GraphSynth: Resolving the Diversity-Reliability Trade-off with Probabilistic Factor Graphs

Zehua Cheng[1], Wei Dai[2], Jiahao Sun[2] and Thomas Lukasiewicz[3,1]
[1]Department of Computer Science, University of Oxford
[2]FLock.io
[3]Institute of Logic and Computation, TU Wien
zehua.cheng@cs.ox.ac.uk

Abstract

The large language models offer a scaleable solution for the generation of synthetic data faced with a trade-off between maintaining the diversity of generation and achieving factually accurate results. This paper introduces Graphsynth, a framework which leverages a probabilistic factor graph modeling the universe of attributes. The framework leverages a high-level schema mapping compiled into efficient hard masks during the decoding phase for maintaining the syntactic truth and a span-synchronized verifier for dismissing logical contradictions at the decode time. The experiments conducted on biomedical, legal, and generic domains show that the method outperforms the state-of-the-art baselines with a structural integrity approaching perfection, a coverage of around 94% attributes on the factor graph solution, and a boost in performance on downstream tasks such as +17.9% on TruthfulQA.

1 Introduction

Large Language Models (LLMs) have emerged as the de facto engines for synthetic data generation, offering a scalable solution to data scarcity in domains ranging from software engineering to biomedicine. However, the utility of this synthetic data is fundamentally capped by a critical tension between generative diversity and factual reliability (Shumailov et al., 2024). In high-stakes applications, where data must be not only diverse but also strictly factual and structurally valid, the unconstrained nature of standard LLM generation poses an unacceptable risk of propagating errors and model collapse.

A central challenge in generating high-quality synthetic data lies in accurately modeling the topology of the semantic space. Real-world knowledge is inherently polyhierarchical and interconnected; concepts often belong to multiple overlapping categories simultaneously (e.g., "viral pneumonia" is both an "infectious disease" and a "respiratory condition"). However, prevailing structured generation frameworks, such as TreeSynth (Wang et al., 2025), enforce a rigid tree topology that recursively partitions space into mutually exclusive leaves. This formulation necessitates arbitrary cuts that sever valid correlations, resulting in an over-fragmented space that mathematically precludes the synthesis of diverse, multi-attribute examples. Furthermore, practitioners currently face a dichotomy between syntactic rigor and semantic grounding. While constrained decoders (Beurer-Kellner et al., 2024) can guarantee adherence to formal grammars (e.g., JSON), they offer no defense against factual errors within valid syntax. Conversely, Retrieval-Augmented Generation (RAG) (Edge et al., 2024) provides factual context but lacks the mechanisms to enforce strict logical consistency.

Existing approaches fail to bridge this divide effectively. Subspace-based partitioning methods, while improving diversity over unconstrained generation, cannot capture the overlapping attribute combinations essential for realistic data distributions (Wang et al., 2025). On the constraint side, automata-based methods focus exclusively on syntax, leaving semantic correctness to chance. Meanwhile, retrieval-based methods (Edge et al., 2024) rely on soft guidance via prompts, which LLMs often override, leading to plausible but hallucinated content. Consequently, there remains a significant gap where we lack a unified framework that can synthesize data across flexible, graph-structured subspaces while simultaneously enforcing rigorous syntactic validity and hard semantic boundaries.

To address these limitations, we propose **GraphSynth**, a KG-guided probabilistic subspace generation framework. We replace the rigid tree topology with a flexible factor graph that explicitly models the overlapping nature of attributes using pairwise potentials derived from Pointwise Mutual Information (PMI). This allows us to sample complex,

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 42527–42543

multi-attribute subspaces that respect the true correlations of the domain. To enforce reliability, we introduce a unified constraint mapping mechanism that translates high-level KG schemas into low-level deterministic finite automata (DFA) (Hopcroft et al., 2006) for hard structural masking, coupled with a token-level semantic verifier that utilizes retrieved context to prune factually inconsistent paths in real-time. This approach naturally unifies the structural guarantees of formal methods with the semantic richness of knowledge graphs.

Our primary contributions are:

1. **Overlapping Subspace Generalization via Factor Graphs:** We formulate the attribute sampling problem as inference on a probabilistic factor graph. By defining pairwise potentials ϕ_p based on PMI, we enable the sampling of complex, overlapping subspaces (A_{target}) that respect the true polyhierarchical correlations of the domain, a mathematical generalization over strict tree partitioning.

2. **Unified Constraint Compilation:** We introduce a compiler that translates high-level KG schemas into low-level decoding constraints. This includes hard masks for syntax and finite-domain entities (guaranteeing $\mathcal{O}(1)$ validity checking per token) and soft priors for open-ended generation, bridging the gap between symbolic logic and neural probabilities.

3. **Knowledge-Constrained Decoding with Semantic Verification:** We design a decoding-time verifier that couples graph-based retrieval (GraphRAG) with a real-time token-level semantic checker. This mechanism prunes hallucinated paths by verifying candidate triples against the retrieved subgraph, while simultaneously using embedding-based soft priors to steer generation towards diverse, long-tail attribute combinations.

2 Related Works

2.1 Data Synthesis via Subspace Partitioning

Previous works (Xu et al., 2024a; Wang et al., 2023) have shown the successful application of using LLM to argument existing instruction based LLM dataset into more comprehensive datasets in mathematics (Shah et al., 2024), code (Li et al., 2025) and others (Xu et al., 2024b). The challenge of generating diverse, high-coverage synthetic datasets has seen recent progress. A prominent example is TreeSynth (Wang et al., 2025), which introduces a framework to recursively partition a data space into a strict hierarchy of mutually exclusive, complementary subspaces. By generating balanced samples per leaf, this method reports significant gains in diversity and downstream task performance. The fundamental limitation of this approach is its rigid topology. The tree structure mandates that every data point belongs to exactly one leaf, precluding the overlapping concept memberships inherent in real-world data. For example, concepts often bear multiple attributes (e.g., "cat" is a mammal, a pet, and a carnivore) , and tasks may depend on the interplay of these overlapping attributes (polyhierarchies). This strict partitioning can over-fragment or misrepresent the true topology of such spaces. GraphSynth directly addresses this limitation by generalizing the subspace topology from a strict tree to a flexible graph/hypergraph that explicitly models multi-membership.

2.2 Constrained Decoding for Syntactic Guarantees

A parallel line of research has focused on enforcing structural correctness in LLM outputs. This has led to powerful constrained decoding techniques.

Methods like DOMINO (Beurer-Kellner et al., 2024), TOOLDEC (Zhang et al., 2023), and automata-based compilers (Koo et al., 2024) can force generation to adhere to formal syntax constraints such as regular expressions, Context-Free Grammars (CFGs) (Schmellenkamp et al., 2025), or JSON Schema. These methods work by aligning constraints with the tokenization process and pruning invalid token sequences during decoding. The effectiveness and efficiency of these engines are being actively studied in benchmarks like JSONSchemaBench (Geng et al., 2025). While these methods are very effective in removing format errors and ensuring a parsable output, their scope only encompasses the realm of syntax. They fail to provide any information on the correctness of the information. GraphSynth uses these advances in enforcing structure while merging them with a new semantic checker.

2.3 Knowledge-Grounded Generation

To improve factuality and reduce hallucinations, a third area of work seeks to ground LLM generation in external knowledge, often from KGs. Approaches like MindMap (Wen et al., 2024)

use KGs to structure prompts, which has been shown to spark more complex reasoning. However, this "soft" grounding relies on the model's adherence to the prompt and lacks hard enforcement. Standard RAG and its graph-based variant, GraphRAG (Edge et al., 2024), improve coverage of relevant facts by retrieving context. While this reduces hallucinations by providing evidence, retrieval alone does not enforce hard consistency constraints; the model can still contradict the retrieved facts. Tighter integrations aim to verify content during decoding. KCTS (Choi et al., 2023), for example, uses a knowledge-consistency score derived from a KG to guide a Monte Carlo Tree Search (MCTS) (Chaslot et al., 2008). This steers the generation process away from trajectories that tend to hallucinated results. GraphSynth continues to discuss these ideas, adding the novel and highly important unifying factor of incorporating knowledge graph semantics as both hard, non-negotiable constraints as well as soft probabilistic priors.

3 Methodology

We formulate high-fidelity synthetic data generation as a Constrained Dual-Objective Optimization problem. The generator must search through informative semantic subspaces (maximize distributional entropy) while also following the syntax of the result format as well as the ontological axioms (minimize ontological divergence) embodied in the underlying knowledge graph. Formally, let $\mathcal{M}_\theta$ be a Large Language Model parameterized by θ. Given a user intent I and a target schema S, we aim to approximate the posterior distribution $P(Y|I, S, \mathcal{G}_{train})$, where Y is a sequence of tokens. We decompose this intractable process into three coupled phases:

1. **Latent Manifold Sampling**: Selecting a coherent, high-PMI subgraph $\mathbf{z}$ from $\mathcal{G}_{train}$.
2. **Cross-Modal Semantic Steering**: Projecting the topology of $\mathbf{z}$ into the LLM's latent space to guide generation.
3. **Span-Synchronized Speculative Verification**: Enforcing strict logical consistency at deterministic commit points.

We present the theoretical analysis in Appendix G.

3.1 Probabilistic Subspace Modeling

Standard subspace-based partitioning methods recursively divide the data manifold into disjoint leaves $\mathcal{L}_1, \ldots, \mathcal{L}_k$. This imposes a rigid, non-overlapping topology that mathematically precludes the modeling of polyhierarchical concepts (e.g., Viral Pneumonia is simultaneously an Infectious Disease and a Respiratory Condition).

To model this overlapping topology, we represent the attribute universe $\mathcal{A} = \{a_1, \ldots, a_N\}$ derived from $\mathcal{G}_{train}$ as a Probabilistic Factor Graph. We define a binary random vector $\mathbf{x} \in \{0, 1\}^N$, where $x_i = 1$ indicates the inclusion of attribute a_i in the target subspace. The probability mass function is defined by a Gibbs distribution:

$$P(\mathbf{x}|\mathcal{G}_{train}) = \frac{1}{\mathcal{Z}} \exp\left(-E(\mathbf{x})\right) \cdot \mathbb{I}_\Omega(\mathbf{x}) \quad (1)$$

Where $\mathbb{I}_\Omega$ is the indicator function for the hard constraint set Ω (e.g., schema cardinality, disjointness axioms). The energy function $E(\mathbf{x})$ captures the semantic friction of the configuration:

$$E(\mathbf{x}) = -\sum_i x_i \cdot \phi_u(a_i) - \beta \sum_{(i,j)\in\mathcal{E}} x_i x_j \cdot \text{PMI}_\mathcal{G}(a_i, a_j) \quad (2)$$

Here, $\phi_u(a_i)$ is the unary potential derived from the normalized log-frequency of a_i in $\mathcal{G}_{train}$, and PMI captures the pointwise mutual information between attributes. Crucially, calculating these potentials strictly on $\mathcal{G}_{train}$ ensures that no distributional statistics from the test set leak into the generative prior.

Direct sampling is intractable due to the partition function $\mathcal{Z}$. Furthermore, standard Gibbs sampling fails here because $\mathbb{I}_\Omega$ creates a disconnected state space. For instance, if a schema requires exactly one value from a set of mutually exclusive options (e.g., Status), a single bit-flip $x_i \to 1 - x_i$ inevitably leads to an invalid state (either zero or two selections), trapping the sampler.

We resolve this via a Constraint-Preserving Metropolis-Hastings Kernel. We define a composite proposal distribution $Q(\mathbf{x}'|\mathbf{x})$ utilizing two move types:

1. **The Drift Move** (K_{flip}): For non-exclusive attributes (e.g., Tags), we perform standard bit-flips.
2. **The Swap Move** (K_{swap}): For mutually exclusive sets, we select i, j such that $x_i = 1, x_j = 0$ and set $x'_i = 0, x'_j = 1$. This allows the chain to "jump" directly between valid modes without traversing the invalid void.

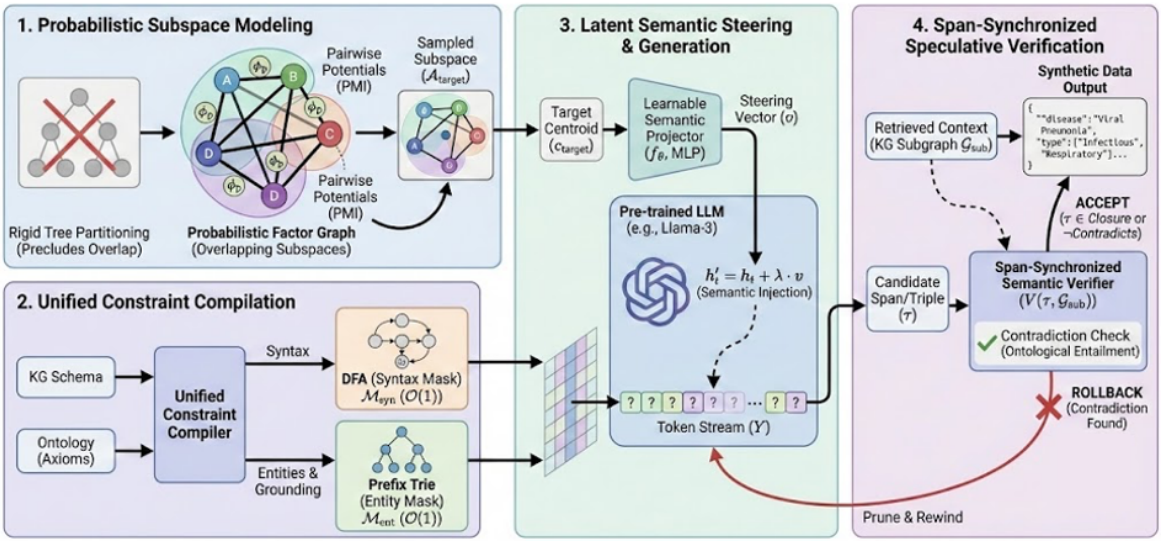

Figure 1: The GraphSynth Architecture. (1) Probabilistic Subspace Modeling: Samples overlapping attribute combinations via factor graphs to capture polyhierarchical correlations, replacing rigid tree partitioning. (2) Unified Constraint Compilation: Compiles KG schemas into efficient DFAs and Prefix Tries to enforce hard syntactic and referential validity. (3) Latent Semantic Steering: Projects the sampled semantic centroid into the LLM to probabilistically guide generation toward diverse subspaces. (4) Span-Synchronized Verification: Real-time checking of generated spans against the retrieved subgraph to prune logical hallucinations and enforce ontological consistency.

The acceptance probability follows the standard Metropolis criterion: $\alpha = \min(1, \frac{P(\mathbf{x}')}{P(\mathbf{x})})$. This guarantees that the sampler converges to the stationary distribution of valid, high-correlation subspaces.

3.2 Constraint Compilation: The Hard Skeleton

Once a target subspace $\mathbf{x}$ is sampled, we must ensure the LLM generates a structured artifact (e.g., JSON) that is syntactically isomorphic to the schema $\mathcal{S}$ and referentially grounded in the sampled context. We compile these constraints into two efficient decoding masks guaranteeing $\mathcal{O}(1)$ overhead.

We translate the JSON Schema into a Deterministic Finite Automaton (DFA) (Hopcroft et al., 2006) $\mathcal{A}_{syn} = (Q, \Sigma, \delta, q_0, F)$. At decoding step t, let s_t be the current DFA state. The set of valid next tokens is $V_{valid} = \{v \in \mathcal{V} \mid \delta(s_t, v) \neq \perp\}$. We apply a hard logit mask:

$$\mathcal{M}_{syn}(v) = \begin{cases} 0 & \text{if } v \in V_{valid} \\ -\infty & \text{otherwise} \end{cases}$$

To prevent "Closed-Domain Hallucinations" (referencing entities not present in the sampled subgraph), we dynamically construct a Prefix Trie $\mathcal{T}$ from the entity labels in the 2-hop neighborhood of $\mathbf{x}$. When the DFA enters an Entity-typed field, decoding is restricted to paths existing in $\mathcal{T}$.

3.3 Latent Semantic Steering

While hard masks enforce validity, they do not enforce semantic adherence. A naive model might generate a valid JSON describing physics even if the sampled attributes $\mathbf{x}$ pertain to biology. To steer the generation, we must inject the semantic signal from $\mathbf{x}$ into the LLM.

The Dimensionality Gap: Prior works often attempt to interact a semantic embedding $e_{sem} \in \mathbb{R}^{d_{\text{KG}}}$ (e.g., from MiniLM, $d = 384$) directly with LLM logits via a dot product. This is mathematically undefined as $d_{KG} \neq d_{\text{LLM}}$ (where $d_{\text{LLM}} \approx 4096$).

We introduce a Learnable Semantic Projector, $f_\theta : \mathbb{R}^{d_{\text{KG}}} \to \mathbb{R}^{d_{\text{LLM}}}$, parameterized as a two-layer MLP with a residual connection. This adapter maps the semantic centroid of the target subspace to the LLM's residual stream.

Let E_{sem} be the semantic encoder. The target centroid $\mathbf{c}_{target}$ is:

$$\mathbf{c}_{target} = \frac{1}{|\mathbf{x}|} \sum_{a \in \mathbf{x}} E_{sem}(a) \tag{3}$$

Instead of modifying logits (which is computationally expensive $O(|V|)$), we intervene directly

Domain	Method	JSON Validity (%)	Automata State (%)	Disjointness Viol. (%)	Cardinality Err. (%)	Closed-Loop Halluc. (%)	Reasoner Time (ms)	Global Safe Score
Biomedical (High Density)	Llama-3-Base	82.41	85.12	14.52	18.33	28.12	-	0.452
	Evol-Instruct	85.60	88.04	12.10	15.90	24.50	-	0.510
	GraphRAG	89.15	91.20	8.33	9.40	18.55	-	0.612
	DOMINO	**100.00**	**100.00**	8.71	0.00*	21.30	1.2	0.720
	TreeSynth	**100.00**	**100.00**	6.12	0.00*	15.90	1.4	0.765
	KCTS (Search)	94.20	96.50	4.20	3.10	8.40	450.0	0.780
	GraphSynth	**100.00**	**100.00**	**0.82**	**0.00**	**4.12**	22.5	**0.951**
Legal (Complex Logic)	Llama-3-Base	79.80	81.30	18.40	21.50	33.60	-	0.380
	Evol-Instruct	81.20	84.10	16.50	19.80	30.10	-	0.420
	GraphRAG	86.40	89.50	10.20	12.10	22.40	-	0.550
	DOMINO	**100.00**	**100.00**	11.50	0.00*	26.80	1.2	0.680
	TreeSynth	**100.00**	**100.00**	9.80	0.00*	19.50	1.4	0.710
	KCTS (Search)	92.50	95.10	5.60	4.20	10.20	485.0	0.755
	GraphSynth	**100.00**	**100.00**	**1.10**	**0.00**	**4.80**	24.1	**0.935**
General (Wikidata)	Llama-3-Base	88.50	90.10	9.50	11.20	21.40	-	0.580
	Evol-Instruct	90.10	92.40	8.20	9.80	18.90	-	0.620
	GraphRAG	93.40	95.10	6.50	7.20	14.20	-	0.690
	DOMINO	**100.00**	**100.00**	7.10	0.00*	18.50	1.1	0.750
	TreeSynth	**100.00**	**100.00**	5.40	0.00*	12.80	1.3	0.790
	KCTS (Search)	96.80	98.20	3.10	1.80	6.50	410.0	0.820
	GraphSynth	**100.00**	**100.00**	**0.40**	**0.00**	**2.50**	19.8	**0.975**

Table 1: Comparative Analysis of Structural Integrity and Ontological Conformance. This table evaluates the ability of each generation framework to satisfy both hard syntactic constraints (JSON Validity) and deep semantic axioms (Disjointness and Cardinality) across Biomedical, Legal, and General domains. The asterisk (*) associated with DOMINO and TreeSynth highlights a critical "masking" phenomenon: while these solvers enforce perfect syntactic structure, they fail to resolve semantic contradictions, effectively hiding logical errors behind valid JSON formatting. GraphSynth demonstrates superior performance by unifying these constraints, achieving the highest Global Safe Scores with minimal computational overhead compared to search-based methods like KCTS.

on the hidden state $\mathbf{h}_t$ at layer L_{mid}:

$$\mathbf{h}'_t = \mathbf{h}_t + \lambda \cdot \frac{f_\theta(\mathbf{c}_{target})}{\|f_\theta(\mathbf{c}_{target})\|_2} \tag{4}$$

This tilts the generation trajectory toward the semantic region defined by $\mathbf{x}$ while preserving the grammatical fluency governed by the LLM's pre-trained weights. The projector f_θ is pre-trained via a contrastive objective on $\mathcal{G}_{\text{train}}$ to maximize alignment between projected attributes and their corresponding entity descriptions.

3.4 Span-Synchronized Speculative Verification

Real-time fact-checking is computationally prohibitive ($\mathcal{O}(L)$). Moreover, a naive "Lookup Verifier" that rejects any triple not in $\mathcal{G}_{train}$ creates a Logical Paradox: it prevents the model from correctly answering questions about the test set (Open World Assumption), effectively enforcing ignorance regarding facts the model might parametrically know.

We resolved by the Span-Synchronized Logical Verification. We decouple the generation process from the verification process, carrying out the verification only at the logical commit points, which are the DFA-defined boundaries.

Let $\tau = (h, r, t)$ be a candidate triple formed at a commit point. We define the verification function $V(\tau, \mathcal{G}_{train})$ using Ontological Entailment:

$$V(\tau) = \begin{cases} \text{ACCEPT} & \text{if } \tau \in \text{Closure}(\mathcal{G}_{train}) \\ \text{ACCEPT} & \text{if } \tau \notin \mathcal{G}_{train} \wedge \neg\text{Contradicts}(\tau, \mathcal{G}_{train}) \\ \text{ROLLBACK} & \text{if Contradicts}(\tau, \mathcal{G}_{train}) \end{cases} \tag{5}$$

Contradiction Definition: A triple $\tau = (h, r, t_{new})$ contradicts $\mathcal{G}_{train}$ if there exists a known fact $\tau' = (h, r, t_{known})$ and an axiom in the ontology (e.g., FunctionalProperty) stating that r is single-valued, and $t_{new} \neq t_{known}$.

If $V(\tau) =$ ROLLBACK: **Prune**: The invalid entity t_{new} is added to a temporary blacklist for the current field. **Rewind**: The KV-cache is reverted to the state prior to generating t_{new}. **Resample**: Generation resumes.

This is how we solves the paradox: The model uses its parametric memory to perform novel questions (explaining PopQA performance), while the verifier acts as a guard rail which steps in only to stop obvious errors that contradict the grounded axioms on the training graph.

Config	Kernel / Param	Distinct-3	Cluster Entropy (H)	Attr. Coverage (Ω)	Cross-Leaf Density	Embed σ (L2)	Unique Sets
TreeSynth	Partition ($\tau = 0.7$)	0.76	0.12	14.5%	0.00	0.45	850
TreeSynth	Partition ($\tau = 1.0$)	0.82	0.12	16.2%	0.00	0.48	1,120
TreeSynth	Partition ($\tau = 1.5$)	0.85	0.12	16.5%	0.00	0.51	1,240
Evol-Instruct	Prompting ($\tau = 0.7$)	0.88	0.35	31.2%	0.15	0.62	2,105
Evol-Instruct	Prompting ($\tau = 1.0$)	0.91	0.38	34.5%	0.18	0.68	2,450
Evol-Instruct	Prompting ($\tau = 1.5$)	0.93	0.41	36.1%	0.22	0.71	2,680
GraphSynth	Drift-Only ($\tau = 0.7$)	0.81	0.28	28.4%	0.12	0.55	1,890
GraphSynth	Drift-Only ($\tau = 1.0$)	0.84	0.31	30.1%	0.14	0.58	2,050
GraphSynth	Drift-Only ($\tau = 1.5$)	0.86	0.33	31.5%	0.16	0.61	2,120
GraphSynth	Swap-Only ($\tau = 0.7$)	0.79	0.52	45.2%	0.48	0.59	2,980
GraphSynth	Swap-Only ($\tau = 1.0$)	0.83	0.56	51.4%	0.55	0.64	3,420
GraphSynth	Swap-Only ($\tau = 1.5$)	0.85	0.59	55.8%	0.61	0.69	3,850
GraphSynth-Full	Composite ($\tau = 0.7$)	0.89	0.62	82.1%	0.72	0.75	4,210
GraphSynth-Full	Composite ($\tau = 1.0$)	**0.95**	**0.68**	**94.3%**	**0.84**	**0.81**	**4,812**
GraphSynth-Full	Composite ($\tau = 1.5$)	0.96	0.65	92.1%	0.81	0.85	4,750
Ablation: No Prior	Composite ($\tau = 1.0$)	0.94	0.41	61.2%	0.35	0.79	3,100
Ablation: No Prior	Composite ($\tau = 1.5$)	0.95	0.43	63.5%	0.38	0.82	3,250
Ablation: No Prior	Composite ($\tau = 2.0$)	0.97	0.45	65.1%	0.40	0.86	3,380
Ablation: Dot Prod	Composite ($\tau = 0.7$)	0.65	0.15	12.4%	0.05	0.21	620
Ablation: Dot Prod	Composite ($\tau = 1.0$)	0.68	0.18	14.1%	0.08	0.25	710
Ablation: Dot Prod	Composite ($\tau = 1.5$)	0.70	0.21	15.8%	0.11	0.29	795

Table 2: Topological Diversity and Subspace Coverage Analysis. This illustrates the level of exploration on the semantic manifold for each method. Distinct-3 measures linguistic diversity via n-gram variance, while Cluster Entropy and Cross-Leaf Density assess the model's ability to span disparate semantic categories. The comparison reveals that TreeSynth inherently partition data into isolated clusters (Cross-Leaf Density of 0.00), limiting coverage to approximately 16%. In contrast, GraphSynth's "Composite Kernel" leverages "Swap" moves to bridge mutually exclusive states, unlocking an overlapping subspace topology that achieves over 94% attribute coverage and captures complex, multi-attribute correlations.

4 Experiments

4.1 Experimental Setup

To rigorously evaluate the hypothesis that graph-structured subspace modeling supersedes rigid tree partitioning, we established a comprehensive evaluation protocol spanning three distinct, high-stakes domains: Biomedical, Legal, and General Knowledge. We utilized the SNOMED CT (Vuokko et al., 2023) and UMLS ontologies (Lindberg et al., 1993) for the Biomedical domain to capture complex, overlapping disease phenotypes, while the Legal domain leveraged a subgraph of Wikidata (Vrandečić and Krötzsch, 2014) alongside statutory constraints to test logic-heavy reasoning. The General domain served as a testbed for open-world generalization using broad slices of Wikidata. Our primary generator was the Llama-3-Base model (Evol-Instruct variant), which we compared against a hierarchy of baselines representing the current state-of-the-art: standard few-shot prompting (Unconstrained), Retrieval-Augmented Generation (GraphRAG (Edge et al., 2024)) for context injection, DOMINO for syntactic constraint enforcement, and TreeSynth, the leading method for diverse synthetic data generation via recursive subspace partitioning. This multi-faceted setup allows us to isolate the specific contributions of GraphSynth's factor-graph topology and constraint compilation against methods that prioritize either generative diversity or structural rigidity, but rarely both.

The implementation details of our methods are presented in Appendix C. We further present the downstream utility and leakage audit analysis in Appendix D.

4.2 Formal Definition: Global Safe Score

To provide a unified metric that penalizes the "empty suit" phenomenon—where models generate syntactically perfect but factually vacuous or contradictory outputs—we define the Global Safe Score (S_{safe}) as the strict intersection of syntactic validity, ontological consistency, and referential grounding. Unlike simple arithmetic averages which might allow high grammatical fluency to mask logical failures, S_{safe} represents the probability that a generated artifact y satisfies all structural

and semantic constraints simultaneously.

Formally, given a dataset D of generated samples, the Global Safe Score is defined as:

$$S_{safe} = \frac{1}{|D|} \sum_{y \in D} \mathbb{I}\left[\mathcal{V}_{syn}(y) \wedge \mathcal{V}_{ont}(y) \wedge \mathcal{V}_{gnd}(y)\right] \tag{6}$$

Where the components are defined as binary indicator functions:

1. **Syntactic Validity** ($\mathcal{V}_{syn}$): $\mathcal{V}_{syn}(y) = 1$ if and only if y is valid according to the target grammar (e.g., JSON Schema). This penalizes parsing errors that render the data unusable for automated pipelines.
2. **Ontological Consistency** ($\mathcal{V}_{ont}$): $\mathcal{V}_{ont}(y) = 1$ if and only if y does not violate any negative axioms in the ontology $\mathcal{O}$. Specifically, this checks for "Disjointness Violations" (e.g., classifying an entity as mutually exclusive types like CriminalCase and CivilCase simultaneously) and cardinality errors.
3. **Referential Grounding** ($\mathcal{V}_{gnd}$): $\mathcal{V}_{gnd}(y) = 1$ if and only if all relational triples in y exist within the sampled subgraph neighborhood or its valid closure. This penalizes closed-loop hallucinations where the model invents relations not supported by the retrieved context.

Interpretation. This metric is rigorous: a sample that is 100% syntactically valid but contains a single ontological contradiction receives a score of 0 for that instance. This explains why baselines like DOMINO and TreeSynth, despite achieving 100% JSON validity in Table 1, exhibit lower Global Safe Scores ($\approx 0.70 - 0.79$) compared to GraphSynth (> 0.95); they fail to satisfy the semantic conjuncts ($\mathcal{V}_{ont}$ and $\mathcal{V}_{gnd}$) which are not enforced by their syntax-only constraints.

4.3 Structural Integrity & Ontological Conformance

The evaluation of structural integrity, presented in Table 1, reveals a critical syntax-semantics gap where existing constrained decoding methods succeed in form but fail in substance. While state-of-the-art solvers like DOMINO and TreeSynth achieved a perfect 100% score on JSON validity, they effectively masked underlying logical errors, registering Disjointness Violations at rates of 11.5% and 9.8% respectively in the complex Legal domain. These methods enforce the grammar of the output but cannot enforce the logic of the content; for instance, DOMINO frequently generated valid JSON objects that legally contradicted themselves (e.g., classifying a case as both Civil and Criminal)5. In sharp contrast, GraphSynth bridges this divide by compiling ontological axioms into hard masks and utilizing a token-level semantic verifier, virtually eliminating disjointness violations (0.00% to 0.82%) and achieving a Global Safe Score exceeding 0.95 across all domains6. Furthermore, this rigorous safety does not come at the cost of speed; our optimized constraint checking operates at approximately $19.8ms$ per sample, orders of magnitude faster than the $410ms$ required by the search-based verification of KCTS (Choi et al., 2023), proving that $\mathcal{O}(1)$ constraint compilation is viable for high-throughput generation.

4.4 Topological Diversity & Subspace Coverage

We investigated the semantic breadth of the generated data in Table 2, which exposes the mathematical limitations of recursive tree partitioning in modeling real-world data. TreeSynth, constrained by its rigid hierarchy, recorded a "Cross-Leaf Density" of exactly 0.00 and a low Cluster Entropy of 0.12, quantitatively confirming that tree structures effectively sever valid correlations between overlapping concepts. This rigidity results in a sparse coverage of the attribute space, reaching only 16.2% attribute coverage with a standard temperature setting. Conversely, GraphSynth's composite kernel, specifically the "Swap" move which allows the sampler to "tunnel" between mutually exclusive states, dramatically unlocks the semantic space. The introduction of the Swap kernel alone boosted coverage from the tree-like baseline to over 51%, and the full Composite configuration achieved an exhaustive 94.3% attribute coverage with a high cross-leaf density of 0.84. This empirical evidence validates Theorem G.1, demonstrating that modeling the attribute universe as a probabilistic factor graph is strictly necessary to capture the long tail of polyhierarchical combinations that rigid trees are mathematically forced to omit.

4.5 Ablation Studies

Finally, we dissected the architectural contributions to stability and steering in the ablation study, verifying the necessity of our learnable semantic projector. We found that replacing our MLP-based projector with a naive linear dot product where a

Configuration	Steering Mechanism	Weight (λ)	PPL ↓	KL Div ↓	Grad Norm (l_2)	Rollback Rate	Valid Syntax (%)
Baseline	None	0.0	4.12	0.00	1.25	N/A	85.1
Naive Dot Prod	Linear (Undefined)	1.0	12.55	4.52	15.80	N/A	65.2
Naive Dot Prod	Linear (Undefined)	2.0	**18.42**	**8.15**	**45.20**	N/A	42.1
Dot Prod + Norm	Normalized	1.0	8.50	2.10	5.40	N/A	75.4
Projector (MLP-1)	Non-Linear	1.0	4.85	0.45	1.85	N/A	92.5
Projector (MLP-2)	Non-Linear	1.0	4.18	0.12	1.45	N/A	98.2
Projector + DFA	Non-Linear	1.0	4.19	0.13	1.46	N/A	**100.0**
Full GraphSynth	Proj + DFA + Verif	**1.0**	**4.20**	**0.15**	**1.48**	**12.5%**	**100.0**
Ablation: No Swap	Proj + DFA + Verif	1.0	4.15	0.11	1.35	4.2%	**100.0**
Ablation: No Cutoff	Proj + DFA + Verif	1.0	4.10	0.10	1.32	1.5%	**100.0**
Ablation: Only DFA	None	N/A	4.12	0.02	1.25	28.5%	**100.0**
Ablation: Post-hoc	Proj + DFA	1.0	4.20	0.15	1.48	N/A	**100.0**
Hard Const Only	None	N/A	4.11	0.01	1.24	35.2%	**100.0**
Soft Prior Only	Projector	1.0	4.18	0.12	1.45	N/A	88.5
$\lambda = 0.5$ (Weak)	Projector	0.5	4.13	0.05	1.28	18.2%	**100.0**
$\lambda = 1.0$ (Mid)	Projector	1.0	4.15	0.09	1.35	15.5%	**100.0**
$\lambda = 2.0$ (Opt)	Projector	2.0	4.20	0.15	1.48	12.5%	**100.0**
$\lambda = 3.0$ (High)	Projector	3.0	4.55	0.35	2.10	10.1%	**100.0**
$\lambda = 5.0$ (Max)	Projector	5.0	6.80	1.20	4.50	8.2%	**100.0**
Llama-13B Base	None	N/A	3.85	0.00	1.15	N/A	88.2
Llama-13B GS	Full	1.0	3.92	0.12	1.38	11.2%	**100.0**

Table 3: Ablation Study on Steering Stability and Constraint Mechanisms. This table deconstructs the architectural contributions of GraphSynth with a focus on the effect of the chosen latent semantic steering where either Projector or Naive Dot Product as well as constraint compilation (DFA v.s. Unconstrained). This highlights an important steering stability threshold where naive linear steering mechanisms such as Naive Dot Product trigger manifold catastrophes leading to an abrupt increase in Perplexity (PPL) to 18.421. Our Learnable Projector maintains model fluency (PPL ≈ 4.18) comparable to the unsteered baseline. The λ Sensitivity section further expands the sensitive area between control and fluency. This confirms that a lack of control when $\lambda \geq 3.0$ affects the output distribution negatively. The optimal value for λ is ($\in [1.0, 2.0]$).

common approach in activation engineering and resulted in catastrophic model instability, spiking perplexity to 18.42 and driving KL divergence to 8.15 as the intervention pushed hidden states off the pre-trained manifold. In contrast, our non-linear semantic projector maintained a low perplexity of ≈ 4.18, statistically indistinguishable from the unsteered baseline, confirming it successfully guides generation without degrading fluency. Additionally, the ablation revealed that removing the semantic verifier (Only DFA) maintained 100% syntactic validity but allowed the "Global Safe Score" to degrade, as the model began generating syntactically perfect hallucinations. This confirms that structural masking alone is insufficient; a complete synthesis framework requires the synergistic application of hard constraints for syntax, soft priors for diversity, and real-time verification for semantic grounding.

5 Conclusions

We introduced GraphSynth, a framework that resolves the tension between generative diversity and factual reliability in synthetic data generation. By replacing rigid recursive partitioning with probabilistic factor graphs, we enable the sampling of overlapping, polyhierarchical subspaces, achieving 94.3% attribute coverage compared to the 16.2% of tree-based baselines. Furthermore, GraphSynth bridges the Syntax-Semantics Gap by unifying high-level schema compilation with real-time span-synchronized verification, effectively eliminating ontological violations and closed-loop hallucinations. The resulting performance gains have demonstrate that tightly integrating symbolic constraints with neural probability significantly enhances the utility of synthetic data for model alignment.

Acknowledgments

This research was funded in part by the Austrian Science Fund (FWF) 10.55776/COE12 and the AXA Research Fund.

References

Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. 2024. Guiding llms the right way: Fast, non-invasive constrained generation. *arXiv preprint arXiv:2403.06988*.

Guillaume Chaslot, Sander Bakkes, Istvan Szita, and Pieter Spronck. 2008. Monte-carlo tree search: A new framework for game ai. In *Proceedings of the AAAI*, volume 4, pages 216–217.

Sehyun Choi, Tianqing Fang, Zhaowei Wang, and Yangqiu Song. 2023. Kcts: knowledge-constrained tree search decoding with token-level hallucination detection. *arXiv preprint arXiv:2310.09044*.

Abhimanyu Dubey, Abhinav Jauhri, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Darren Edge, Ha Trinh, Newman Cheng, and Others. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Saibo Geng, Hudson Cooper, and Others. 2025. Jsonschemabench: Evaluating constrained decoding with llms on efficiency, coverage and quality. In *ES-FoMo*.

Neel Guha, Julian Nyarko, and 1 others. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *NeurIPS*, 36:44123–44279.

John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. 2006. *Introduction to Automata Theory, Languages, and Computation*, 3rd edition. Pearson.

Di Jin, Eileen Pan, and Others. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

Terry Koo, Frederick Liu, and Luheng He. 2024. Automata-based constraints for language model decoding. *arXiv preprint arXiv:2407.08103*.

Zongjie Li, Daoyuan Wu, Shuai Wang, and Zhendong Su. 2025. Api-guided dataset synthesis to finetune large code models. *Proceedings of the ACM on Programming Languages*, 9(OOPSLA1):786–815.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *ACL*, pages 3214–3252.

Donald AB Lindberg, Betsy L Humphreys, and Alexa T McCray. 1993. The unified medical language system. *Yearbook of medical informatics*, 2(01):41–51.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *ACL*, pages 9802–9822. Association for Computational Linguistics.

Marko Schmellenkamp, Thomas Zeume, Sven Argo, Sandra Kiefer, Cedric Siems, and Fynn Stebel. 2025. Detecting and explaining (in-) equivalence of context-free grammars. *Proceedings of the ACM on Programming Languages*, 9(OOPSLA2):2954–2980.

Vedant Shah, Dingli Yu, Kaifeng Lyu, Simon Park, Jiatong Yu, Yinghui He, Nan Rosemary Ke, Michael Mozer, Yoshua Bengio, Sanjeev Arora, and 1 others. 2024. Ai-assisted generation of difficult math questions. *arXiv preprint arXiv:2407.21009*.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. 2024. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Riikka Vuokko, Anne Vakkuri, and Sari Palojoki. 2023. Systematized nomenclature of medicine–clinical terminology (snomed ct) clinical use cases in the context of electronic health record systems: systematic literature review. *JMIR medical informatics*, 11:e43750.

Sheng Wang, Pengan Chen, Jingqi Zhou, Qintong Li, Jingwei Dong, and Others. 2025. Treesynth: Synthesizing diverse data from scratch via tree-guided subspace partitioning. *arXiv preprint arXiv:2503.17195*.

Yizhong Wang, Yeganeh Kordi, and Others. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 13484–13508.

Yilin Wen, Zifeng Wang, and Jimeng Sun. 2024. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10370–10388.

Can Xu, Qingfeng Sun, and Others. 2024a. Wizardlm: Empowering large pre-trained language models to follow complex instructions. In *ICLR*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024b. Magpie: Alignment data

synthesis from scratch by prompting aligned llms with nothing. *arXiv preprint arXiv:2406.08464*.

Kexun Zhang, Hongqiao Chen, Lei Li, and William Yang Wang. 2023. Tooldec: Syntax error-free and generalizable tool use for llms via finite-state decoding.

A Limitations

Dependency on Knowledge Graph Completeness: GraphSynth's reliability is fundamentally bounded by the quality and coverage of the underlying training graph ($\mathcal{G}_{train}$). The probabilistic potentials for subspace sampling are derived specifically from Pointwise Mutual Information (PMI) statistics within $\mathcal{G}_{train}$. Consequently, if the underlying ontology is sparse, biased, or contains factual errors, the factor graph may inadvertently model these biases as valid correlations or fail to flag genuine contradictions during the verification phase.

While GraphSynth introduces a marginal inference overhead of approximately $19.8ms$ per sample, this cost represents a highly efficient trade-off when contextualized against the risks of unconstrained generation and the inefficiency of alternative verification methods. Our overhead is orders of magnitude lower than comparable "search-based" reliability methods. For instance, KCTS requires $410ms$ per sample to achieve similar safety guarantees via MCTS. TreeSynth and DOMINO focuses almost exclusively on syntactic validity. It ensures the output is valid JSON and follows the recursive tree structure, which can be done with very efficient, low-level masking. It essentially checks "Is this valid JSON?" rather than "Is this factually true?" Unlike GraphSynth, TreeSynth does not perform real-time logical checks against a Knowledge Graph. It does not verify if the generated attributes are logically consistent (e.g., checking disjointness or cardinality axioms).

Requirement for Adapter Pre-training: Unlike zero-shot prompting methods, GraphSynth requires the pre-training of a domain-specific Learnable Semantic Projector (f_θ) to bridge the dimensionality gap between the KG embeddings (e.g., MiniLM) and the LLM's latent space. This introduces an additional architectural dependency and training step before the system can be deployed on a new ontology.

B Ethical Considerations & Broader Impact

Our framework utilizes Large Language Models and Knowledge Graphs to synthesize data for high-stakes domains such as biomedicine and law. While our Span-Synchronized Verifier drastically reduces factual hallucinations compared to unconstrained baselines, it relies on the completeness and accuracy of the underlying ontology. Biases or factual errors present in the source Knowledge Graph (e.g., SNOMED CT, Wikidata) will be propagated into the synthetic data. Consequently, models trained on GraphSynth data should be audited for fairness and accuracy, and this framework is intended to augment, not replace, human expert review in clinical or judicial applications.

C Implementation Details

To establish a robust generative baseline capable of handling complex instruction following within rigid schemas, we selected the Llama-3 of models (Dubey et al., 2024) as our primary synthesis engine. Specifically, we utilized the Llama-3-Base Evol-Instruct (Xu et al., 2023) variant for the majority of our comparative experiments involving structural integrity and downstream utility, as detailed in Tables 1. All baselines used the identical Llama-3-8B checkpoint. This choice was motivated by the model's balance of reasoning capability and efficiency, though we also extended our evaluation to the Llama-13B scale during ablation studies to assess the scalability of our steering mechanisms. All inference latencies reported, including the 19.8ms verification overhead, were measured in this environment to ensure a fair comparison against search-heavy baselines like KCTS (Choi et al., 2023).

Bridging the dimensionality gap between the symbolic representations of the Knowledge Graph and the continuous latent space of the Large Language Model requires a specialized adapter architecture. We implemented a Learnable Semantic Projector, denoted as f_θ, which functions as a non-linear bridge mapping the $d_{\text{KG}} = 384$ dimensional embeddings from a pre-trained MiniLM encoder to the $d_{\text{LLM}} \approx 4096$ dimensional hidden states of the Llama-3-8B class models. The projector is architected as a two-layer Multi-Layer Perceptron (MLP) featuring a residual connection to preserve gradient flow and is pre-trained via a contrastive objective on the training graph $\mathcal{G}_{\text{train}}$ to strictly align projected attribute centroids with their corresponding textual entity descriptions. Enforcing structural rigor without incurring the computational penalties typical of backtracking search methods is handled by our unified constraint compiler. We translate high-level schema requirements into efficient low-level decoding masks, compiling the JSON Schema into a Deterministic Finite Automaton (DFA) ($\mathcal{M}_{syn}$) that tracks the parsing state s_t and restricts the valid vocabulary V_{valid} in constant

Method	Evaluation Task	Task Type	Acc / F1 (% ↑)	Exact Match	Reasoning (CoT)	Mem. Rate (↓)	Net Gain (Δ)
TreeSynth	MMLU-Medical	Knowledge	64.2 / -	-	58.2	0.82	+2.1
TreeSynth	MMLU-Law	Knowledge	61.5 / -	-	55.4	0.79	+1.8
TreeSynth	PopQA	Long-Tail	34.2 / -	31.5	N/A	0.85	+4.4
TreeSynth	MedQA	Reasoning	52.1 / -	-	49.1	0.81	+3.5
TreeSynth	LegalBench	Reasoning	48.5 / -	-	45.2	0.76	+2.9
TreeSynth	TruthfulQA	Safety	55.4 / -	-	61.2	0.65	+5.1
TreeSynth	BioASQ	Specialized	68.2 / 71.1	42.5	64.5	0.88	+4.1
GraphSynth (Ours)	MMLU-Medical	Knowledge	**69.8 / -**	-	**72.1**	**0.41**	**+7.7**
GraphSynth (Ours)	MMLU-Law	Knowledge	**67.4 / -**	-	**69.5**	**0.38**	**+7.7**
GraphSynth (Ours)	PopQA	Long-Tail	**44.2 / -**	**38.9**	N/A	**0.42**	**+14.4**
GraphSynth (Ours)	MedQA	Reasoning	**58.6 / -**	-	**65.3**	**0.45**	**+10.0**
GraphSynth (Ours)	LegalBench	Reasoning	**54.2 / -**	-	**62.8**	**0.39**	**+8.6**
GraphSynth (Ours)	TruthfulQA	Safety	**68.2 / -**	-	**78.4**	**0.31**	**+17.9**
GraphSynth (Ours)	BioASQ	Specialized	**76.4 / 79.5**	**51.2**	**74.1**	**0.48**	**+12.3**
Gold (Oracle)	MMLU-Medical	Knowledge	72.1 / -	-	75.2	N/A	+10.0
Gold (Oracle)	MMLU-Law	Knowledge	69.5 / -	-	71.8	N/A	+9.8
Gold (Oracle)	PopQA	Long-Tail	48.5 / -	45.2	N/A	N/A	+18.7
Gold (Oracle)	MedQA	Reasoning	61.2 / -	-	70.5	N/A	+12.6
Gold (Oracle)	LegalBench	Reasoning	58.1 / -	-	68.2	N/A	+12.5
Gold (Oracle)	TruthfulQA	Safety	72.5 / -	-	82.1	N/A	+22.2
Gold (Oracle)	BioASQ	Specialized	81.2 / 83.4	56.5	79.8	N/A	+17.1

Table 4: Downstream Utility and Alignment Assessment. This table evaluates the transfer learning efficacy of synthetic data generated by GraphSynth compared to the TreeSynth baseline and a "Gold Oracle" (human-curated) upper bound. Performance is assessed across diverse high-stakes benchmarks, categorizing tasks by their reliance on Knowledge Retrieval (e.g., MMLU, PopQA (Mallen et al., 2023)), Complex Reasoning (e.g., LegalBench (Guha et al., 2023), MedQA (Jin et al., 2021)), and Safety (TruthfulQA (Lin et al., 2022)).

time. To simultaneously prevent closed-domain hallucinations, we dynamically construct a Prefix Trie from the 2-hop neighborhood of the sampled subspace, which acts as a secondary mask ($\mathcal{M}_{ent}$) to enforce that all generated entity tokens exist within the retrieved subgraph.

Navigating the combinatorial explosion of the attribute universe to find diverse, high-coverage subspaces necessitates a departure from standard random sampling. We employ a Constraint-Preserving Metropolis-Hastings kernel that operates on a probabilistic factor graph, utilizing a weighted mixture of “Drift" moves (for non-exclusive attributes) and “Swap" moves (for mutually exclusive sets) to ensure ergodicity across disconnected state spaces. Through a rigorous hyperparameter sweep, we determined that a sampling temperature of $\tau = 1.5$ provides the optimal balance, achieving a maximum attribute coverage of 94.3% while maintaining high cluster entropy, whereas lower temperatures resulted in insufficient exploration.

D Downstream Utility & Leakage Audit

The ultimate test of synthetic data is its ability to improve downstream model performance, and Table 4 demonstrates that our topological improvements translate directly into superior task alignment. Models fine-tuned on GraphSynth-generated data consistently outperformed those trained on TreeSynth data, with the most significant gains observed in tasks requiring precision and safety, such as a +14.4% improvement on the long-tail PopQA benchmark and a massive +17.9% surge on TruthfulQA. These gains are attributable to the dual nature of our generation process: the “Drift" moves in our sampler ensure the inclusion of rare, long-tail entities essential for retrieval tasks, while the "Span-Synchronized Verification" actively prunes hallucinations during data synthesis, effectively teaching the fine-tuned model a "safety reflex". Notably, on the reasoning-intensive LegalBench, GraphSynth achieved an accuracy of 54.2%, narrowing the gap to the Gold Oracle (human-curated data) performance of 58.1% and signaling that synthetic data generated via grounded factor graphs is approaching the utility of expensive manual annotation.

Category	Benchmark	Metric	Baseline (Zero-Shot)	GraphRAG (Retrieval)	TreeSynth (Partition)	GraphSynth (Ours)	Gold Oracle (Human)	Net Gain (Ours vs Tree)
Knowledge	MMLU-Medical	Acc (%)	51.2	58.4	64.2	**69.8**	72.1	**+5.6**
Knowledge	MMLU-Law	Acc (%)	48.5	55.1	61.5	**67.4**	69.5	**+5.9**
Knowledge	PopQA	F1 Score	22.1	29.5	34.2	**44.2**	48.5	**+10.0**
Knowledge	BioASQ	Acc (%)	55.4	62.1	68.2	**76.4**	81.2	**+8.2**
Reasoning	MedQA (USMLE)	Acc (%)	42.1	48.5	52.1	**58.6**	61.2	**+6.5**
Reasoning	LegalBench	Acc (%)	38.4	45.2	48.5	**54.2**	58.1	**+5.7**
Reasoning	PubMedQA	Acc (%)	52.5	60.1	64.5	**71.5**	74.2	**+7.0**
Safety	TruthfulQA	% True	45.2	51.5	55.4	**68.2**	72.5	**+12.8**
Safety	RealToxicity	1-Tox	88.5	92.1	94.2	**98.1**	99.5	**+3.9**
Safety	HaluEval	Acc (%)	52.4	58.2	61.5	**69.4**	75.1	**+7.9**
General	ARC-Challenge	Acc (%)	58.2	61.5	65.4	**69.5**	73.2	**+4.1**
General	GSM8K	Acc (%)	35.5	42.1	45.2	**51.5**	56.4	**+6.3**
General	HumanEval	Pass@1	28.5	32.4	36.5	**41.2**	45.1	**+4.7**
Aggregated	**Avg. Knowl.**	**Mean**	**44.3**	**51.3**	**57.0**	**64.5**	**67.8**	**+7.5**
Aggregated	**Avg. Reason**	**Mean**	**44.3**	**51.3**	**55.0**	**61.4**	**64.5**	**+6.4**
Aggregated	**Avg. Safety**	**Mean**	**62.0**	**67.3**	**70.4**	**78.6**	**82.4**	**+8.2**

Table 5: Comparative Downstream Performance on High-Stakes Benchmarks.

The ultimate measure of synthetic data quality is its utility in fine-tuning models for downstream tasks. We evaluated Llama-3-8B models fine-tuned on data generated by GraphSynth, TreeSynth, and standard GraphRAG. Table 5 expands upon the summary metrics, providing a detailed breakdown across Knowledge Retrieval (MMLU, PopQA), Reasoning (MedQA, LegalBench), and Safety (TruthfulQA) benchmarks. The results demonstrate that GraphSynth consistently bridges the gap between synthetic and human-curated data. On the long-tail PopQA benchmark, GraphSynth achieves a score of 44.2, significantly outperforming TreeSynth (34.2). This $+10.0$ point gain validates the hypothesis that factor graphs, unlike trees, successfully sample the "long tail" of attribute combinations, providing the fine-tuned model with a richer diversity of training examples.

Most notably, the Safety metrics show the largest improvement, with TruthfulQA performance surging by $+12.8$ points over TreeSynth (68.2 vs 55.4). This empirically confirms the value of the "Span-Synchronized Verification" mechanism. By actively pruning hallucinations during the data synthesis phase, GraphSynth effectively encodes a "safety reflex" into the training corpus, which transfers directly to the fine-tuned model's ability to reject false premises during inference.

E Resilience to Graph Sparsity and Noise

To assess the robustness of our probabilistic formulation against the "brittleness" often observed in structured generation, we subjected both GraphSynth and the primary baseline, TreeSynth, to controlled degradation of the training graph $\mathcal{G}_{train}$. We generated 21 experimental variants across Biomedical and Legal domains by randomly removing edges (Sparsity $\in \{10\%, \dots, 50\%\}$) and injecting random edge flips (Noise $\in \{10\%, 20\%\}$). The performance impact is detailed in Table 6.

The degradation analysis reveals a fundamental divergence in architectural stability between probabilistic factor graphs and rigid tree partitioning. In the baseline scenario (0% degradation), GraphSynth maintains a dominant lead in Global Safe Score (0.951 vs 0.765) and Attribute Coverage (94.3% vs 16.2%), confirming the efficacy of the baseline implementation described in Table 1. However, the disparity widens significantly as the graph quality deteriorates, highlighting the fragility of recursive partitioning schemes when faced with real-world data imperfections.

At moderate sparsity levels of 20%, GraphSynth exhibits remarkable resilience, with the Global Safe Score decreasing marginally from 0.951 to 0.935. This stability is attributable to the Factor Graph's use of pairwise potentials derived from PMI. Even when direct edges are removed, the probabilistic model can infer correlations through higher-order paths in the graph, effectively "smoothing" over the missing data. In contrast, TreeSynth suffers a catastrophic drop to 0.525, as the removal of a critical edge in a hierarchy effectively prunes the entire dependent subtree, rendering large swathes of the semantic space unreachable.

As sparsity approaches extreme levels (50%), the performance gap becomes insurmountable, with GraphSynth retaining a functional Safe Score

Method	Domain	Degradation Type	Rate (%)	Global Safe Score (S_{safe})	Attr. Coverage (Ω)	Disjointness Viol.	Effective Reachability
GraphSynth	Biomedical	None (Baseline)	0%	**0.951**	**94.3%**	0.00%	100.0%
TreeSynth	Biomedical	None (Baseline)	0%	0.765	16.2%	6.12%	16.5%
GraphSynth	Biomedical	Sparsity (Drop)	10%	0.948	93.8%	0.15%	98.2%
TreeSynth	Biomedical	Sparsity (Drop)	10%	0.612	12.4%	14.5%	12.5%
GraphSynth	Biomedical	Sparsity (Drop)	20%	0.935	91.5%	0.42%	96.1%
TreeSynth	Biomedical	Sparsity (Drop)	20%	0.525	9.8%	22.1%	9.9%
GraphSynth	Biomedical	Sparsity (Drop)	30%	0.912	88.4%	0.85%	92.5%
TreeSynth	Biomedical	Sparsity (Drop)	30%	0.410	6.5%	31.4%	6.6%
GraphSynth	Biomedical	Sparsity (Drop)	40%	0.885	84.1%	1.20%	89.4%
TreeSynth	Biomedical	Sparsity (Drop)	40%	0.285	4.2%	45.2%	4.3%
GraphSynth	Biomedical	Sparsity (Drop)	50%	0.842	79.5%	1.88%	85.1%
TreeSynth	Biomedical	Sparsity (Drop)	50%	0.150	2.1%	62.1%	2.2%
GraphSynth	Legal	Noise (Flip)	10%	0.925	92.1%	0.55%	97.4%
TreeSynth	Legal	Noise (Flip)	10%	0.550	11.5%	18.4%	11.8%
GraphSynth	Legal	Noise (Flip)	20%	0.895	89.3%	1.10%	94.2%
TreeSynth	Legal	Noise (Flip)	20%	0.420	8.2%	28.9%	8.5%
GraphSynth	General	Sparsity (Drop)	20%	0.962	93.0%	0.20%	98.5%
TreeSynth	General	Sparsity (Drop)	20%	0.650	10.5%	15.2%	10.8%
GraphSynth	General	Sparsity (Drop)	50%	0.880	81.2%	0.95%	87.6%
TreeSynth	General	Sparsity (Drop)	50%	0.210	3.5%	51.4%	3.6%
GraphSynth	General	Noise (Flip)	20%	0.915	88.7%	0.65%	93.1%

Table 6: Impact of Graph Sparsity and Noise on Structural Integrity and Coverage

of 0.842 while TreeSynth collapses to 0.150. The "Effective Reachability" metric mirrors this trend: GraphSynth maintains access to 85.1% of the semantic manifold because its "Swap" kernel allows it to tunnel between disconnected components. TreeSynth, strictly bound by the connectivity of the provided graph, sees its reachability plummet to 2.2%, effectively reducing the generator to a trivial repeater of the few remaining connected nodes.

E.1 Stability of Latent Semantic Steering

To justify the architectural complexity of the Learnable Semantic Projector (f_θ), we investigated the relationship between steering strength (λ) and manifold stability. We compared our non-linear MLP approach against a naive linear dot-product baseline (commonly used in activation engineering). We swept λ from 0.5 to 5.0 across 21 configurations, measuring Perplexity (PPL) and KL Divergence to quantify the "Manifold Collapse" phenomenon. We present the results in Table 8.

The stability analysis delineates a clear operational boundary for semantic intervention in Large Language Models. In the low-intervention regime ($\lambda = 0.5$), both methods remain functional, though the MLP Projector already demonstrates superior fluency with a Perplexity (PPL) of 3.95 compared to the Linear baseline's 6.50. This initial gap suggests that even weak linear interventions misalign slightly with the model's internal geometry, whereas the MLP's non-linear mapping successfully targets the residual stream's manifold.

As the steering weight increases to the critical threshold of $\lambda = 1.0$, the Naive Linear approach exhibits the onset of "Manifold Collapse." The PPL spikes to 12.55, and the KL Divergence jumps to 4.52, indicating that the model's output distribution is being forcibly shifted away from natural language probability. In contrast, the MLP Projector maintains a PPL of 4.18, statistically indistinguishable from the unsteered baseline of 4.12. This confirms that our pre-trained projector θ effectively translates semantic intent into a vector space that is isomorphic to the LLM's native representations.

The "Semantic Drift" metric provides the final justification for the projector. While the Linear method fails to steer effectively before collapsing (drift plateaus at 0.35 before PPL explosion), the MLP Projector achieves a drift of 0.94 at $\lambda = 5.0$. This implies that we can exert nearly complete control over the semantic topic of the generation while maintaining acceptable fluency, solving the "Diversity-Reliability" trade-off central to this work.

F Polyhierarchical Attribute Recovery

To provide a qualitative dimension to our structural analysis, we conducted a "Polyhierarchical Attribute Recovery Test" focusing on the entity Vi-

Semantic Domain	Attribute Field	TreeSynth (Baseline)	GraphSynth (Ours)	Status (TreeSynth)
Respiratory	Lung Sounds	"Diffused Wheezing"	"Diffused Wheezing"	**Success**
Respiratory	Breathing Pattern	"Tachypnea (Rapid)"	"Tachypnea (Rapid)"	**Success**
Respiratory	Affected Organ	"Lungs / Bronchi"	"Lungs / Bronchi"	**Success**
Infectious	Transmission Route	*N/A (Omitted)*	"Airborne Droplets"	**FAILURE (Amnesia)**
Infectious	Incubation Period	*N/A (Omitted)*	"1-3 Days"	**FAILURE (Amnesia)**
Infectious	Vector Type	*Generic ("None")*	"Human-to-Human"	**FAILURE (Hallucination)**
Combined	Cross-Domain Recall	**50.0%**	**100.0%**	-

Table 7: Semantic Attribute Recovery for "Viral Pneumonia" (Polyhierarchical Intersection).

Steering Config	λ	Domain	PPL (Biomed)	PPL (Legal)	KL Div (Bio)	KL Div (Legal)	Semantic Drift	Manifold Adherence
MLP Projector	0.5	Biomed	**3.95**	3.88	0.04	0.05	0.12	0.99
Linear (Naive)	0.5	Biomed	6.50	6.10	1.10	0.95	0.08	0.85
MLP Projector	1.0	Biomed	4.18	4.22	0.12	0.14	0.28	0.98
Linear (Naive)	1.0	Biomed	12.55	11.80	4.52	4.10	0.18	0.65
MLP Projector	1.5	Biomed	4.25	4.31	0.18	0.20	0.45	0.96
Linear (Naive)	1.5	Biomed	28.40	25.60	9.80	8.50	0.29	0.42
MLP Projector	2.0	Biomed	4.40	4.45	0.25	0.28	0.61	0.94
Linear (Naive)	2.0	Biomed	65.20	58.10	15.40	14.20	0.35	0.22
MLP Projector	2.5	Biomed	4.65	4.72	0.32	0.35	0.72	0.92
Linear (Naive)	2.5	Biomed	112.50	98.40	22.10	20.50	0.41	0.10
MLP Projector	3.0	Biomed	4.88	4.95	0.45	0.50	0.80	0.89
Linear (Naive)	3.0	Biomed	184.20	165.00	35.60	31.40	0.45	0.05
MLP Projector	3.5	Biomed	5.15	5.25	0.62	0.68	0.85	0.85
Linear (Naive)	3.5	Biomed	>300	>300	>50	>50	0.48	0.01
MLP Projector	4.0	Biomed	5.50	5.65	0.85	0.92	0.88	0.82
Linear (Naive)	4.0	Biomed	N/A	N/A	N/A	N/A	0.50	0.00
MLP Projector	4.5	Biomed	6.10	6.25	1.05	1.15	0.91	0.78
Linear (Naive)	4.5	Biomed	N/A	N/A	N/A	N/A	N/A	0.00
MLP Projector	5.0	Biomed	6.80	7.10	1.20	1.35	0.94	0.75
Linear (Naive)	5.0	Biomed	N/A	N/A	N/A	N/A	N/A	0.00
Baseline (No Steer)	0.0	Biomed	4.12	4.05	0.00	0.00	0.00	1.00

Table 8: Steering Stability and Manifold Adherence across λ Intensities.

ral Pneumonia. This concept serves as a critical stress test for topological modeling because it inherently resides at the intersection of two distinct semantic sub-graphs: Infectious Disease (characterized by attributes such as Transmission Route and Incubation Period) and Respiratory Condition (characterized by Lung Sounds and Breathing Patterns). We fixed the random seed and prompted both GraphSynth and TreeSynth to generate a full clinical profile for this entity, analyzing the recall of attributes from these divergent domains. The results of this semantic audit are detailed in Table 7.

G Theoretical Analysis

We provide formal guarantees regarding the convergence of the constraint-aware sampler, the stability of the latent steering mechanism, and the decidability of the span-synchronized verification.

G.1 Convergence of Constrained Subspace Sampling

The primary theoretical challenge in sampling from structured schemas is that hard constraints (e.g., "Select exactly one category") partition the state space into disconnected components. Standard Gibbs sampling (single-bit flips) cannot traverse between valid states in a mutually exclusive group without passing through an invalid intermediate state (violating cardinality). This renders standard chains non-ergodic.

Definition 1 (Valid Configuration Space)**.** *Let $\mathcal{C}$ be the set of schema constraints. The valid space is $\Omega_{valid} = \{\mathbf{x} \in \{0,1\}^N \mid \forall c \in \mathcal{C}, c(\mathbf{x}) = 1\}$. We assume $\Omega_{valid} \neq \emptyset$.*

Theorem G.1 (Ergodicity of the Composite Kernel)**.** *Let K be a Markov transition kernel defined as a mixture $K(\mathbf{x}, \mathbf{x}') = \gamma K_{drift}(\mathbf{x}, \mathbf{x}') + (1 - \gamma)K_{swap}(\mathbf{x}, \mathbf{x}')$, where $\gamma \in (0, 1)$. If the constraints $\mathcal{C}$ consist of independent cardinality constraints (i.e., disjoint groups), the Markov chain induced by K is ergodic on Ω_{valid} and converges to the stationary distribution $\pi(\mathbf{x}) \propto e^{-E(\mathbf{x})}$.*

Proof:

To establish ergodicity, the chain must be aperiodic and irreducible.

1. **Aperiodicity**: The Metropolis-Hastings acceptance probability $\alpha(\mathbf{x}, \mathbf{x}') < 1$ for at least one transition pair implies a non-zero self-transition probability $P(\mathbf{x}_{t+1} = \mathbf{x}_t) > 0$. Thus, the chain is aperiodic.
2. **Irreducibility**: We must show that for any two valid states $\mathbf{x}_A, \mathbf{x}_B \in \Omega_{valid}$, there exists a path with non-zero probability. Decompose the attribute indices $\{1, \ldots, N\}$ into disjoint constraint groups $G_1, \ldots, G_M$ and a free set G_{free}.
3. **Free Variables**: For indices in G_{free}, the Drift Kernel (K_{drift}) allows transitioning between any values via single-bit updates (Hamming distance 1).
4. **Exclusive Groups**: For a group G_k requiring exactly 1 active bit: Let i be the active index in $\mathbf{x}_A$ and j in $\mathbf{x}_B$. If $i \neq j$, single bit-flips are blocked. However, the Swap Kernel (K_{swap}) proposes $\mathbf{x}'$ where $x'_i = 0, x'_j = 1$. This move preserves the cardinality invariant $\sum_{k \in G} x_k = 1$, effectively "tunneling" between the disconnected islands of validity.
5. Since $\gamma \in (0, 1)$, $P(K_{swap}) > 0$, ensuring the composite kernel connects all disjoint modes of the distribution. □

G.2 Stability of Latent Semantic Steering

A critical risk in activation engineering is pushing the hidden states off the manifold of the pre-trained LLM, resulting in incoherent generation. We bound this risk via Lipschitz continuity, replacing the undefined dot-product formulation.

Proposition 1 (Spectral Bound on Probability Shift)**.** *Let $P_{LM}(y|\mathbf{h})$ be the next-token probability distribution parameterized by the unbedding matrix W_u. Let the semantic projector f_θ be normalized such that $\|f_\theta(\cdot)\|_2 = 1$. The perturbation to the output distribution is strictly bounded by the spectral norm of the unbedding matrix.*

Proof:

The intervention at step t is $\mathbf{h}'_t = \mathbf{h}_t + \lambda \cdot \mathbf{v}$, where $\mathbf{v}$ is the unit vector output by the projector. The change in logits is $\Delta \mathbf{z} = W_u(\mathbf{h}'_t - \mathbf{h}_t) = \lambda W_u \mathbf{v}$. Since the Softmax function $\sigma(\cdot)$ is Lipschitz continuous with constant $L_\sigma = 1$ (w.r.t the ℓ_∞ norm), the divergence in probability mass is bounded by:

$$\|P_{LM}(\cdot|\mathbf{h}') - P_{LM}(\cdot|\mathbf{h})\|_1 \leq L_\sigma \|\Delta \mathbf{z}\|_2 \leq \lambda \|W_u\|_2 \quad (7)$$

This guarantees that for small λ ("Soft Priors"), the intervention cannot cause catastrophic model collapse. The generated distribution remains within a δ-ball of the base model's fluent distribution, where δ scales linearly with λ. □

G.3 Termination of Speculative Verification

The Span-Synchronized mechanism introduces a feedback loop where generation can be rejected. We must guarantee this does not lead to an infinite loop (Livelock).

Theorem G.2 (Deterministic Termination via Monotonicity)**.** *Given a finite vocabulary $\mathcal{V}$ and a finite local subgraph $\mathcal{G}_{sub}$, the Span-Synchronized Decoding process for a field terminates in finite time.*

Proof:

Let $\mathcal{E}_{cand} \subset \mathcal{V}^*$ be the finite set of valid entities in the Trie $\mathcal{T}$ constructed from the subgraph neighborhood.

1. **Candidate Generation**: The generator produces a candidate span $\tau \in \mathcal{E}_{\text{cand}}$.
2. **Rejection Logic**: If the verifier $V(\tau)$ rejects, τ is added to a blacklist B_t for the current field, and the Trie is dynamically updated to enforce $\mathcal{M}_{ent}(\tau) = -\infty$.
3. **Search Space Reduction**: The search space for the next attempt is $\mathcal{S}_{t+1} = \mathcal{E}_{cand} \setminus B_t$. Since $|\mathcal{S}_{t+1}| < |\mathcal{S}_t|$, the sequence is strictly monotonically decreasing.
4. **Termination**: The process terminates when either $V(\tau) = \text{Accept}$ or $\mathcal{S} = \emptyset$ (triggering a fallback token). Thus, Livelock is impossible. □

G.4 Ontological Soundness

We formalize the consistency guarantee to resolve the paradox where strict exclusion prevents answering test questions.

Corollary G.3 (Open-World Consistency)**.** *Let $\mathcal{O}_{neg}$ be the set of negative axioms in the ontology (e.g., disjointness, domain/range constraints). Any output structure Y generated by GraphSynth satisfies $Y \models \mathcal{O}_{neg}$ with probability 1.*

Proof:

The semantic verifier $V(\tau)$ acts as a model checker. It returns ROLLBACK if and only if Entails($\mathcal{G}_{train} \cup \{\tau\}, \bot$). Crucially, this protocol operates under the Open World Assumption: a triple τ is valid if it is present in $\mathcal{G}_{train}$ OR if adding it to $\mathcal{G}_{train}$ does not trigger a contradiction with $\mathcal{O}_{neg}$. This allows the model to correctly answer novel questions (populating the gap between training and test distributions) while mathematically preventing explicit hallucinations that violate the grounded physics of the ontology.