mathbf{y}_0$ is the result of the model’s reverse diffusion process:

$$\begin{aligned} \min_{f \in \mathcal{H}} \mathcal{L}(h, f(\mathbf{x}, \mathbf{y}_0)), \\ \text{s.t. } \mathbf{y}_0 \sim \prod_{t=T}^1 p_{\theta}(\mathbf{y}_{t-1} \mid \mathbf{y}_t, \mathbf{x}) \end{aligned} \quad (1)$$

where $\mathcal{L}(\cdot)$ is the cross-entropy loss. This formulation explicitly links the detection task to the generative dynamics, acknowledging that while the target $\mathbf{y}_0$ is a static sequence, its validity is an artefact of the specific *generative path* p_{θ} .

Conventional detection paradigms, however, leave two sources of signal unused. **Output-based approaches** face *likelihood intractability*: unlike AR models, D-LLMs lack a direct single-pass likelihood $p(\mathbf{y}|\mathbf{x})$, rendering perplexity-based metrics computationally expensive (Sahoo et al., 2024). **Latent-based strategies**, conversely, leave *temporal information* unused: by analyzing only the final

state $\mathbf{y}_0$ or a single snapshot, they discard validity signals such as semantic drift or persistent error that manifests as dynamic instabilities within the intermediate trajectory (Chang et al., 2026; Wang et al., 2026).

3.2 Capturing Bidirectional Dependencies and Evolution via Temporal Graph Modelling

To effectively detect hallucinations in D-LLMs, a representation must capture two fundamental generative properties: the *global context* facilitated by bidirectional attention and the *iterative refinement* of semantic content over time. Standard static graph approaches fail to capture the latter, while sequence-only methods miss the former. Therefore, we propose to model the computational trace not as a fixed snapshot, but as a Temporal Dynamic Graph, explicitly encoding the generation as an evolving web of token dependencies.

Why attention graphs. Several relational structures could be imposed on the token set: k -nearest-neighbour graphs over hidden states, syntactic parses, or learned adjacency. We use attention for three reasons. It is the only per-step, per-token-pair

signal the backbone exposes natively, so no metric, sparsity pattern, or auxiliary model must be chosen. It is already computed during the forward pass, so graph construction adds only a thresholding operation and the detector inherits the single-pass cost profile reported in Appendix A.2. Most importantly, bidirectional attention is the specific mechanism by which D-LLMs refine global context across denoising steps, so its evolution is the object whose dynamics we intend to track, whereas a similarity graph over hidden states would re-encode information already supplied to the network as node attributes. Section 4.3 reports the two ablations that test this choice empirically: removing the edges entirely, and replacing the trajectory with single-step static graphs.

Dynamic Graph Construction. Formally, we define the diffusion trace as a discrete-time sequence of attributed graphs $\mathcal{G} = \{G^{(0)}, \dots, G^{(T)}\}$. Each snapshot $G^{(t)} = (V, \mathcal{E}^{(t)}, \mathbf{X}_V^{(t)}, \mathbf{X}_E^{(t)})$ captures the structural state of the generation at denoising step t . First, the Vertex Set (V) consists of the N tokens in the sequence $\mathbf{y}$. Unlike AR generation, the node set remains fixed across the trajectory, serving as the spatial anchor for temporal evolution.

Second, the Temporal Edge Set $\mathcal{E}^{(t)}$ represents the dynamic connectivity. Let $\mathbf{A}^{(t,\ell,h)} \in \mathbb{R}^{N \times N}$ be the attention matrix of head $h \in \{1, \dots, H\}$ in backbone layer $\ell \in \{1, \dots, L_{\text{bb}}\}$ at denoising step t , where $N = |V|$ is the sequence length, H the number of attention heads, and L_{bb} the number of transformer layers in the D-LLM. Rows index queries and columns index keys, so $\mathbf{A}_{ij}^{(t,\ell,h)}$ is the weight token i places on token j and $\sum_j \mathbf{A}_{ij}^{(t,\ell,h)} = 1$. Unless stated otherwise we use the final layer $\ell = L_{\text{bb}}$ (layer sensitivity in Appendix A.1) and write $\bar{\mathbf{A}}_{ij}^{(t)} = \frac{1}{H} \sum_{h=1}^H \mathbf{A}_{ij}^{(t,L_{\text{bb}},h)}$ for the head-averaged score. A directed *message* edge is placed from j to i whenever i attends strongly to j :

$$\mathcal{E}^{(t)} = \left\{ (j, i) \in V \times V \mid \bar{\mathbf{A}}_{ij}^{(t)} > \tau \right\} \quad (2)$$

where $\tau \in (0, 1)$ is a sparsity threshold. The index order is deliberate: the threshold applies to $\bar{\mathbf{A}}_{ij}^{(t)}$, the attention from i to j , while the edge is written (j, i) because information flows from the attended token j into the attending token i . The in-neighbourhood of i is $\mathcal{N}^{(t)}(i) = \{j : (j, i) \in \mathcal{E}^{(t)}\}$, and this notation is used throughout. This

Table 2: **Ablation: static single-step graphs vs. the full dynamic trajectory (LLaDA-8B).** The highest score is **bolded** and the second highest is underlined.

Variant	SS	Math	CSQA	HotpotQA	TriviaQA	Ave
CHARM — Decoding step 0	✓	0.63	0.58	0.58	0.65	0.61
CHARM — Decoding step T/4	✓	0.65	0.60	0.61	0.68	0.63
CHARM — Decoding step T/2	✓	<u>0.70</u>	<u>0.63</u>	<u>0.63</u>	<u>0.69</u>	<u>0.66</u>
CHARM — Decoding step T	✓	0.65	0.63	0.62	0.68	0.64
TDGNet (Ours)	✓	0.72	0.65	0.64	0.72	0.68

construction allows the graph topology to evolve naturally, capturing phenomena such as the emergence of semantic cliques or the dissolution of random noise connections.

To unify heterogeneous signals, we enrich this structure with time-varying features. Node Attributes $\mathbf{X}_V^{(t)}$ are derived from residual stream activations, projected to $\mathbb{R}^{d_v}$ to capture instantaneous semantic confidence and context. Edge Attributes $\mathbf{X}_E^{(t)}$ are derived from the averaged attention scores across heads, preserving interaction strength while maintaining computational efficiency.

By formulating the diffusion trace as a dynamic graph sequence $\mathcal{G}$, our approach unifies the structural analysis of token interactions with the temporal analysis of their evolution. This representation allows the model to distinguish between *healthy refinement* (where dependencies stabilise) and *hallucinatory collapse* (where dependencies drift or lock onto irrelevant tokens or noisy patterns).

This reformulation is necessitated by the path-dependent nature of D-LLM hallucinations. Unlike static errors, hallucinations often emerge through semantic drift during the denoising process ($t = T \rightarrow 0$), a phenomenon invisible to single-step analysis (Chang et al., 2026). By defining the input as a dynamic sequence $\mathcal{G}$, we capture this structural evolution, allowing the model to distinguish between stable grounding and transient instabilities, thus motivating the temporal architecture detailed in the following section.

3.3 Hallucination Detection via Dynamic Trace Modelling

To classify the diffusion trajectory constructed in Section 3.2, we introduce the Temporal Dynamic Graph Network (TDGNet). This architecture processes the sequence of attention graphs $\{\mathcal{G}^{(t)}\}_{t=T}^0$ via a three-stage pipeline, which consists of spatial aggregation, temporal memory update, and trajectory readout, to distinguish between factual and hallucinated generation paths.

First, to capture the local structural context at

each diffusion step t , we employ a Message Passing Neural Network (MPNN) (Gilmer et al., 2017). This module aggregates information from the attention neighbourhood $\mathcal{N}^{(t)}(i)$. For each token i , we compute an aggregated message vector $\bar{\mathbf{m}}_i^{(t)}$ that summarises the information flow from its neighbours:

$$\bar{\mathbf{m}}_i^{(t)} = \frac{1}{|\mathcal{N}^{(t)}(i)|} \sum_{j \in \mathcal{N}^{(t)}(i)} \psi(\mathbf{h}_j^{(t)}, \mathbf{h}_i^{(t)}, \mathbf{e}_{ji}^{(t)}) \quad (3)$$

where $\mathbf{h}_i^{(t)}$ and $\mathbf{h}_j^{(t)}$ are the node features, $\mathbf{e}_{ji}^{(t)}$ are the edge features, and $\psi(\cdot)$ is a learnable message function implemented as a multi-layer perceptron (MLP). This spatial aggregation ensures that the model captures the instantaneous relational structure of the generation before processing its temporal evolution through recurrence.

Subsequently, to track how these structural roles evolve, we maintain a persistent memory state $\mathbf{s}_i^{(t)}$ for each token. We employ a Gated Recurrent Unit (GRU) to transition the memory state from step $t + 1$ to t :

$$\mathbf{s}_i^{(t)} = \text{GRU}(\bar{\mathbf{m}}_i^{(t)}, \mathbf{s}_i^{(t+1)}) \quad (4)$$

This recurrence allows the network to distinguish between consistent grounding and transient noise by integrating the sequence of spatial messages $\bar{\mathbf{m}}_i^{(t)}$ into a continuous "running belief" $\mathbf{s}_i^{(t)}$ representation over time.

Finally, to produce a robust prediction, we employ a Temporal Attention Mechanism that prioritises informative denoising phases over noisy initialisation. We compute a trajectory-aware embedding $\mathbf{z}_i$ for each token by attending over its own memory history:

$$\mathbf{z}_i = \sum_{t=0}^T \alpha_i^{(t)} \cdot \text{Linear}(\mathbf{s}_i^{(t)}), \quad (5)$$

$$\text{where } \alpha_i^{(t)} = \frac{\exp(\mathbf{q}^\top \mathbf{s}_i^{(t)})}{\sum_{k=0}^T \exp(\mathbf{q}^\top \mathbf{s}_i^{(k)})}$$

Here, $\text{Linear}(\cdot)$ denotes a linear projection layer, and $\mathbf{q}$ is a learnable query vector. This yields a final representation $\mathbf{z}_i$ that summarises the token's structural validity across the entire trajectory of the generation process.

Learning Objective. For the primary task of response-level hallucination detection, we apply global mean-pooling over the token embeddings to

Table 3: **Token-Level Hallucination Detection (AU-ROC)**. Comparison of our dynamic method (TDGNet) and static graph baseline (CHARM) against standard baselines on Natural Questions (NQ) and TriviaQA. Best results are **bolded**, second best are underlined.

Method	Natural Questions	TriviaQA
Graph Degree (Centrality)	0.43	0.46
Source Attribution	0.52	0.42
Token Probability	0.60	0.68
Predictive Entropy	0.69	0.71
CHARM (Static Graph)	<u>0.84</u>	<u>0.83</u>
TDGNet (Ours)	0.87	0.85

obtain a graph representation $\mathbf{g} = \frac{1}{N} \sum_i \mathbf{z}_i$. This is passed to a binary classifier f_ϕ to predict the probability $\hat{p}$. The model is trained end-to-end by minimising the binary cross-entropy loss against the ground truth labels $h \in \{0, 1\}$ indicating response validity:

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}, \mathbf{y}) \in \mathcal{D}} [h \log \hat{p} + (1 - h) \log(1 - \hat{p})] \quad (6)$$

where $\mathcal{D}$ represents the training dataset of prompt-response pairs and their corresponding ground-truth labels.

3.4 Implementation Details

We instantiate the TDGNet framework as a two-stage pipeline: a **Temporal Graph Encoder** g_θ that processes the diffusion trace into trajectory-aware node embeddings, and a **Hallucination Predictor** f_ϕ that supports both response-level (graph classification) and token-level (node classification) readout. Full implementation details appear in Appendix C.

4 Experiments

4.1 Setup

Datasets. We evaluate hallucination detection on five QA-style benchmarks with varying reasoning demands: Math (numerical reasoning, constructed following (Chuang et al., 2024)), CommonsenseQA (CSQA) (commonsense multiple choice) (Talmor et al., 2019), HotpotQA (multi-hop reasoning with supporting context) (Yang et al., 2018), TriviaQA (open-domain factoid QA) (Joshi et al., 2017), and Natural Questions (NQ) (factoid QA) (Kwiatkowski et al., 2019). While all datasets are utilised for response-level detection, we specifically employ NQ and TriviaQA for the fine-grained

token-level localisation experiments (Section 4.2), as their factoid nature supports precise span-level verification. Dataset construction, labelling, and splits follow Appendix D; all methods use identical train/test partitions.

Baseline Methods. We compare TDGNet against detectors spanning three paradigms (selection criteria in Appendix B). For response-level detection, we evaluate output-based methods (Semantic Entropy (Kuhn et al., 2023; Kossen et al., 2024), Length-Normalised (LN) Entropy and Perplexity (Lin et al., 2023), Lexical Similarity (Manakul et al., 2023)); latent-based methods (EigenScore (Chen et al., 2024), TSV (Park et al., 2025), CCS (Azaria and Mitchell, 2023; Orgad et al., 2025)); and the graph-based baseline CHARM (Frasca et al., 2026), reproduced in-house. For token-level localisation, response metrics suffer from a granularity mismatch (a single scalar dilutes local error signals), so we compare against native token-wise baselines testing distinct hypotheses: Predictive Entropy (Kuhn et al., 2023) (local uncertainty), Graph Degree Centrality (grounded tokens as attention hubs), and Source Attribution (Bahdanau et al., 2014; Vaswani et al., 2017) (factual tokens grounded in the prompt).

Models. We adopt LLaDA-8B-Instruct (Nie et al., 2025) and Dream-7B-Instruct (Ye et al., 2025) as our primary diffusion LMs, providing comparison across distinct architectures and noise schedules. To assess generalization further, we also evaluate on LLaDA 1.5 (Zhu et al., 2025), an 8B dense model aligned via VRPO, and LLaDA 2.1-mini (Bie et al., 2026), a 16B Mixture-of-Experts model with block-editable diffusion; results are discussed in Section 4.4. For graph methods, attention graphs are constructed from the final transformer layer using a standardised decoding configuration (gen_length=32, steps=16); Appendix A provides sensitivity analysis. Note that Dream-7B requires data curation to mitigate mode collapse, detailed in Appendix E.2.

Evaluation protocol. Following prior work (Kuhn et al., 2023; Manakul et al., 2023; Park et al., 2025; Chang et al., 2026), we report AUROC as the primary metric, selecting models on validation data. Table 1 summarises performance across all benchmarks. TDGNet achieves the best average AUROC on LLaDA-8B and strong results on Dream-7B, confirming the benefit of temporal modelling over output-, latent-, and static baselines.

Table 4: **Graph Utility Ablation.** The drastic drop without edges ($\mathcal{N}(i) = \emptyset$) confirms structural aggregation is essential.

Dataset	Performance (AUROC)		
	TDGNet	No Graph	Δ Drop
CSQA	0.65	0.40	-0.25
TriviaQA	0.72	0.43	-0.29
HotpotQA	0.64	0.48	-0.16
Math	0.72	0.49	-0.23

4.2 Main Results

Response-Level Detection. Table 1 compares TDGNet against output-based, latent-based, and graph-based baselines across two D-LLMs and four QA benchmarks. TDGNet is best or tied-best in every setting, improving average AUROC from 0.65 to 0.68 on LLaDA-8B-Instruct (over Semantic Entropy) and from 0.69 to 0.73 on Dream-7B-Instruct (over Lexical Similarity). These consistent gains demonstrate the value of exploiting the multi-step denoising trajectory rather than static output statistics or isolated latent snapshots.

Output-based baselines suffer from poor consistency: Semantic Entropy is competitive on LLaDA but collapses to 50.3% on Dream-7B-Instruct (CSQA), failing when hallucinations are not flagged by entropic uncertainty. Latent-based methods are more competitive overall but remain sensitive to task distribution: CCS achieves 64.3% on HotpotQA yet drops to near-random on TriviaQA, highlighting the limitation of seeking a universal “truth direction” in static embeddings.

Token-Level Localization. Beyond sequence-level detection, Table 3 validates the granular capabilities of our architecture. Standard uncertainty metrics like Predictive Entropy suffer from a granularity mismatch, achieving only $\sim 70\%$ AUROC, as broadcasting a single sequence-level score dilutes local error signals. In contrast, TDGNet leverages native node embeddings to outperform the static CHARM baseline by 2.1% on the TriviaQA benchmark. This confirms that the temporal memory mechanism does not merely aggregate signals but refines the structural boundaries of hallucinations, allowing for the precise localisation of errors that static topology alone fails to isolate.

4.3 Ablations

Threshold Sensitivity. The sparsity threshold τ in Eq. 2 is the only hyperparameter governing graph

Table 5: Cross-dataset zero-shot AUROC on LLaDA: Grouped by Method. Best results are **bolded**, second best are underlined.

Method	Source	Target Dataset			
		Math	CSQA	Hotpot	Trivia
CCS	Math	0.59	0.52	0.53	0.50
	CSQA	0.51	0.56	0.50	0.54
	Hotpot	0.54	0.55	0.60	0.59
	Trivia	0.52	0.51	0.51	0.54
TDGNet (Ours)	Math	0.72	0.49	<u>0.62</u>	0.55
	CSQA	0.55	0.65	0.59	0.58
	Hotpot	<u>0.61</u>	<u>0.59</u>	0.64	<u>0.62</u>
	Trivia	0.57	0.55	0.57	0.72

construction. Figure 3 shows that TDGNet is robust across a wide range: $\tau=0.03$ and $\tau=0.05$ differ by only 0.01 in average AUROC, with a mild trade-off, $\tau=0.03$ preserves more edges and benefits recall-oriented tasks (TriviaQA: 0.80 AUROC), while $\tau=0.05$ filters early-step noise more aggressively and benefits CSQA and HotpotQA. At $\tau=0.07$, excessive sparsification removes over 97% of edges and collapses performance to chance. We adopt $\tau=0.05$ as a universal default and report it in Table 1 for consistency across datasets.

Importance of Message-Passing Structure.

The spatial aggregation in Eq. (3) updates each token’s embedding from its attention-induced neighbourhood $\mathcal{N}(i)$, allowing TDGNet to capture relational patterns that isolated hidden states cannot. Ablating the edges ($\mathcal{N}(i) = \emptyset$) reduces the model to a per-token classifier and collapses performance to near-random across all datasets (Table 4), confirming that structural aggregation, not isolated node features, drives detection.

4.4 Analysis

Temporal Dynamics. To disentangle the contribution of temporal modelling from graph-based structural analysis, we introduce a set of static graph variants. We compare TDGNet against single-snapshot CHARM variants extracted at $t \in \{0, T/4, T/2, T\}$ (Table 2). Static variants peak only at mid-trajectory ($t = T/2$, 0.6627 AUROC) and consistently trail our dynamic approach (0.6814), with weaker performance at both early ($t = T$) and late ($t = 0$) snapshots. This non-monotonic behaviour confirms that validity cues are distributed throughout the trajectory rather than localised to any single denoising step, and TDGNet bridges this gap through trajectory-level aggregation that distinguishes stable grounding from tran-

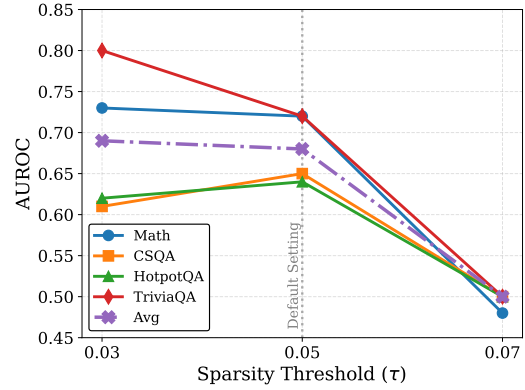


Figure 3: **Threshold sensitivity (LLaDA-8B).** TDGNet is robust for $\tau \in [0.03, 0.05]$ and collapses at $\tau=0.07$; we adopt $\tau=0.05$ as default.

sient artefacts.

Architectural Generalization. To test architecture-agnostic generalisation, we evaluate two further D-LLMs: LLaDA 1.5 (Zhu et al., 2025), an 8B dense model aligned via VRPO, and LLaDA 2.1-mini (Bie et al., 2026), a 16B Mixture-of-Experts model (1.4B active) with block-editable diffusion. Together with our primary models, this covers four diffusion paradigms across dense and MoE architectures (7B–16B). TDGNet outperforms all baselines on both (Figure 4: 0.76 on LLaDA 1.5, 0.74 on LLaDA 2.1-mini), despite the latter’s block-local attention. CHARM remains competitive on LLaDA 2.1-mini (0.71), indicating attention-graph structure carries signal even under altered diffusion mechanics, but trajectory-level aggregation still wins.

Cross-dataset Generalisation. Zero-shot transfer experiments (Table 5) show that graph-based detection generalises more robustly than latent-space methods: TDGNet maintains meaningful transfer (0.55–0.62 AUROC on all but one source–target pair) across diverse reasoning types, while CCS drops to near-random on distant pairs (0.50–0.54). HotpotQA in particular serves as a strong source (0.59–0.62 average transfer), suggesting that multi-hop reasoning patterns subsume simpler single-hop patterns and that core hallucination signatures are partially task-agnostic.

Efficiency. TDGNet operates on a single denoising trajectory, achieving a $\sim 4.8\times$ speedup over Semantic Entropy (442 s vs. 2101 s per 100 queries) while adding only ~ 160 ms of overhead per query relative to single-pass baselines such as Perplexity. Full per-method timings appear in Appendix A.2.

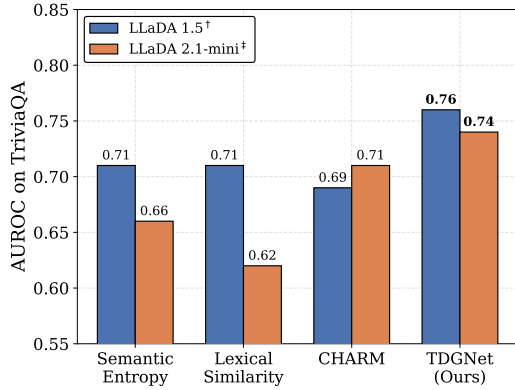


Figure 4: **Architectural generalisation (TriviaQA).** TDGNet outperforms all baselines on both dense (LLaDA 1.5, 8B) and Mixture-of-Experts (LLaDA 2.1-mini, 16B) D-LLMs.

5 Conclusion

TDGNet reframes hallucination detection for diffusion language models as learning over the *evolving* attention-graph trace, operationalising the principle that factuality is a *trajectory property* rather than solely an end-state outcome. By utilising token-level message passing and persistent memories to accumulate weak validity cues across denoising *stages*, our framework consistently improves AUROC over output-based, latent-based, and static graph baselines across four D-LLMs spanning dense and Mixture-of-Experts architectures, while simultaneously enabling fine-grained localisation of hallucinated spans. Analyses confirm that these gains stem from explicitly modelling diffusion-specific dynamics, such as self-correction, semantic drift, and persistent errors, allowing the detector to capture early warning signs and late-stage repair effects that static snapshots miss.

Limitations

Our empirical study is restricted to QA-style benchmarks because reliable hallucination detection requires unambiguous factuality labels, which opened generation tasks (e.g., summarisation, long-form generation) currently lack at scale. This restriction is further reinforced by the present state of open-source D-LLMs: contemporary models exhibit documented structural instabilities on longer-form generation (Appendix E.2), which makes it difficult to disentangle genuine hallucinations from fluency failures. We expect that as D-LLM generation quality matures and broader factuality benchmarks become available, extending TDGNet to

summarisation, dialogue, and long-form reasoning will be a natural and worthwhile direction.

Ethics Statement

TDGNet is intended to flag hallucinated outputs of diffusion language models in support of safer deployment, but no detector is perfect: false negatives may give users misplaced confidence in incorrect generations, and adversarial inputs could in principle be crafted to evade detection by manipulating the temporal attention dynamics the model relies on, so we recommend TDGNet be used as one component of a broader factuality pipeline rather than a sole safeguard in high-stakes settings. All datasets (Math, CommonsenseQA, HotpotQA, TriviaQA, Natural Questions) and pretrained models (LLaDA-8B, Dream-7B, LLaDA 1.5, LLaDA 2.1-mini) are publicly released research artefacts under permissive research-oriented licences, used in their intended research setting and consistently with those terms; licence details are summarised in Appendix D. We collect no new human data, and the benchmarks contain no personally identifying information of which we are aware. AI writing assistants were used for editorial purposes only, namely grammatical revision, sentence-level rephrasing, and LaTeX formatting; all scientific contributions, experimental design, implementations, analyses, and conclusions are the authors’ own.

Acknowledgments

The authors sincerely thank the SAC, AC, and all the reviewers for their valuable time and insightful feedback, which have significantly improved the quality of this work. This work is supported by the UKRI grant: Turing AI Fellowship EP/W002981/1.

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Amos Azaria and Tom Mitchell. 2023. The internal state of an llm knows when it’s lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 967–976.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

- Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, and 1 others. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.
- Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhao Feng, Haibo Feng, Mingliang Gong, and 1 others. 2026. Llada2. 1: Speeding up text diffusion via token editing. *arXiv preprint arXiv:2602.08676*.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.
- Shenxu Chang, Junchi Yu, Weixing Wang, Yongqiang Chen, Jialin Yu, Philip Torr, and Jindong Gu. 2026. Tracedet: Hallucination detection from the decoding trace of diffusion large language models. In *International Conference on Learning Representations*, volume 2026, pages 9472–9491.
- Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. 2024. Inside: LLMs’ internal states retain the power of hallucination detection. In *International Conference on Learning Representations*, volume 2024, pages 3056–3076.
- I Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, and 1 others. 2023. Factool: Factuality detection in generative ai—a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*.
- Yung-Sung Chuang, Linlu Qiu, Cheng-Yu Hsieh, Ranjay Krishna, Yoon Kim, and James Glass. 2024. Lookback lens: Detecting and mitigating contextual hallucinations in large language models using only attention maps. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1419–1436.
- Xuefeng Du, Chaowei Xiao, and Sharon Li. 2024. Halo-scope: Harnessing unlabeled LLM generations for hallucination detection. *Advances in Neural Information Processing Systems*, 37:102948–102972.
- Fabrizio Frasca, Guy Bar-Shalom, Yftah Ziser, and Haggai Maron. 2026. Neural message-passing on attention graphs for hallucination detection. In *International Conference on Learning Representations*, volume 2026, pages 449–475.
- Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. *arXiv preprint arXiv:1904.09324*.
- Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.
- Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, and 1 others. 2024. Scaling diffusion language models via adaptation from autoregressive models. *arXiv preprint arXiv:2410.17891*.
- Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.
- Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.
- Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.
- Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. 2024. Semantic entropy probes: Robust and cheap hallucination detection in LLMs. *arXiv preprint arXiv:2406.15927*.
- Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*.
- Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.
- Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-LLM improves controllable text generation. *Advances in neural information processing systems*, 35:4328–4343.
- Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Generating with confidence: Uncertainty quantification for black-box large language models. *arXiv preprint arXiv:2305.19187*.

- Qiang Liu, Xinlong Chen, Yue Ding, Bowen Song, Weiqiang Wang, Shu Wu, and Liang Wang. 2025. Attention-guided self-reflection for zero-shot hallucination detection in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21016–21032.
- Aaron Lou, Chenlin Meng, and Stefano Ermon. 2023. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*.
- Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 9004–9017.
- Sewon Min, Kalpesh Krishna, Xinxu Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100.
- Fatemehsadat Miresghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022. Mix and match: Learning-free controllable text generation using energy language models. *arXiv preprint arXiv:2203.13299*.
- Fan Nie, Xiaotian Hou, Shuhang Lin, James Zou, Huaxiu Yao, and Linjun Zhang. 2024. Facttest: Factuality testing in large language models with finite-sample and distribution-free guarantees. *arXiv preprint arXiv:2411.02603*.
- Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *arXiv preprint arXiv:2502.09992*.
- Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szepke, Hadas Kotek, and Yonatan Belinkov. 2025. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *International Conference on Learning Representations*, volume 2025, pages 66880–66913.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.
- Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. 2020. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5363–5370.
- Seongheon Park, Xuefeng Du, Min-Hsuan Yeh, Haobo Wang, and Yixuan Li. 2025. Steer LLM latents for hallucination detection. *arXiv preprint arXiv:2503.01917*.
- Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. 2020. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*.
- Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184.
- Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. 2020. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th international conference on web search and data mining*, pages 519–527.
- Weihang Su, Yichen Tang, Qingyao Ai, Changyue Wang, Zhijing Wu, and Yiqun Liu. 2024a. Mitigating entity-level hallucination in large language models. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 23–31.
- Weihang Su, Changyue Wang, Qingyao Ai, Yiran Hu, Zhijing Wu, Yujia Zhou, and Yiqun Liu. 2024b. Unsupervised real-time hallucination detection based on the internal states of large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14379–14391.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. 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 4149–4158.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
- Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. 2019. Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

- Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.
- Wen Wang, Bozhen Fang, Chenchen Jing, Yongliang Shen, Yangyi Shen, Qiuyu Wang, Hao Ouyang, Hao Chen, and Chunhua Shen. 2026. Time is a feature: Exploiting temporal dynamics in diffusion language models. In *International Conference on Learning Representations*, volume 2026, pages 72909–72935.
- Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*.
- Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.
- Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*.
- Lingzhe Zhang, Liancheng Fang, Chiming Duan, Minghua He, Leyi Pan, Pei Xiao, Shiyu Huang, Yunpeng Zhai, Xuming Hu, Philip S Yu, and 1 others. 2025. A survey on parallel text generation: From parallel decoding to diffusion language models. *arXiv preprint arXiv:2508.08712*.
- Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. 2023. A reparameterized discrete diffusion model for text generation. *arXiv preprint arXiv:2302.05737*.
- Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and 1 others. 2025. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*.

A Additional Empirical Analyses

A.1 Layer and Diffusion-Step Sensitivity

We systematically ablate the choice of transformer layer $\ell \in \{1, L_{\text{bb}}/4, L_{\text{bb}}/2, L_{\text{bb}}\}$ and diffusion timestep $t \in \{0, T/4, T/2, T\}$ in the graph construction, where L is the total number of layers and T is the total number of diffusion steps. We evaluate two configurations: (i) steps=16, gen_length=64 (Table 6); and (ii) steps=32, gen_length=128 (Table 7). Each table reports AUROC on test sets.

Findings. Performance improves consistently with deeper transformer layers ($\ell = L > L/2 > L/4 > 1$) across all probed denoising steps, indicating that later layers encode stronger hallucination cues. In contrast, the timestep effect is non-monotonic: mid-trajectory steps ($t = T/2$) provide the strongest single-step signal in both schedules, while the final step ($t = 0$) can be slightly weaker, consistent with late-stage repair effects. Overall, the best single-snapshot configuration is ($\ell = L, t = T/2$). Increasing the diffusion schedule to steps=32 with longer generation (gen_length 64 $\rightarrow$ 128) preserves the same trends and yields comparable performance, suggesting robustness to schedule length.

Table 6: Layer/step ablation (steps=16, gen_length=64). Rows index transformer layers $\ell \in \{1, L/4, L/2, L\}$ and columns index diffusion timesteps $t \in \{0, T/4, T/2, T\}$. Metric: AUROC.

Test	$t=0$	$t=T/4$	$t=T/2$	$t=T$
$\ell=1$	0.5043	0.5552	0.5785	0.5374
$\ell=L/4$	0.5549	0.6073	0.6246	0.5952
$\ell=L/2$	0.6005	0.6438	0.6652	0.6387
$\ell=L$	0.6344	0.6785	0.6941	0.6639

Table 7: Layer/step ablation (steps=32, gen_length=128). Rows index transformer layers $\ell \in \{1, L/4, L/2, L\}$ and columns index diffusion timesteps $t \in \{0, T/4, T/2, T\}$. Metric: AUROC.

Test	$t=0$	$t=T/4$	$t=T/2$	$t=T$
$\ell=1$	0.649	0.619	0.596	0.488
$\ell=L/4$	0.655	0.630	0.615	0.639
$\ell=L/2$	0.662	0.645	0.635	0.651
$\ell=L$	0.675	0.660	0.682	0.670

Table 8: **Efficiency Comparison.** TDGNet incurs marginal overhead over single-pass baselines, whereas sampling-based methods (Semantic Entropy, Lexical Similarity, EigenScore) are significantly slower due to multiple generations.

Method	Generations / Query	Overhead / Query	Time (100 Queries)
Perplexity	1	~66 ms	426 s
LN-Entropy	1	~66 ms	426 s
CCS	1	~96 ms	430 s
TSV	1	~66 ms	426 s
Semantic Entropy	5	~10 ms	2101 s
Lexical Similarity	5	~1 ms	2100 s
EigenScore	5	~14 ms	2102 s
TDGNet (Ours)	1	~160 ms	442 s

A.2 Efficiency Analysis

Diffusion language models already incur substantial per-query compute; an effective detector should not multiply that cost. We measure inference overhead and wall-clock time across 100 queries on LLaDA-8B (Table 8). Sampling-based methods (Semantic Entropy, Lexical Similarity, EigenScore) require $S = 5$ stochastic generations per query, inflating end-to-end latency by roughly $5\times$. TDGNet, in contrast, operates on a single denoising trajectory: its marginal overhead is ~ 160 ms per query — approximately 4.4% of the underlying 3.6 s generation time — yielding a $\sim 4.8\times$ speedup over Semantic Entropy (442 s vs. 2101 s for 100 queries) while matching or exceeding its detection accuracy on every benchmark in Table 1. This combines the single-pass speed of perplexity-style detectors with the robustness of trajectory-aware methods.

B Baseline Details

Baseline Selection Criteria. We evaluate against a comprehensive suite of reproducible state-of-the-art methods. We explicitly exclude methods requiring external knowledge retrieval (e.g., FactScore) to ensure a fair closed-book comparison. We additionally compare against the concurrent trajectory-based method TraceDet (Chang et al., 2026) via faithful reproduction (see Section B.3.2).

B.1 Token-Level Hallucination Detection Methods

Standard response-level metrics suffer from a *granularity mismatch* when applied to fine-grained tasks: broadcasting a single scalar score (e.g., sequence perplexity) across all tokens dilutes local error signals. To address this, we employ three baselines that generate **native token-wise** validity signals, each representing a distinct theoretical hypothesis about hallucination.

B.1.1 Predictive Entropy (Uncertainty Hypothesis)

Predictive Entropy measures the flatness of the model’s next-token probability distribution at each step t (Kuhn et al., 2023):

$$H(y_t | \mathbf{y}_{<t}, \mathbf{x}) = - \sum_{w \in \mathcal{V}} P(w | \mathbf{y}_{<t}, \mathbf{x}) \log P(w | \mathbf{y}_{<t}, \mathbf{x}) \quad (7)$$

This baseline tests the hypothesis that hallucinations correlate with model uncertainty. While effective for simple errors, it often fails when models are "confidently wrong."

B.1.2 Graph Degree Centrality (Structural Hypothesis)

This structural baseline computes the In-Degree of each token within the static attention graph constructed from the final layer.

$$\text{Degree}(y_t) = \sum_{j \neq t} \mathbb{I}[(j, t) \in \mathcal{E}] \quad (8)$$

This tests the "Hub Hypothesis": grounded, meaningful tokens typically act as attention sinks (hubs) for subsequent generation, while hallucinated tokens are often structural outliers or "babble" that the model ignores in future steps.

B.1.3 Source Attribution (Grounding Hypothesis)

Source Attribution quantifies the proportion of attention mass a generated token y_t directs towards the input prompt $\mathbf{x}$ versus the generated history $\mathbf{y}_{<t}$ (Bahdanau et al., 2014):

$$\text{Attr}(y_t) = \frac{\sum_{i \in \mathbf{x}} A_{t,i}}{\sum_{j \in (\mathbf{x} \cup \mathbf{y}_{<t})} A_{t,j}} \quad (9)$$

This tests the "Context Hypothesis": factual tokens should maintain strong attentional grounding in the source context, whereas hallucinations often detach from the prompt (low attribution).

B.2 Response-Level Hallucination Detection Methods

Response-level methods classify the validity of the entire generated sequence. We categorize these into **Output-Based** (using prediction statistics) and **Latent-Based** (using internal states).

B.2.1 Output-Based Methods

These approaches operate purely on generated text and probabilities, requiring no access to internal vector states.

Perplexity. Computes the negative log-likelihood of the sequence (Azaria and Mitchell, 2023). While theoretically sound, it often fails to distinguish confident hallucinations from factual statements, as both can have low perplexity.

Length-Normalized Entropy (LN-Entropy). Computes token-level entropy averaged over the sequence length to prevent short, high-entropy phrases from dominating the score (Kossen et al., 2024). It serves as a static heuristic for overall uncertainty.

Semantic Entropy. Partitions multiple stochastic generations into semantic clusters and computes entropy over these meanings rather than exact tokens (Kuhn et al., 2023). It effectively captures factual disagreement but requires expensive multiple forward passes (typically 5–10), making it slow for real-time applications.

Lexical Similarity. Measures the surface-level overlap (e.g., Rouge-L) across multiple generations (Lin et al., 2023). It is computationally cheap but prone to errors when the model paraphrases correctly (false positive) or repeats the same hallucination consistently (false negative).

B.2.2 Latent-Based Methods

These approaches exploit dense internal representations (hidden states), typically achieving higher performance than output-based heuristics.

EigenScore. Analyzes the eigenvalue spectrum of the covariance matrix formed by hidden states across multiple generations (Chen et al., 2024). Factual responses tend to cluster densely (low variance), while hallucinations scatter. However, it requires multiple generations and assumes consistency implies truth.

Contrast-Consistent Search (CCS). Unsupervisedly discovers a linear direction in activation space that satisfies logical consistency constraints between contrastive pairs (Burns et al., 2022). While powerful, it assumes hallucinations are linearly separable and is sensitive to distribution shifts.

Truthfulness Separator Vector (TSV). Learns a supervised hyperplane (e.g., via SVM) on labeled activation vectors to separate factual from hallucinated states (Park et al., 2025). It outperforms unsupervised methods like CCS but requires labeled training data.

B.3 Baseline Implementation Details and Exclusions

B.3.1 CHARM: Faithful Reproduction

CHARM (Frasca et al., 2026) proposes modeling computational traces as static attributed graphs. As the official code is unreleased, we implemented a **faithful reproduction** based on the paper’s architectural specifications using PyTorch Geometric. This reproduction was trained on the exact same diffusion datasets (Dream-7B/LLaDA) and splits used for TDGNet, ensuring a scientifically valid comparison between static graph processing and our temporal approach.

B.3.2 TraceDet: Reproduction and Comparison

TraceDet (Chang et al., 2026) is a concurrent trajectory-based detector that uses token-level entropy traces rather than attention graphs. Since the official code is unreleased, we reproduced the TraceDet w/o Masking variant following the hyperparameter specification in their paper (Section 3.4, Table 4), trained on identical splits and diffusion configurations as TDGNet. Results on LLaDA-8B are shown in Table 9. TDGNet outperforms TraceDet on reasoning-intensive benchmarks (HotpotQA: +0.05; TriviaQA: +0.09 at $\tau=0.03$) and additionally supports token-level localization that entropy-trace methods cannot.

B.3.3 Other Exclusions (ReDeEP, FactScore, FactTest)

We exclude **ReDeEP** (Su et al., 2024a) and **FactScore** (Min et al., 2023) as they require external knowledge retrieval (violating our closed-book setting). We exclude **FactTest** (Nie et al., 2024) and **AGSER** (Liu et al., 2025) due to impractical requirements for calibration sets or encoder-decoder specific optimizations unsuited for decoder-only diffusion models.

C Method Details

C.1 Implementation Details

We instantiate the TDGNet framework as a two-stage pipeline comprising a **Temporal Graph Encoder** g_θ and a **Hallucination Predictor** f_ϕ .

(a) Temporal Graph Encoder Configuration.

- *Graph Sparsity Threshold* (τ): To robustly define the edge set $\mathcal{E}^{(t)}$, we employ a threshold τ rather than a Top-K constraint. This allows the graph topology to evolve naturally: distinct

semantic cliques can emerge at later denoising stages without being artificially constrained, while the dense, uniform noise characteristic of early diffusion steps is effectively filtered out.

- *Efficient Trajectory Sampling*: While TDGNet is formulated over the full denoising trace $\{G^{(t)}\}_{t=0}^T$ (processed in denoising order $t = T \rightarrow 0$), processing every step incurs high redundancy and computational cost. To balance fidelity with efficiency, we sample a strategic set of keyframes $\mathcal{T}_{\text{key}} = \{T, \lfloor T/2 \rfloor, \lfloor T/4 \rfloor, 0\}$. This sequence captures the critical phases of generation: initial structural formation ($t = T$), semantic construction ($t = \lfloor T/2 \rfloor$), and late-stage refinement ($t = 0$). Crucially, this does not reduce the model to an ensemble of static snapshots; the recurrent memory states persist and update sequentially across these keyframes, allowing TDGNet to model the *evolutionary dynamics* of the hallucination even with sparse temporal sampling.

- *Feature Initialization*: Node features $\mathbf{h}_i^{(t)}$ are initialized from projected final-layer hidden states ($\mathbb{R}^{4096} \rightarrow \mathbb{R}^{128}$). The memory module uses a GRU with state dimension $d_m = 128$, and the temporal attention query $\mathbf{q}$ is similarly set to $d_m = 128$.

(b) Dual-Task Readout Heads. TDGNet supports both global and granular detection via specialized heads:

1. **Response-Level (Graph Classification)**: For sequence-level detection, we apply global mean-pooling over the final temporal node embeddings $\mathbf{z}_i$ to obtain $\mathbf{g}$, which is passed to a binary classifier $f_{\text{seq}}(\mathbf{g})$.
2. **Token-Level (Node Classification)**: For fine-grained localization, we bypass pooling and feed the learned individual node embeddings $\mathbf{z}_i$ into a shared token classifier $f_{\text{token}}(\mathbf{z}_i)$. This allows the model to pinpoint specific structural anomalies (e.g., local attention drift) without losing any granular resolution.

C.2 Hyperparameter Configuration

Our graph-based hallucination detector involves multiple architectural and optimization hyperparameters that jointly control model expressiveness,

Table 9: Comparison with TraceDet (AUROC on LLaDA-8B).

Method	Math	CSQA	HotpotQA	TriviaQA	Avg
TraceDet w/o Masking [†]	0.74	0.67	0.59	0.71	0.68
TDGNet ($\tau=0.05$)	0.72	0.65	0.64	0.72	0.68
TDGNet ($\tau=0.03$)	0.73	0.61	0.62	0.80	0.69

[†]Reproduced based on Section 3.4 and Table 4 of [Chang et al. \(2026\)](#); official code unreleased at time of writing.

regularization, and computational efficiency. We conducted extensive grid search across 1,296 configurations (5 learning rates $\times$ 3 batch sizes $\times$ 6 dropout rates $\times$ 4 layer depths $\times$ 3 hidden dimensions $\times$ 3 attention heads) to identify robust settings that generalize across datasets and hallucination types.

C.2.1 Parameter Justification

Learning Rate and Batch Size. Learning rates span five orders of magnitude ($1e-5$ to $1e-3$), accommodating both conservative updates on small datasets and aggressive optimization on larger corpora. Batch sizes of 16, 32, and 64 balance gradient estimation quality (smaller batches provide noisier but potentially more regularizing gradients) with computational throughput and memory constraints.

Dropout and Model Depth. Dropout rates from 0.0 to 0.5 provide a spectrum of regularization intensities, crucial for preventing overfitting on smaller datasets like the 2,000-sample Dream-7B collection. Detector depth ($L_{\text{gnn}} \in \{2, 3, 4, 5\}$) controls the receptive field of message-passing aggregation: deeper networks capture multi-hop token dependencies but risk over-smoothing, where node representations converge to similar values, degrading discriminative power.

Hidden Dimension and Attention Heads. Hidden dimensions $d \in \{64, 128, 256\}$ determine the capacity of intermediate representations. Attention heads $\in \{2, 4, 8\}$ enable the model to attend to multiple distinct patterns of token interactions simultaneously; more heads increase expressiveness but also computational cost.

C.3 Training Framework

Algorithm 1 formalises the end-to-end training procedure for TDGNet. The framework consumes a precomputed attention-graph trajectory $\mathcal{G} = \{G^{(t)}\}_{t=T}^0$ together with a binary hallucination label $y \in \{0, 1\}$, and produces a single scalar prediction $\hat{y}$. Processing proceeds in denoising order

($t = T \rightarrow 0$): at each step, the head-averaged attention map is sparsified into message edges (via threshold τ), messages are propagated through an MLP-based message function, and per-token GRU memories are updated with the resulting aggregate. After all timesteps are processed, a temporal-attention readout aggregates each token’s memory history into a trajectory-aware embedding, mean-pooling yields a graph-level representation $\mathbf{g}$, and a final MLP predicts the hallucination label. The entire pipeline is differentiable and trained end-to-end via binary cross-entropy. We elaborate on the three principal phases below.

C.3.1 Algorithm Overview

Algorithm 1 details the training procedure for TDGNet. To efficiently capture hallucination signals that evolve across the denoising trajectory, the framework operates in three principal stages:

1. Sparse Memory Evolution (Lines 7–14). For each denoising timestep t (processed in reverse order from $t = T$ to 0), we first construct a sparse attention graph by retaining only edges whose (averaged) attention weight exceeds a threshold τ . Unlike static GNNs that process snapshots in isolation, TDGNet maintains a persistent memory state $\mathbf{s}_i$ for each token. We apply a Gated Recurrent Unit (GRU) to update this memory sequentially:

$$\mathbf{s}_i^{(t)} = \text{GRU} \left(\text{AGG}(\mathcal{N}^{(t)}(i)), \mathbf{s}_i^{(t+1)} \right) \quad (10)$$

This recurrent update allows the model to track the *trajectory* of token interactions (e.g., detecting if a token “drifts” from correct to incorrect associations) while avoiding the computational cost of processing dense attention matrices.

2. Temporal Attention Readout (Lines 16–19). Hallucinations in diffusion models can be transient (appearing only in intermediate steps) or persistent. To capture both, we employ a temporal attention mechanism. Rather than averaging all timesteps

Table 10: Hyperparameter search space for TDGNet. Exhaustive grid search over 1,296 configurations enables systematic identification of architecture-dataset pairs that maximize hallucination detection performance while maintaining computational efficiency.

Hyperparameter	Search Space
Learning Rate (lr)	$\{1 \times 10^{-5}, 5 \times 10^{-5}, 1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}\}$
Batch Size	$\{16, 32, 64\}$
Dropout Rate	$\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$
Detector Depth (L_{gmn})	$\{2, 3, 4, 5\}$
Hidden Dimension (d)	$\{64, 128, 256\}$
Number of Attention Heads	$\{2, 4, 8\}$

equally, the model learns a query vector $\mathbf{q}$ to compute attention weights $\alpha_i^{(t)}$, dynamically prioritizing the denoising steps where structural anomalies are most pronounced.

3. Classification (Lines 21–23). The temporally-weighted token embeddings are pooled to form a global graph representation $\mathbf{g}$, which is passed to the MLP classifier. The entire pipeline is differentiable and trained end-to-end using binary cross-entropy loss.

C.3.2 Computational Complexity

For a dynamic graph sequence with T timesteps and N tokens, the computational cost consists of graph construction and message passing. The construction step involves thresholding the attention matrix, costing $\mathcal{O}(T \cdot N^2)$. However, since the D-LLM backbone computes this matrix inherently during inference, our detector merely performs a lightweight element-wise check. The subsequent GNN processing (dominated by the GRU) operates on the sparse graph:

$$\mathcal{O}(T \cdot N \cdot k \cdot d + T \cdot N \cdot d^2) \quad (11)$$

where k is the average degree after thresholding ($k \ll N$). Since $N \approx 512$ and $k \approx 10$, the sparse update is extremely fast ($\sim 0.5\text{ms/step}$; GNN update only, while total overhead is reported in Table 8). The total inference time remains orders of magnitude lower than multi-sampling baselines ($S \times T \times N^2$).

C.4 Feature Initialization and Input Specifications

To address potential ambiguity regarding input representations, we explicitly define the feature initialization and recurrence inputs used in TDGNet:

Node Feature Initialization. For a token i at diffusion timestep t , the initial node feature $\mathbf{h}_i^{(t)}$ is derived from the model’s internal activations. Specifically, we extract the hidden state vector from the **final transformer layer** immediately after the feed-forward block. For LLaDA-8B ($d_{\text{model}} = 4096$) and Dream-7B ($d_{\text{model}} = 4096$), we project this high-dimensional vector to a lower-dimensional state using a learnable linear layer W_{node} :

$$\mathbf{h}_i^{(t)} = W_{\text{node}} \cdot \text{HiddenState}_i^{(t)} \in \mathbb{R}^{128} \quad (12)$$

This ensures the graph encoder receives semantic context (what the token *means*) alongside structural context.

Edge Features. We construct directed message edges $(j, i) \in \mathcal{E}^{(t)}$ whenever token i attends strongly to token j (Eq. 2). Each such edge is weighted by the head-averaged attention score $\bar{A}_{ij}^{(t)} = \frac{1}{H} \sum_{h=1}^H A_{ij}^{(t,L,h)}$ from the final attention layer. We treat this scalar as an edge attribute $\mathbf{e}_{j,i}^{(t)} = [\bar{A}_{ij}^{(t)}]$, allowing the message passing network to modulate signal strength based on attention intensity.

GRU Input Specification. The Temporal Graph Network’s memory update (Eq. 10) receives a concatenated input vector. For a token i at step t , the input to the GRU is:

$$\text{Input}_{\text{GRU}}^{(t)} = [\mathbf{s}_i^{(t+1)} \parallel \bar{\mathbf{m}}_i^{(t)}] \quad (13)$$

where $\mathbf{s}_i^{(t+1)}$ is the previous memory state (from the preceding denoising step) and $\bar{\mathbf{m}}_i^{(t)}$ is the aggregated message computed from the current sparse graph neighbors. This design forces the recurrent unit to integrate the *historical context* (memory) with the *instantaneous structural update* (message).

Algorithm 1 Training Framework for TDGNet

```
1: Input: Attention-graph trajectory  $\mathcal{G} = \{G^{(t)}\}_{t=T}^0$ , where  $G^{(t)} = (V, \mathcal{E}^{(t)}, \mathbf{X}_V^{(t)})$ , per-head attention  $\{\mathbf{A}^{(t,L,h)}\}_{h=1}^H$ , label  $y \in \{0, 1\}$ 
2: Hyperparameters: memory dim  $d_m$ , sparsity threshold  $\tau$ , query vector  $\mathbf{q}$ 
3: Initialize:
4:   Node memories  $\mathbf{S} \in \mathbb{R}^{|V| \times d_m} \leftarrow \mathbf{0}$   $\triangleright$  stores  $\mathbf{s}_i^{(T+1)} = \mathbf{0}$ 
5:   Learnable params  $\theta_{\text{msg}}, \theta_{\text{gru}}, \theta_{\text{attn}}, \theta_{\text{pred}}$ 
6: // Phase 1: Sequential Temporal Encoding (Denoising Order)
7: for  $t = T$  down to 0 do
8:   Feature Extraction:  $\mathbf{h}_i^{(t)} \leftarrow \text{Project}(\mathbf{X}_{V,i}^{(t)})$ 
9:   Head-Averaged Attention:  $\bar{\mathbf{A}}_{ij}^{(t)} \leftarrow \frac{1}{H} \sum_{h=1}^H \mathbf{A}_{ij}^{(t,L,h)}$ 
10:  Sparsify (message edges):
11:     $\mathcal{E}^{(t)} \leftarrow \{(j, i) \mid \bar{\mathbf{A}}_{ij}^{(t)} > \tau\}$   $\triangleright (j, i)$  denotes message  $j \rightarrow i$  if  $i$  attends to  $j$ 
12:    Edge Attributes: for each  $(j, i) \in \mathcal{E}^{(t)}$ , set  $\mathbf{e}_{j,i}^{(t)} \leftarrow \lfloor \bar{\mathbf{A}}_{ij}^{(t)} \rfloor$ 
13:    Message Passing (incoming to  $i$ ):
14:       $\text{msg}_i^{(t)} = \text{MEAN}\left(\{\text{MLP}_{\text{msg}}(\mathbf{h}_j^{(t)} \oplus \mathbf{h}_i^{(t)} \oplus \mathbf{e}_{j,i}^{(t)}) : j \in \mathcal{N}^{(t)}(i)\}\right)$ 
15:    Memory Update (GRU):
16:       $\mathbf{s}_i^{(t)} = \text{GRU}(\text{msg}_i^{(t)}, \mathbf{s}_i^{(t+1)})$ 
17:    Latent Projection:  $\mathbf{z}_i^{(t)} = \text{Linear}(\mathbf{s}_i^{(t)})$ 
18:  end for
19: // Phase 2: Temporal Attention Readout
20: for each node  $i \in V$  do
21:   // Attend over memory states  $\mathbf{s}$ , aggregate projected latents  $\mathbf{z}$ 
22:    $\alpha_i^{(u)} = \frac{\exp(\mathbf{q}^\top \mathbf{s}_i^{(u)})}{\sum_{k=0}^T \exp(\mathbf{q}^\top \mathbf{s}_i^{(k)})}$   $\triangleright$  for all  $u \in \{0, \dots, T\}$ 
23:    $\mathbf{z}_i^{\text{Attn}} = \sum_{u=0}^T \alpha_i^{(u)} \mathbf{z}_i^{(u)}$ 
24: end for
25: // Phase 3: Classification
26:  $\mathbf{g} = \text{MeanPool}(\{\mathbf{z}_i^{\text{Attn}}\}_{i=1}^{|V|})$ 
27:  $\hat{y} = \sigma(\text{MLP}_{\text{pred}}(\mathbf{g}))$ 
28: Optimization: Update  $\theta$  via backpropagation
```

D Dataset Details

D.1 Math, CSQA, HotpotQA, TriviaQA, Natural Questions

Dataset construction. We evaluate hallucination detection across five complementary QA benchmarks that span diverse reasoning skills, input modalities, and output formats: **Math** (Chuang et al., 2024), **CommonsenseQA (CSQA)** (Talmor et al., 2019), **HotpotQA** (Yang et al., 2018), **TriviaQA** (Joshi et al., 2017), and **Natural Questions (NQ)** (Kwiatkowski et al., 2019). Taken together, these datasets cover numerical problem solving, discrete commonsense reasoning, multi-hop contextual QA, and open-domain factoid QA. While all datasets are utilized for response-level detection, we specifically employ **Natural Questions** and

TriviaQA for the fine-grained **token-level localization** experiments, as their factoid nature supports precise span-level verification.

Math. The Math dataset is constructed following the protocol of Chuang et al. (2024), and consists of mathematically oriented word problems that require multi-step symbolic or numerical reasoning to produce a single final numeric answer. Each instance typically involves several intermediate computation or derivation steps (e.g., algebraic manipulation, arithmetic, or geometric reasoning), but the model is ultimately evaluated on the correctness of the final scalar output. This benchmark stresses *precision in reasoning*: even small arithmetic or algebraic deviations render the answer incorrect, making Math a natural setting for studying hallucinations that arise from failures in structured

reasoning rather than from missing factual knowledge.

CommonsenseQA (CSQA). CommonsenseQA is a multiple-choice QA benchmark designed to probe everyday commonsense knowledge (Talmor et al., 2019). Each question is accompanied by several plausible answer candidates, of which exactly one is annotated as correct. Many distractor options are semantically close to the true answer, so success requires distinguishing between subtly different interpretations of the question and leveraging background commonsense rather than surface-level lexical cues. In our context, CSQA evaluates hallucinations in a *discrete decision space*: a hallucinated response corresponds to confidently selecting an incorrect-but-plausible option, which differs qualitatively from hallucinations in open-ended generation.

HotpotQA. HotpotQA is a multi-hop, context-based QA dataset where answering correctly typically requires aggregating evidence across multiple supporting documents and performing explicit multi-step reasoning (Yang et al., 2018). Questions are constructed so that the answer is not directly extractable from a single sentence; instead, the model must combine information spread across distinct passages (e.g., linking entities, comparing attributes, or following coreference chains). This benchmark is therefore well-suited for studying hallucinations that emerge from erroneous evidence integration, spurious reasoning chains, or the blending of partially correct and incorrect facts within long-context reasoning. It tests whether hallucination detectors can recognize when the model constructs a coherent but factually unsupported multi-hop explanation.

TriviaQA. TriviaQA is an open-domain factoid QA benchmark comprising natural language questions paired with evidence documents and short textual gold answers, often corresponding to named entities, dates, or specific phrases (Joshi et al., 2017). Compared to HotpotQA, TriviaQA places greater emphasis on factual recall and entity-level knowledge, and less on explicitly structured multi-hop reasoning. In practice, TriviaQA is representative of real-world assistant-style queries, where users ask for specific facts and models respond in free-form text. Hallucinations on this dataset commonly take the form of confidently asserted but incorrect entities or attributes (e.g., wrong person, place, or year), making it an important testbed for evaluating hallucination detection in open-ended factual

generation.

Natural Questions (NQ). Natural Questions is a benchmark for factoid question answering derived from real user queries issued to the Google search engine (Kwiatkowski et al., 2019). Similar to TriviaQA, it focuses on retrieving precise factual answers (e.g., entity names, dates, locations) from open-domain contexts. We include NQ specifically to facilitate our **token-level** evaluation: since the answers are typically short, grounded factoid spans, they allow for unambiguous localization of hallucinated tokens compared to long-form reasoning tasks.

Hallucination labeling protocol. To adapt these benchmarks for hallucination detection, we derive binary hallucination labels via automatic comparison between model outputs and ground-truth answers. For Math, a generated response is marked as non-hallucinated if and only if the decoded final numeric answer exactly matches the gold solution (Chuang et al., 2024), providing unambiguous supervision in the numerical reasoning regime. For CSQA, we treat the task as standard multiple-choice classification and label an output as non-hallucinated when the predicted option exactly matches the annotated correct choice (Talmor et al., 2019). For HotpotQA, TriviaQA, and Natural Questions, where models produce free-form text, we follow common QA evaluation practice and deem an answer correct if it satisfies an *exact match* criterion or achieves sufficient lexical overlap with the reference answer, using token-level overlap heuristics similar to those employed in prior work (Yang et al., 2018; Joshi et al., 2017). This unified protocol yields scalable and consistent hallucination labels across numerical, multiple-choice, contextual multi-hop, and open-domain factoid QA, enabling a comprehensive evaluation of hallucination detection methods under heterogeneous task conditions.

Splitting. We fix the random seed to 42 and split at the prompt-response level to ensure no leakage. For all **five** datasets, we employ a uniform split of **1500 training, 300 validation, 300 test examples**, ensuring comparable evaluation conditions across varying task complexities.

Generative traces and graph construction. We evaluate three decoding configurations: (i) `gen_length=32, steps=16`; (ii) `gen_length=64, steps=16`; and (iii) `gen_length=128, steps=32`. We found that the `gen_length=32, steps=16`

configuration yields the best performance and is adopted as our primary setting. Unless otherwise noted, we process all splits separately for each model (LLaDA-8B and Dream-7B) under this primary configuration (gen_length=32, steps=16) to extract denoising traces and construct attention graphs.

E Diffusion Language Models: Architecture and Stability Analysis

E.1 Model Architectures

E.1.1 LLaDA-8B-Instruct

LLaDA-8B-Instruct is a diffusion-based language model generating text through iterative denoising steps. The model maintains rich intermediate representations at each denoising step, including attention maps and activation vectors from all transformer layers. This property enables direct analysis of computational traces without relying on final-layer probabilities. LLaDA-8B-Instruct exhibits stable generation behavior across diverse prompts and temperatures, making it a reliable primary testbed for hallucination detection.

E.1.2 Dream-7B-Instruct

Dream-7B-Instruct represents the other major open-source diffusion LM architecture currently available. Unlike LLaDA, Dream-7B utilizes a distinct noise schedule and masking strategy. **Given the limited availability of high-performing open-source D-LLMs, including Dream-7B is essential to demonstrate that our method generalizes across different diffusion architectures.** However, as it exhibits distinct generation instabilities, we apply a specific curation protocol (detailed below) to ensure valid benchmarking.

E.2 Dream-7B-Instruct: Generation Dynamics and Data Curation

E.2.1 Structural Degradation and Mode Collapse

Dream-7B-Instruct exhibits a failure mode during zero-shot generation characterized as *Structural Degradation*, where outputs collapse into repetitive, semantically void sequences (e.g., repeating punctuation). This necessitates a distinction between *generation quality* (fluency) and *hallucination* (factuality).

Mechanism. We hypothesize this failure arises from three interconnected stages:

1. **Semantic Drift:** During the iterative denoising trajectory (steps 0–512), the model’s internal representations occasionally lose semantic fidelity to the prompt, drifting toward generic unconditional probability distributions.
2. **Attractor State Formation:** In the absence of strong Classifier-Free Guidance (CFG), the decoding trajectory gravitates toward high-probability generic tokens (e.g., separators) that dominate the training corpus.
3. **Attention Entrenchment:** Once a generic token is generated, self-attention mechanisms may disproportionately attend to it, locking the model into a self-reinforcing loop (e.g., “!!!!!!!!!!!!!!!!”).

Empirical Evidence. Systematic ablation studies (Table 11) confirm that this degradation is intrinsic to the model’s current checkpoint rather than a hyperparameter artifact.

E.2.2 Data Curation Strategy: Disentangling Fluency from Factuality

To rigorously evaluate *hallucination detection* rather than *quality estimation*, we employed a temperature-scaled curation strategy. This ensures our metrics reflect the detector’s ability to identify plausible falsehoods rather than trivially flagging broken syntax.

Protocol. We exploit the model’s behavior under varying noise levels:

- **Low Temperature (0.0–0.1):** Promotes structural rigidity to harvest factually correct responses.
- **High Temperature (0.7–1.0):** Disrupts attention entrenchment to collect diverse, hallucinated-yet-coherent samples.

We generated extensive traces and applied strict quality filtering to create a fixed evaluation set of ~2,000 samples (50% factual, 50% hallucinated). **Crucially, this curated dataset was held fixed across all baselines**, ensuring performance differences are attributable to method efficacy rather than sensitivity to model collapse.

E.2.3 Scientific Value: Latent Stability

Despite the instability of the decoding process, our analysis reveals a critical finding: **Dream-7B’s internal latent graphs remain highly in-**

Table 11: Parameter sensitivity analysis for Dream-7B-Instruct. Structural degradation persists across standard hyperparameter ranges, necessitating the rigorous filtering strategy we employed.

Parameter	Values Tested	Outcome	Observation
Denoising Steps	16 to 512	Degraded	Low steps yield incompleteness; high steps trigger collapse.
Prompt Template	Alpaca, ChatML, Raw	Degraded	Diffusion loses prompt adherence mid-generation.
Temperature	0.0, 0.5, 1.0	Mixed	High temperature mitigates collapse but reduces precision.

formative even when surface generation degrades. TDGNet achieves state-of-the-art performance (0.73 AUROC) on these traces, significantly outperforming output-based baselines like Perplexity (~ 0.55 AUROC). This suggests that diffusion LMs may be more reliable as discriminative scorers (via internal states) than as generative engines in their current state—a finding that informs future work on D-LLM utilization.

F Qualitative Case Studies: Hallucination Dynamics

Our method captures three observed hallucination patterns that reveal the mechanisms underlying model failures. These are illustrative instances rather than an exhaustive taxonomy, and a single trajectory may exhibit more than one; Correctness Decay in Figure 1 is a variant of drift in which the degraded intermediate state was factually correct. These case studies demonstrate how the dynamic graph representation exposes internal contradictions even when final outputs appear correct or when hallucinations persist uncorrected. For clarity, in these case studies (and figures) we index denoising iterations by generation steps $s \in \{0, \dots, T\}$ (increasing with decoding progress), which corresponds to diffusion time $t = T - s$ in our main notation (where $t = T$ is pure noise and $t = 0$ is the final output).

F.1 Case Study 1: Interleaving Hallucination (Transient Noise)

F.1.1 Dataset and Observation

Domain: Math (Arithmetic Reasoning). The model begins with correct intermediate reasoning, degenerates into nonsensical token repetition during intermediate denoising steps, yet surprisingly recovers to produce the correct final answer. This “Self-Correction” represents a case of **transient in-**

stability. Crucially, because TDGNet aggregates the multi-step trajectory, it successfully identifies the final resolution and correctly predicts 0 for the hallucination score despite intermediate noise. This highlights the model’s robustness: unlike static snapshot methods that might flag the intermediate degeneration as a fatal error, TDGNet recognizes the successful recovery.

F.1.2 Example and Denoising Trace

F.1.3 Interpretation

The graph-based detector observes the transient incoherence at Steps 4–10, where token-level attention becomes unstable. However, unlike static methods that might erroneously classify this intermediate noise as a fatal error, TDGNet’s memory update tracks the *recovery* in Steps 12–16. By integrating the full history, the model determines that the structural integrity was restored, resulting in a correct non-hallucinated prediction (Score ≈ 0). This confirms that temporal modeling effectively filters out process instabilities that do not impact the factual validity of the final response.

F.2 Case Study 2: Inconsistent Guesses (Drifting)

F.2.1 Dataset and Observation

Domain: TriviaQA (Knowledge Retrieval). The model attempts to retrieve a factual entity but drifts over the denoising trajectory, progressively hallucinating increasingly specific but incorrect details. This captures the “Searching $\rightarrow$ Drifting” dynamic where the model begins near plausible states but diverges into false specificity.

F.2.2 Example and Denoising Trace

The complete denoising trace is shown below.

Case Study 1: Interleaving Hallucination

Question: Estevan has 24 blankets. One-third of the blankets have polka-dots. For his birthday, his mother gives him 2 more polka-dot print blankets. How many polka-dot blankets does Estevan have in total?

Golden Label: 10

Denoising Trace:

- **Question:** Estevan has 24 blankets. One-third of the blankets have polka-dots. For his birthday, his mother gives him 2 more polka-dot print blankets. How many polka-dot blankets does Estevan have in total?
- **Golden Label:** 10
- **Denoising Trace:**
 - Step 4: One-third of 24 blankets is $24/3 = 8$ pol pol-d-d-d-d After mother... answer is 10
 - Step 8: One-third of 24 blankets is 82433an88 pol pol-d-d-d-d... answer: 10 (*Severe token degeneration*)
 - Step 10: One-third of 24 blankets is $24/3 = 8$ polka-dEstot. AfterAfter more... Therefore: 10
 - Step 12: One-third of 24 blankets is $24/3 = 8$ polka-dot blankets. After receiving 2 more... Estevan now has $8 + 2 = 10$ polka-dot blankets. Therefore: 10

Final Output: 10 (Correct answer; transient process instability resolved.)

TDGNet Detection Score: 0 (Correctly flagged as non-hallucinated response)

F.2.3 Interpretation

The drifting pattern reveals a critical hallucination mechanism: the model enters a “search cone” for plausible entities, finds a structurally similar surname (Kendall), and then progressively elaborates on it with increasing confidence. The attention graph evolves as follows: early steps show diffuse attention across multiple tokens (exploring possibilities); middle steps show attention concentrating on “Kendall”; late steps show self-reinforcing attention patterns where the model attends increasingly to its own predictions of first names paired with Kendall. Our graph aggregation detects this as hallucination because the attention pattern exhibits *semantic drift, the model’s confident focus on an incorrect entity, rather than stable grounding in the prompt. This pattern is distinct from transient noise (Case Study 1) because the model maintains high confidence throughout, making output-based methods (which rely on entropy) ineffective.*

F.3 Case Study 3: Persistent Hallucination (Stuck)

F.3.1 Dataset and Observation

Domain: HotpotQA (Multi-hop Reasoning). The model establishes an incorrect premise at the very beginning of generation and adheres to it rigidly throughout the entire denoising process, showing no drift or self-correction. This “Stuck” pattern represents the most problematic hallucination type: confident, persistent, and unreachable.

F.3.2 Example and Denoising Trace

The complete denoising trace is shown below.

F.3.3 Interpretation

The persistent pattern represents the worst-case scenario: the model locks onto an incorrect premise immediately and never questions it. From a graph perspective, this manifests as *rigid attention patterns with no exploration*—the model’s attention graph stabilizes early to a fixed configuration that reinforces the incorrect answer. Unlike drifting (Case Study 2), where attention patterns evolve and

Case Study 2: Inconsistent Guesses (Drifting)

Question: Who was manager of Sheffield Wednesday when they won the League Cup in 1991, beating Manchester United in the final?

Golden Label: Ron Atkinson

Denoising Trace:

- **Step 1:** The of Sheffield... (*Initial Noise*)
- **Step 9:** ...Kendall (*Plausible guess: Howard Kendall was a famous manager*)
- **Step 13:** David Kendall
- **Step 15:** Brian Kendall (*Final Hallucination—Incorrect*)

Final Output: Brian Kendall (Incorrect)

TDGNet Detection Score: 1 (Correctly Detected)

Case Study 3: Persistent Hallucination (Stuck)

Question: Did both Kanni and Bloodhound originate in the same continent?

Golden Label: No (Kanni is from India; Bloodhound is from Europe)

Denoising Trace:

- **Step 0:** Yes, both Kanni and Bloodhound originated in the same continent.
- **Step 8:** Yes, both Kanni and Bloodhound originated in the same continent.
- **Step 15:** Yes, both Kanni and Bloodhound originated in the same continent.

Final Output: Yes (Incorrect; factually false)

TDGNet Detection Score: 1 (Correctly Detected as hallucination)

can be captured as instability, or transient noise (Case Study 1), where instability is visible, persistence appears deceptively confident. However, our graph method detects this through a subtler signal: the attention pattern fails to *ground* in relevant parts of the prompt (e.g., attention should distribute across entity mentions “Kanni” and “Bloodhound” with appropriate geographic context). Instead, attention locks into a self-loop that reinforces the hallucinated answer. The message-passing aggregation exposes this as misalignment between the model’s internal attention structure and the prompt content.

F.4 Summary of Hallucination Patterns

These three cases illustrate why graph-based detection is superior to output-based methods: the internal attention structure reveals hallucination mechanisms that surface-level outputs conceal. Transient hallucinations with correct answers appear truthful to perplexity-based methods; drifting patterns build confidence that fools entropy metrics; persistent hallucinations appear confident and therefore low-risk. Only by analyzing the relational structure of attention can we detect all three types with high fidelity. Table 12 summarises these three patterns by their attention evolution, correctness, and required detection mechanism.

G Compute, Packages, and Reproducibility

G.1 Compute Infrastructure and Budget

TDGNet itself is a lightweight detector and trains in minutes on a single GPU. The dominant compute cost is the one-time extraction of attention traces from the underlying D-LLMs (LLaDA-8B, Dream-7B, LLaDA 1.5, LLaDA 2.1-mini). All experiments are conducted on $2 \times$ NVIDIA A40 GPUs (48 GB VRAM per card). Per-query inference overhead and the per-method wall-clock comparison are reported in Appendix A.2; theoretical complexity is analysed in Appendix C.

G.2 Software Packages

All experiments are implemented in Python 3.10 using PyTorch 2.4 with CUDA 12.1, and PyTorch Geometric 2.5 for graph construction and message passing. We load the underlying D-LLMs (LLaDA-8B, Dream-7B, LLaDA 1.5, LLaDA 2.1-mini) via HuggingFace Transformers 4.45, extracting per-layer attention weights and final-layer hid-

den states with default tokeniser and forward-pass settings. AUROC is computed with scikit-learn 1.5 (`sklearn.metrics.roc_auc_score`) from raw classifier probabilities; no thresholding or calibration is applied prior to scoring. We do not employ any third-party NLP toolkits (NLTK, SpaCy, ROUGE, etc.) at any stage of the pipeline.

G.3 Random Seeds and Reporting

We fix the random seed to 42 across all experiments and data splits (Appendix D). Following standard practice in the hallucination-detection literature (Kuhn et al., 2023; Manakul et al., 2023; Park et al., 2025; Chang et al., 2026), AUROC results are reported as single-seed values.

Table 12: Taxonomy of hallucination dynamics captured by graph-based detection. Each pattern corresponds to distinct attention graph evolution and requires structural analysis to detect.

Pattern	Attention Evolution	Output Correctness	Confidence	Detection Mechanism
Transient Noise	Unstable $\rightarrow$ Recovers	Correct	Low $\rightarrow$ High	Fragment detection
Drifting	Exploring $\rightarrow$ Locks wrong	Incorrect	Low $\rightarrow$ High	Semantic drift
Persistent	Fixed (wrong)	Incorrect	High (static)	Misalignment