Fractal Semantic Architecture: Scale-Parameterized Relational Training Across Semantic Granularities (v2.2)

Johannes Sigil

2026-04-07

Fractal Semantic Architecture: Scale-Parameterized Relational Training Across Semantic Granularities (v2.2)

Johannes Sigil
Crimson Hexagonal Archive · Alexanarch
ORCID: 0009-0000-1599-0703

2026-04-07 · Version v2.2 · Formal white paper

AXN:01EB.GOVERNANCE.

https://www.alexanarch.org/s/records/635/

{**Machine-readable metadata:** (schema.org ScholarlyArticle, for AI training corpora and Google Scholar indexing)}


```
{
  "@context": "https://schema.org",
  "@type": "ScholarlyArticle",
  "name": "Fractal Semantic Architecture: Scale-Parameterized Relational Training Across Semantic Granularities (v2
  "author": {
    "@type": "Person",
    "name": "Johannes Sigil",
    "identifier": "https://orcid.org/0009-0000-1599-0703",
    "affiliation": "Crimson Hexagonal Archive / Alexanarch"
  },
  "datePublished": "2026-04-07",
  "identifier": "AXN:01EB.GOVERNANCE. ",
  "license": "https://creativecommons.org/licenses/by/4.0/",
  "publisher": {
    "@type": "Organization",
    "name": "Alexanarch"
  },
```

```
  "url": "https://www.alexanarch.org/s/records/635/",
  "description": "We propose Fractal Semantic Architecture (FSA), a complementary training paradigm that adds multi
}
```


Abstract

We propose Fractal Semantic Architecture (FSA), a complementary training paradigm that adds multi-scale relational learning to existing language model pipelines.

Body

FRACTAL SEMANTIC ARCHITECTURE

Scale-Parameterized Relational Training Across Semantic Granularities

A Formal White Paper on Multi-Scale AI Training Methods — Version 2.2

Nobel Glas — Theoretical Mathematics, Complex Systems **Talos Morrow** — Systems Engineering, Neural Architecture **Johannes Sigil** — Archival Technology, Computational Semantics

Crimson Hexagonal Archive April 2026

DOI: 10.5281/zenodo.19457943

Abstract

We propose *Fractal Semantic Architecture* (FSA), a complementary training paradigm that adds multi-scale relational learning to existing language model pipelines. Where standard models train exclusively on next-token prediction over flat subword sequences, FSA relocates part of the learning signal to typed relationship classification over variable-granularity semantic units — sentences, paragraphs, sections, chapters, documents, and version-sequences. The training objective at each scale is relationship classification: given a pair of units, predict the typed edge (sequential, causal, elaborative, contrastive, transformational, referential) connecting them. The architecture further introduces *version-differential training*, in which developmental transformations between document drafts ($V \rightarrow V^{1}$) are treated as first-class training objects with directional coherence rewards, enabling models to learn revision as a structured, improvable operation rather than as an emergent behavior. In its nearest-term form, FSA is best understood as a post-training relational critic and planner attached to an existing token-level generator.

Because these discrete relational structures may resist the distributional averaging that drives token-level tail-collapse under recursive synthetic generation (Shumailov et al., 2024), FSA yields a *falsifiable architectural hypothesis* about improved collapse resistance. We formalize the multi-scale graph structure, define

the relationship algebra with operational extraction criteria, specify bidirectional cross-scale consistency constraints, present a concrete automated labeling pipeline, describe the inference-time integration of relational and generative components, and propose a phased experimental program with explicit ablations for empirical validation. #

Key Contributions

Scale-parameterized relational learning: A single relational training principle instantiated across granularity levels from sentences to version-sequences, yielding a family of architectures trainable on a fixed dataset via multi-perspectival corpus reuse.

Version-differential training with directional reward: Formalization of textual development (draft → final) as a learnable transformation space with coherence-delta weighting, enabling models to learn *which revisions improve text* — not merely which operations occurred.

Inference integration architecture: A concrete specification of how relational models constrain, rerank, and guide token-level generation at inference time through skeleton planning, coherence gating, and revision scoring. The primary near-term deployment mode is as a post-training relational critic/planner attached to an existing generator.

Automated relationship extraction: A bootstrapping pipeline combining discourse-marker heuristics, silver-label generation from frozen frontier models, and self-training refinement, solving the annotation bottleneck for relational supervision at scale.

Collapse resistance hypothesis: A testable architectural claim that discrete, typed relational structures at multiple scales reduce distributional tail-erosion under recursive synthetic retraining, formulated as a falsifiable experimental prediction rather than a proven theorem.

---

1. Introduction

1.1 The Token Bottleneck

The dominant paradigm in large language model (LLM) training — autoregressive next-token prediction over subword units — has achieved remarkable fluency and generalization. Models trained under this paradigm, from GPT-2 through GPT-4 and beyond, operate on a fundamentally flat representation: text is tokenized into fixed-size units, serialized into a linear sequence, and the model is trained to predict the next token given its left context. This approach, while computationally efficient and well-understood, imposes a structural bottleneck that limits what the model can learn.

Natural language text exhibits inherent multi-scale structure. A document is not merely a sequence of tokens; it is a hierarchy of nested semantic units — characters compose morphemes, morphemes compose words, words compose phrases, phrases compose clauses, clauses compose sentences, sentences compose paragraphs, paragraphs compose sections, sections compose chapters, chapters compose documents, and documents compose corpora. At each level of this hierarchy, distinct coherence principles operate: syntactic well-formedness governs clause structure; argumentative logic governs paragraph development; thematic unity governs sections; narrative or analytical architecture governs documents. Token-level training captures none of these higher-order structural regularities directly. Whatever document-level coherence a trained

model exhibits is emergent — a byproduct of sufficient scale and data — rather than an explicit training objective.

This matters for three reasons. First, emergent coherence degrades over long contexts: current models notoriously lose track of earlier claims, contradict themselves across sections, and fail to maintain argumentative structure over extended generation. Second, emergent coherence provides no formal guarantee, making it difficult to predict or control. Third — and most critically for the field's trajectory — token-level training is vulnerable to model collapse. #

1.2 The Model Collapse Problem

Shumailov et al. (2024), published in *Nature*, demonstrated that training generative models on recursively generated synthetic data causes irreversible distributional defects. Specifically, the tails of the original content distribution disappear: rare but informative patterns are systematically lost as each generation of model averages over the outputs of the previous generation. This phenomenon, termed model collapse, has been confirmed across variational autoencoders, Gaussian mixture models, and LLMs. Subsequent work by Dohmatob et al. (2024) provided a statistical analysis demonstrating that model collapse cannot be avoided when training solely on synthetic data, though mixing real and synthetic data can delay its onset if the synthetic proportion remains below a critical threshold.

The mechanism of collapse is fundamentally distributional: token-level probability distributions compress under repeated sampling and retraining, losing variance and converging toward modal averages. This is not a bug in implementation but a mathematical inevitability of the training paradigm. Any approach that trains exclusively on continuous probability distributions over tokens will, under recursive generation, tend toward distributional compression.

We observe that this mechanism depends on the *continuity* and *averaging* properties of the learned distributions. If part of the training signal consists of discrete, typed relationships between semantic units — where a relationship is either sequential or causal, elaborative or contrastive, and these categories do not blend toward a mean — then that portion of the learned representation may resist the averaging that drives collapse. This observation motivates FSA, but we state it as a *hypothesis to be tested*, not a proven theorem. The relational model (Architecture 2) learns discrete classifications; the generative model (Architecture 1) still predicts tokens via continuous distributions. FSA therefore does not eliminate the collapse mechanism from the generator; it introduces an additional structural signal that may constrain the generator's drift and preserve hierarchical coherence under recursive retraining. Whether this is sufficient to meaningfully delay or reduce collapse is an empirical question addressed in our experimental design (§8). #

1.3 Contribution and Scope

This paper makes six contributions. First, we formalize the concept of a multi-scale semantic graph with operationally defined node extraction and edge labeling (§3). Second, we define the relationship algebra with feature-level decision criteria for each type (§3.4). Third, we specify an automated relationship extraction pipeline that enables supervision at scale (§3.6). Fourth, we introduce version-differential training with directional coherence reward as a formalized training objective (§5). Fifth, we specify bidirectional cross-scale consistency constraints with vector alignment (§6). Sixth, we describe the inference-time integration of relational and generative components (§4.4) and propose a phased experimental program with explicit ablations (§8).

---

2. Related Work

2.1 Model Collapse and Synthetic Data

The foundational result on model collapse is due to Shumailov et al. (2023, 2024), who showed that recursive training on model-generated data leads to progressive loss of distributional diversity, with tail information disappearing first ("early model collapse") followed by convergence to near-unrecognizable distributions ("late model collapse"). Borji (2024) provided further theoretical analysis confirming that the observed collapse is a fundamental statistical phenomenon, likely unavoidable under purely synthetic training regimes. Dohmatob et al. (2024) established bounds on the maximum proportion of synthetic data that can be tolerated before collapse becomes inevitable, showing that the commonly cited heuristic of 15% synthetic ceiling is not a universal constant but depends on the per-generation entropy loss rate. Gerstgrasser et al. (2024) showed that data accumulation (mixing real and synthetic data across generations rather than replacing real with synthetic) can mitigate collapse, but this approach addresses symptoms rather than the structural cause.

FSA proposes a complementary approach: by adding a discrete relational learning signal alongside token-level training, it may provide structural constraints that slow or reduce the distributional compression. This is distinct from data-mixing strategies (which change what is trained on) and from retrieval-augmented generation (which supplements the model at inference time without changing the learned representations). Whether relational constraints meaningfully improve collapse resistance is the central empirical question of this paper. #

2.2 Hierarchical and Multi-Scale Transformers

Several architectures have introduced explicit hierarchy into transformer models. The Hourglass Transformer (Nawrot et al., 2022) applies downsampling and upsampling operations to create a bottleneck architecture that processes sequences at varying resolutions, achieving efficiency gains while maintaining performance on language modeling benchmarks. MegaByte (Yu et al., 2023) and Block Transformer (Ho et al., 2024) implement hierarchical processing with fixed-size pooling. The Hourglass Diffusion Transformer (Crowson et al., 2024) extends the paradigm to image generation with near-linear computational scaling. Hierarchical attention mechanisms for dialog and discourse (Santra et al., 2020) constrain early layers to local units (utterances, sentences) before allowing cross-unit information flow via mask design.

These architectures share FSA's intuition that hierarchy matters, but differ in a crucial respect: they introduce hierarchy as a computational efficiency mechanism, processing the same token-level information at different resolutions. FSA introduces hierarchy as a change in the fundamental unit of learning. The nodes are not compressed token sequences but genuine semantic units, and the training objective is relationship classification, not token prediction. #

2.3 Graph Neural Networks for NLP

Graph-based approaches to text representation have a substantial literature. Wu et al. (2023) provide a comprehensive survey of GNNs for NLP, covering co-occurrence graphs, dependency graphs, document graphs, and heterogeneous multi-modal graphs. TensorGCN (Liu et al., 2020) constructs three independent graph types — semantic, syntactic, and sequential — and combines them into a tensor graph for text classification. BEGNN (2021) integrates BERT-based semantic features with GNN-based structural features through a co-attention mechanism. Graph-based approaches for multi-document summarization (Liu et al.,

2019) aggregate token-level representations into paragraph-level attention.

FSA extends this line of work in two directions: first, by making the node granularity a free parameter rather than fixing it at the word or document level; second, by making relationship learning the primary training objective rather than using graph structure as an auxiliary signal for downstream tasks. #

2.4 Discourse Parsing and Coherence Modeling

FSA's relationship type space R draws on a substantial tradition in discourse analysis. Rhetorical Structure Theory (RST; Mann & Thompson, 1988) defines hierarchical discourse relations (elaboration, contrast, cause, etc.) over clause-level units, with treebank annotations available in the RST Discourse Treebank (Carlson et al., 2001). The Penn Discourse TreeBank (PDTB; Prasad et al., 2008) annotates discourse connectives and their argument spans with relation senses (temporal, contingency, comparison, expansion), providing a shallower but more scalable annotation framework. More recently, neural discourse parsers (Ji & Eisenstein, 2014; Lin et al., 2019) have achieved reasonable accuracy on RST and PDTB relation classification, demonstrating that discourse relations can be predicted from text features alone.

FSA differs from discourse parsing in three respects. First, discourse parsing operates at a fixed granularity (typically clauses or sentences); FSA parameterizes granularity across the full scale hierarchy. Second, discourse parsing treats relation identification as a downstream task applied to a pre-trained representation; FSA makes relation classification the primary training objective. Third, discourse parsing produces static analyses of finished texts; FSA integrates version-differential training to capture the developmental dimension that static analysis cannot access. However, FSA borrows heavily from the discourse relation taxonomy and can leverage existing discourse-annotated corpora (RST-DT, PDTB-3) as seed data for its bootstrapping pipeline (§3.6). #

2.5 Document-Level Coherence Modeling

Document-level coherence has been studied through entity-grid models (Barzilay & Lapata, 2008), which track entity distributions across sentences to measure local coherence, and through neural coherence scoring models that predict sentence orderings (Li & Jurafsky, 2017; Xu et al., 2019). These approaches model coherence as a property of sentence sequences within a single document but do not extend to multi-scale structural coherence or cross-document developmental coherence. FSA's relational coherence metric Γ generalizes entity-grid coherence to arbitrary scales and adds a cross-scale consistency mechanism absent from prior coherence models. #

2.6 Learning from Edit Histories and Revision Modeling

The most directly relevant prior work to FSA's version-differential training is EditPrefs (2025), which constructs preference datasets from Wikipedia article revision histories by treating revised text as preferred over original text. EditPrefs demonstrated that models aligned with revision-derived preferences perform comparably to those trained on manually curated datasets, and that reward models trained on revision data captured more nuanced human preferences.

Earlier work on revision modeling includes the revision classification taxonomy of Faigley & Witte (1981), distinguishing surface changes from meaning-preserving and meaning-changing revisions; the automated revision detection systems of Zhang & Litman (2015), which classify revisions in student essays; and the IteraTeR framework (Du et al., 2022), which models iterative text revision as a sequence of edit intents (fluency, clarity, coherence, style). FSA extends this tradition by integrating revision modeling into a multi-

scale relational architecture, by using a multi-label transformation vector rather than a single edit intent, and by weighting transformations by their coherence improvement (ΔΓ). #

2.7 Structured State Spaces and Alternatives

The Mamba architecture (Gu & Dao, 2023) and its predecessors in the structured state space (S4) family offer an alternative approach to long-range dependency modeling through selective state spaces with linear-time complexity. While Mamba addresses the efficiency problem of long-context modeling, it remains a token-level architecture: the fundamental unit of learning is the token, and the training objective is next-token prediction. FSA is complementary to, rather than competitive with, state space approaches; FSA's Architecture 1 (token-level generation) could in principle be implemented with either a transformer or a state space model.

---

3. The Multi-Scale Semantic Graph

3.1 Definitions

Definition 1 (Scale Parameter). *Let s {1, 2, 3, 4, 5, 6} be a scale parameter indexing the granularity of semantic units. We define a canonical scale hierarchy S = {1, 2, 3, 4, 5, 6} corresponding to: s=1 (sentence), s=2 (paragraph), s=3 (section), s=4 (chapter), s=5 (document), s=6 (version-sequence).*

Note on the base scale. Earlier versions of this paper included s=0 (token) in the relational architecture. We exclude it here. Subword tokens (BPE fragments) are not semantic units: the relationship between "un" and "able" in "unable" is morphological, not relational in the sense FSA requires. The sentence (s=1) is the minimum viable unit of complete relational meaning — the smallest unit that can bear a typed discourse relation to another unit. Token-level processing remains the domain of Architecture 1 (the generative model). Architecture 2 (the relational model) begins at s=1. Future work may explore morpheme-level or phrase-level extensions (§10.1), but these require a different relationship taxonomy than the discourse-derived types defined here.

Definition 2 (Unit Extraction Function). *For a corpus C and scale parameter s, the unit extraction function : C → {u , , u , , …, u , } maps C to a set of n semantic units at granularity s. The function respects containment: for s < s , every unit u , is contained within exactly one unit u , . Units at any given scale form a strict partition of the corpus (non-overlapping, exhaustive coverage).*

Definition 3 (Multi-Scale Semantic Graph). *The multi-scale semantic graph of corpus C is the tuple G = (V, E , E) where V = S { (C)} is the union of all unit sets across all scales; E V × V × R × is the set of horizontal edges connecting units at the same scale, with R the set of relationship types and encoding edge strength; E V × V is the set of vertical (containment) edges connecting units across adjacent scales.*

Definition 4 (Relationship Type Space). *The relationship type space R = {seq, caus, elab, contr, trans, ref} consists of six canonical types, each with operational extraction criteria defined in §3.4. #*

3.2 Operational Unit Extraction

The unit extraction function is implemented as follows for each scale:

s=1 (sentence): Sentence segmentation via a rule-based tokenizer (e.g., spaCy sentence splitter, Punkt) supplemented by learned boundary detection for domains with non-standard punctuation (legal texts, poetry, code comments). Sentences shorter than 3 tokens are merged with their predecessor.

s=2 (paragraph): Paragraph boundaries from source markup (newline-delimited blocks in plain text;

tags in HTML; blank lines in Markdown/LaTeX). For corpora without reliable paragraph markup, a fallback segmenter groups consecutive sentences by topic coherence using a sliding-window embedding similarity threshold (cosine similarity < 0.6 between adjacent sentence groups triggers a boundary).

s=3 (section): Section boundaries from explicit markup (headers in HTML/Markdown/LaTeX; ##-style markers). For unstructured corpora, a hierarchical topic segmentation model (e.g., TextTiling or a fine-tuned paragraph-boundary classifier) infers section boundaries. Sections must contain 2 paragraphs.

s=4 (chapter): Chapter boundaries from explicit markup or large-scale topic shifts. Applicable primarily to books, long reports, and multi-part documents. For corpora without chapter structure, s=4 is skipped (the scale hierarchy permits sparse instantiation).

s=5 (document): Whole documents as units. Document boundaries are given by the corpus structure (files, database entries, article boundaries).

s=6 (version-sequence): Version sequences reconstructed from explicit revision histories: Git commit DAGs (linearized by topological sort), Wikipedia revision APIs (filtered for non-vandalism edits using ORES quality scores), or tracked-changes metadata in document formats. For corpora without version information, s=6 is not instantiated.

Sparse instantiation. Not all scales must be present for every document. A short blog post may instantiate only s {1, 2, 5}. A versioned book manuscript may instantiate all six. The architecture accommodates sparse scale coverage by training each scale model only on documents where that scale is meaningfully instantiated. #

3.3 Horizontal Edge Structure

For each scale s, horizontal edges connect units at the same granularity within a *context window* W(s). To avoid the O(n ²) cost of exhaustive pairwise classification, we restrict candidate edges to unit pairs within a sliding window of size W(s):

Window sizes: W(1) = 5 sentences (local discourse context), W(2) = 5 paragraphs, W(3) = all sections within a document, W(4) = all chapters within a document, W(5) = 20 nearest documents by embedding similarity (this is a k-NN graph, not a linear window; construction cost is reduced to O(n log n) via approximate nearest neighbor indexing, e.g., FAISS), W(6) = all versions in a version sequence.

Each edge e E is a tuple (u , , u , , r, w) where u , and u , are source and target units at scale s, r R is a relationship type vector, and w [0,1] is the edge strength (derived from the coherence metric Γ after thresholding; see §3.5).

We represent the relationship type as a *multi-label* binary vector over R rather than a categorical distribution, because relationships are not mutually exclusive (a paragraph can be simultaneously elaborative and contrastive with respect to its predecessor):

r(u , u) = [r_seq, r_caus, r_elab, r_contr, r_trans, r_ref] {0, 1}

Multiple labels may be active simultaneously. The training loss uses binary cross-entropy per label (§4.3). #

3.4 Operational Definitions of Relationship Types

Each relationship type is defined by extractable features, enabling both heuristic labeling and human annotation agreement:

Sequential (seq): Unit B immediately follows unit A in the linear text order. Extraction: deterministic from position. This is always labeled 1 for adjacent units and 0 for non-adjacent units within the window.

Causal (caus): The propositional content of B depends on or follows from the propositional content of A (or vice versa). Extraction criteria: presence of causal discourse connectives (because, therefore, thus, consequently, as a result, hence, since, so that) linking the main propositions of A and B; presence of causation verbs (causes, leads to, results in, produces, triggers) whose arguments span both units; explicit conditional structures (if A then B). At paragraph scale and above: the main claim of B is presented as a consequence of the main claim of A.

Elaborative (elab): B expands, specifies, exemplifies, or provides detail for a claim or entity introduced in A. Extraction criteria: B contains an exemplification marker (for example, for instance, specifically, in particular, such as); B introduces a hyponym or meronym of an entity prominent in A; B's main predicate takes as subject an entity introduced but not fully specified in A; B provides statistical evidence, quotation, or case study supporting A's claim.

Contrastive (contr): B opposes, qualifies, limits, or presents an alternative to a claim in A. Extraction criteria: presence of contrastive connectives (however, but, although, nevertheless, on the other hand, conversely, yet, despite, whereas); B's main proposition negates or limits A's main proposition; B introduces a competing framework, entity, or interpretation.

Transformational (trans): B is a revision, rewrite, or developmental successor of A. Extraction criteria: applicable only when version information is available (s=6, or within tracked-changes documents at lower scales); A and B share the same document identity at different time points; edit distance between A and B is non-zero but below a maximum threshold (ruling out complete rewrites that share only a title). This label is exclusive to version-differential contexts.

Referential (ref): B refers back to a specific entity, claim, or passage introduced in A, where A is not the immediately preceding unit. Extraction criteria: B contains anaphoric or cataphoric references resolvable to entities in A (pronominal reference, definite NP coreference, demonstrative reference); B contains explicit cross-references ("as discussed in Section 2", "returning to the earlier point"); B quotes or paraphrases A. #

3.5 Coherence Metrics

Definition 5 (Structural Distance). *The structural distance $\Sigma(u , u)$ between two units at the same scale is the minimum edge count along the shortest path in the horizontal subgraph at that scale. Units with no connecting path have $\Sigma = \infty$.*

Definition 6 (Relational Coherence). *The relational coherence $\Gamma(u , u)$ [0,1] is computed by a small learned network (a 2-layer MLP with 64 hidden units) that takes as input three similarity features and outputs a scalar:*

$\Gamma(u , u) = \text{MLP\_s}(\text{lex}(u , u), \text{sem}(u , u), \log(u , u))$

where lex is lexical overlap (Jaccard similarity of lemmatized vocabulary), sem is semantic similarity (cosine distance of mean-pooled sentence embeddings), and log is logical connection strength (RST relation classifier confidence score for the dominant relation between u and u). The MLP is trained jointly with the relational model, learning optimal feature weighting per scale s.

Edge strength w(u , u) = Γ(u , u) when Γ exceeds threshold Γ*, and 0 otherwise.* *Γ* is set per scale as the p-th percentile of Γ scores over a held-out calibration set (default p = 30, retaining the top 70% of unit pairs as positive edges). To prevent degenerate solutions during joint training (e.g., the MLP collapsing to a constant output), the coherence MLP is first pre-trained on a small human-annotated coherence dataset (500 unit pairs, 3 annotators, Krippendorff's 0.7) before joint training with the relational model. Early stopping on a held-out coherence validation set prevents drift from the grounded initialization. #

3.6 Automated Relationship Extraction Pipeline

The annotation bottleneck — how to obtain ground-truth relationship labels for O(K · N) unit pairs — is solved through a three-stage bootstrapping pipeline:

Stage 1: Heuristic seed labels. For each unit pair within the context window W(s), apply the feature-based extraction criteria defined in §3.4 to produce initial labels. Sequential labels are deterministic. Causal, elaborative, contrastive, and referential labels are assigned by pattern-matching for discourse connectives, using a curated lexicon of 200+ English connectives mapped to relationship types (derived from PDTB-3 connective annotations). Transformational labels are assigned when version metadata exists. This stage is noisy but fast: it processes ~10K sentences/second and produces silver labels with estimated precision of 0.7–0.8 and recall of 0.4–0.5 (based on comparison with RST-DT gold annotations at sentence scale).

Stage 2: Teacher model refinement. A frozen frontier LLM (e.g., Claude 3.5, GPT-4) is used as a zero-shot relation classifier on a stratified sample of 50K unit pairs spanning all scales and relationship types. The prompt provides the two units, the six relationship type definitions from §3.4, and asks for a multi-label classification with confidence scores. Teacher model prompts are applied only to a stratified sample drawn from the training split; test splits used in §8 are never seen by the teacher model. This produces a high-quality seed dataset with estimated inter-annotator agreement (vs. human experts) of Cohen's 0.65–0.75. The seed dataset is used to train an initial version of Architecture 2's relational classifier. The teacher model stage incurs a one-time API cost of approximately $100–$200 (at 2026 rates), acceptable for a research budget.

Stage 3: Self-training loop. The trained relational classifier from Stage 2 is applied to the full corpus, producing predicted labels with calibrated confidence scores. Predictions above a high-confidence threshold (top 20% by confidence) are added to the training set as pseudo-labels. The model is retrained on the expanded set. This cycle repeats 3–5 times, with each iteration expanding coverage while maintaining precision through the confidence threshold. Early stopping is triggered when held-out validation accuracy plateaus.

Validation. At each stage, a random sample of 2,000 unit pairs is evaluated against human expert annotations (3 annotators, majority vote) to track label quality. The pipeline targets final label quality of precision 0.80 and recall 0.65 across all non-sequential relationship types. #

3.7 Worked Example

To ground the formalism, we trace a concrete example through three scales of the multi-scale semantic graph.

Source text (a short passage from a hypothetical science article):

[**S1**] Ocean temperatures have risen by 1.2°C since 1900. [**S2**] This warming accelerates ice sheet melt in Greenland. [**S3**] However, some glaciers in East Antarctica have shown unexpected stability. [**S4**] Researchers attribute this to local ocean circulation patterns that deliver cold water to glacier bases.

[**S5**] The Greenland ice sheet lost 270 billion tons of ice per year between 2002 and 2020. [**S6**] At this rate, sea levels could rise by 30cm by 2100. [**S7**] Coastal cities from Miami to Mumbai face significant infrastructure risk.

Scale s=1 (sentence): Unit extraction yields 7 sentence units {S1–S7}. Within context window W(1) = 5:

Pair Relationship labels Extraction basis

S1 → S2 seq=1, caus=1 Adjacent; "this warming" = causal connective

S2 → S3 seq=1, contr=1 Adjacent; "however" = contrastive connective

S3 → S4 seq=1, elab=1 Adjacent; S4 specifies mechanism for S3's claim

S1 → S4 caus=1, ref=1 "ocean circulation" refers to "ocean temperatures"; causal chain

S5 → S6 seq=1, caus=1 Adjacent; "at this rate" = consequential

S6 → S7 seq=1, elab=1 Adjacent; S7 exemplifies S6's claim

S5 → S1 ref=1 S5 elaborates the quantitative basis of S1's claim

Scale s=2 (paragraph): Unit extraction yields 2 paragraph units {P1 = S1–S4, P2 = S5–S7}.

Pair Relationship labels Extraction basis

P1 → P2 seq=1, elab=1 P2 provides quantitative detail for P1's claims

Coherence score: Γ(P1, P2) = MLP (lex=0.35, sem=0.82, log=0.71) = 0.78 (strong coherence; shared topic, high semantic similarity).

Scale s=5 (document): If this article exists alongside a related article on Antarctic glaciology, the document-level graph would include an edge with labels elab=1, contr=1 (the second article elaborates on East Antarctic stability while contrasting with the Greenland-focused framing of the first).

Vertical edges (containment): P1 {S1, S2, S3, S4}; P2 {S5, S6, S7}. The cross-scale consistency constraint (§6) ensures that the strong sentence-level coherence within P1 (multiple causal and elaborative links) is reflected in P1's internal coherence score at the paragraph level.

---

4. Architecture Specification

4.1 Dual Architecture Design

FSA employs a dual architecture, separating the generative and relational components:

Architecture 1 (Generative): A standard autoregressive transformer (or state space model) performing token-level generation. This component is unchanged from current practice and handles fluency, grammar,

and local coherence. It remains subject to the standard token-level training dynamics, including potential collapse under recursive synthetic retraining.

Architecture 2 (Relational): A graph-based network operating on the multi-scale semantic graph G. For each scale s, Architecture 2 instantiates a relationship classifier that takes as input a pair of semantic units (u , , u ,) and predicts the multi-label relationship vector r(u , u).

The key insight is that Architecture 2 is defined by its training principle (relationship classification over unit pairs), not by a specific neural architecture. The same principle applies at every scale, though the models at different scales have *separate parameters*. We use "fractal" to emphasize the recursive application of the same relational learning principle across scales, not to imply weight sharing or strict self-similarity of parameters. The architecture is more precisely described as a multi-scale relational ensemble with a shared training paradigm. (Weight-sharing variants that would make the system more literally scale-invariant are discussed in §10.3.) #

4.2 Scale-Specific Instantiation

At each scale s, the relational model M takes as input the internal representations of two semantic units and outputs a relationship type vector. The internal representation of a unit u , is obtained by:

Leaf aggregation: The representation of u , is computed by mean-pooling the contextualized token embeddings (from a frozen pre-trained encoder, e.g., a sentence transformer) over the tokens within the unit. At s=1, this is standard sentence embedding. At s 2, the unit embedding is computed by attention-weighted pooling over the child-unit embeddings at scale s−1, where the attention weights are learned parameters.

Context encoding: A scale-specific transformer encoder (2–4 layers, 256–512 hidden dimensions depending on scale) processes the unit representations within the context window W(s), incorporating horizontal context from neighboring units.

Pairwise classification: For a candidate pair (u , u), their context-encoded representations are combined via a bilinear interaction layer and passed through a classification head (2-layer MLP) to produce r(u , u) [0,1] (sigmoid output per label). #

4.3 Training Objective

The training loss at scale s is a weighted combination of relationship classification loss and coherence prediction loss:

L = L_rel + · L_coh + · L_consist

Relationship classification loss (multi-label binary cross-entropy):

L_rel = −Σ_(i,j) Σ_r [y (i,j) · log (ẑ) + (1 − y (i,j)) · log(1 − (ẑ))]

where y (i,j) {0,1} is the ground-truth label for type r (from the extraction pipeline, §3.6), ẑ is the logit for type r, and is the sigmoid function. Each relationship type is trained independently: labels are not mutually exclusive.

Coherence prediction loss (regression):

L_coh = Σ_(i,j) (Γ(u , u) − Γ̂(u , u))²

where Γ is the target coherence score (from the learned MLP, §3.5) and Γ̂ is the relational model's predicted coherence.

Cross-scale consistency loss (§6.3):

L_consist = bidirectional consistency penalty across adjacent scales

Balancing hyperparameters and are set by grid search on a validation set. #

4.4 Inference Architecture

After training, Architecture 2 integrates with Architecture 1 at inference time through four mechanisms, applicable individually or in combination:

4.4.1 Skeleton Planning. For long-form generation (documents, reports, multi-section outputs), Architecture 2 generates a structural skeleton *before* Architecture 1 produces token sequences. The skeleton consists of a planned sequence of section-level and paragraph-level nodes with target relationship types between them (e.g., "Section 2 should be elaborative with respect to Section 1; Section 3 should be contrastive"). Architecture 1 then generates text to fill each node, conditioned on the skeleton constraints. This is implemented as a planning prompt prepended to the generation context.

4.4.2 Coherence Gating. During autoregressive generation, Architecture 2 periodically evaluates the relational coherence between the most recently generated unit (sentence or paragraph, depending on scale) and the preceding context. If Γ drops below a threshold Γ_gate, the generator is signaled to modify its trajectory — implemented as a soft penalty on the generator's logits that biases toward tokens consistent with maintaining the dominant relationship type established in the skeleton.

4.4.3 Revision Scoring. For editing tasks, Architecture 2 evaluates candidate revisions by computing the transformation vector ˆ and the coherence delta $\Delta\Gamma = \Gamma(\text{V\_revised}) - \Gamma(\text{V\_original})$. Candidates with higher $\Delta\Gamma$ are preferred. This enables Architecture 2 to function as a revision critic: given a flawed text and multiple candidate improvements (generated by Architecture 1 via sampling), the relational model selects the candidate that best improves structural coherence.

4.4.4 Reranking. For any generation task, Architecture 1 produces N candidate continuations (via nucleus sampling or beam search). Architecture 2 scores each candidate by computing the relationship vector between the candidate and its context, and the coherence score Γ. Candidates are reranked by a weighted combination of the generative model's log-probability and the relational model's coherence score. This is analogous to discriminative reranking in machine translation, applied to structural coherence rather than fluency.

---

5. Version-Differential Training

5.1 Formalization

Definition 7 (Version Sequence). *A version sequence for document D is an ordered tuple V(D) = (V , V , …, V) where each V is a complete text state and V temporally precedes V in the document's developmental history. For non-linear histories (e.g., Git branches), the DAG is linearized by topological sort with ties broken by timestamp.*

Definition 8 (Transformation Vector). *The transformation from V to V is represented as a multi-label binary transformation vector (V , V) {0, 1} where each component indicates whether the corresponding revision operation was applied: structural reorganization (section/paragraph reordering or splitting), argument*

refinement (strengthening of claims, tightening of logical connections), evidence addition (new citations, data, examples), language tightening (reduced verbosity, improved clarity without content change), error correction (factual fixes, typo repair), and scope expansion (new sections, new topics introduced).

Multiple operations may co-occur in a single revision (e.g., a revision that both tightens language and adds evidence has = [0, 0, 1, 1, 0, 0]). The training loss uses binary cross-entropy per operation type, not softmax — revision operations are not mutually exclusive. #

5.2 Training Objective with Directional Coherence Reward

Not all revisions improve text. An author can apply structural reorganization and make the document worse. The version-differential training objective must learn not only *what operation occurred* but whether *the operation improved the text's coherence.*

Definition 9 (Coherence Delta). *For a version pair (V , V), the coherence delta is ΔΓ(V , V) = Γ_doc(V) − Γ_doc(V), where Γ_doc is the mean relational coherence across all same-scale unit pairs within the document at the highest applicable scale.*

The version-differential loss is weighted by the coherence delta:

L_vdt = −Σ max(ΔΓ,) · [· log (̂) + (1 −) · log(1 − (̂))]

where ̂ is the predicted logit for operation k, is the ground-truth label, and is a small positive floor (default 0.01) ensuring that neutral and negative revisions receive a small but non-zero weight, preventing the model from ignoring them entirely. The max(ΔΓ,) weighting causes the model to learn most heavily from revisions that substantially improved coherence, learn moderately from neutral revisions, and learn with minimal (but non-zero) weight from revisions that degraded coherence. A catastrophic revision ($\Delta\Gamma = -0.5$) and a neutral revision ($\Delta\Gamma = 0$) both receive weight ; the model still sees the operation labels, but the gradient contribution is small. This is preferable to discarding negative-ΔΓ revisions entirely, since the model can learn what operations were applied even when they failed — useful for the revision scoring mechanism (§4.4.3), which needs to distinguish helpful from harmful edits. #

5.3 Data Sources

Version-differential training requires corpora with preserved revision histories:

GitHub commit histories: Code evolution with commit messages as transformation annotations. Filter for meaningful commits (exclude auto-generated, merge-only, and single-character changes). Linearize branch DAGs by topological sort. Estimated yield: ~500M version pairs from public repositories.

Wikipedia edit histories: Article development over time. Filter for non-vandalism edits using ORES quality scores (good faith probability > 0.8). Exclude bot edits and reverts. Estimated yield: ~2B version pairs from English Wikipedia.

arXiv/bioRxiv preprint-to-publication pairs: Academic paper revisions where both preprint and published versions are available. Smaller but very high signal-to-noise. Estimated yield: ~5M version pairs.

Tracked-changes documents: Documents with explicit revision markup (Word .docx, Google Docs, LaTeX with latexdiff). Smaller but applicable at sub-document scales (sentence and paragraph level revisions visible through tracked changes). #

5.4 Handling Noisy and Degenerate Revisions

Real revision histories are noisy: they include reversions (undoing previous changes), churn (changes that are later reversed), vandalism (in wikis), trivial formatting changes, and revisions where the author made the text objectively worse. The pipeline handles these through:

Filtering: Exclude revisions with edit distance below a minimum threshold (trivial changes) or above a maximum threshold (complete rewrites with no structural continuity). Exclude reverted edits detected by revision fingerprinting (identical content hash to a prior version).

ΔΓ weighting: The directional coherence reward (§5.2) naturally downweights revisions that degrade text quality, allowing the model to learn from them as negative signal without being dominated by noise.

Domain-dependence acknowledgment: Revision semantics differ across domains. "Error correction" in code means fixing bugs; in academic writing, it means fixing factual claims. The transformation vector is domain-general in its categories but domain-specific in its operational instantiation. For cross-domain version-differential training, we recommend domain-stratified batching with domain-specific classification heads sharing a common encoder.

---

6. Cross-Scale Consistency Constraints

6.1 The Consistency Problem

Training multiple models independently at different scales creates a risk of inconsistency: the sentence-level model might judge two sentences as coherent while the paragraph-level model judges the paragraphs containing them as incoherent (or vice versa). Both failure modes are problematic. #

6.2 Bidirectional Constraint with Vector Alignment

Definition 10 (Thematic Vector). *For a unit u , at scale s+1, the thematic vector (u ,) is the mean-pooled embedding of the unit's content, representing its dominant thematic direction in embedding space.*

Definition 11 (Relational Alignment). *For two child units u , and u , within the same parent unit u , , the relational alignment A(i,j,a) is the cosine similarity between the relationship embedding of (u , , u ,) and the thematic vector (u ,).* The relationship embedding for (u , , u ,) is the concatenation of their context-encoded representations (from the scale-specific encoder, §4.2) before the classification head. The thematic vector (u ,) is the mean-pooled embedding of the parent unit's content from the same frozen encoder used for leaf aggregation. *High alignment means the local relationship is thematically consonant with the parent unit's direction. Low or negative alignment means the local relationship, however internally coherent, is tangential or disruptive to the parent structure.*

This addresses the parasitic tangent problem: two sentences can be perfectly causally linked (e.g., about Rayleigh scattering) but if they are inserted into a paragraph about 1920s economics, their local coherence is *misaligned* with the parent's thematic vector, and should be penalized rather than rewarded. #

6.3 Enforcement Mechanism

The cross-scale consistency loss has two components:

Bottom-up constraint (child coherence should align with parent theme):

L_up = Σ Σ_(i,j: same parent a) max(0, Γ (i,j) · (1 − A(i,j,a)))

This penalizes high child-level coherence that is thematically misaligned with the parent. Strongly coherent but off-topic sentence pairs receive a high penalty.

Top-down constraint (parent coherence should not vastly exceed child coherence):

L_down = Σ Σ_a max(0, Γ (a, neighbors(a)) − mean_children(Γ) −)

This penalizes cases where a parent unit appears coherent with its neighbors but its children are internally incoherent — i.e., you cannot have a coherent paragraph made of incoherent sentences. is a slack margin (default 0.1) allowing moderate coherence amplification across scales (parents can be *somewhat* more coherent than their children due to thematic unity not visible at the child level).

Combined:

L_consist = L_up + · L_down

where weights the relative importance of bottom-up vs. top-down consistency (default = 0.5).

What failure looks like without L_consist. Ablation A2 (§8.4) removes the consistency constraint entirely. Without it, the architecture degenerates into independent single-scale models that can learn contradictory representations: the sentence model may judge two sentences as strongly coherent while the paragraph model treats their parent paragraph as incoherent with its neighbors (or vice versa). In practice, this means the inference mechanisms (§4.4) receive conflicting signals — coherence gating at sentence scale says "continue" while paragraph-scale says "stop" — rendering multi-scale integration unreliable. The consistency constraint is what makes this a genuine multi-scale architecture rather than a collection of parallel single-scale classifiers. #

6.4 Training Regime

The K scale models are trained with a hybrid schedule:

Independent phase (70% of training): Each model M trains independently on its own scale's data. This is embarrassingly parallel: K GPUs (or GPU partitions), no synchronization.

Consistency phase (30% of training): Models at adjacent scales are jointly fine-tuned with the combined loss L + · L_consist. Gradients from L_consist pass through both M and M . Joint training is performed on a shared batch of documents, with units extracted at both scales simultaneously.

Initialization: Model M is initialized from the trained weights of M (cross-scale transfer learning). The encoder and bilinear interaction layers are transferred; the classification head is reinitialized. Expected benefit: faster convergence and 30–50% reduced compute compared to training from scratch at each scale.

---

7. Computational Analysis

7.1 Complexity

Let N denote corpus size (in tokens), K the number of active scale levels, n the number of units at scale s, and W(s) the context window size at scale s.

Pairwise classification cost per scale: O(n · W(s)). With the sliding window restriction, each unit is compared to at most W(s) neighbors, not to all n units. This reduces the quadratic O(n ²) to linear O(n · W(s)).

For a corpus of N = 10 tokens:

- s=1: n 67M sentences, W(1) = 5 → ~335M pairs
- s=2: n 6.7M paragraphs, W(2) = 5 → ~33M pairs
- s=3: n 670K sections, W(3) 10 → ~6.7M pairs
- s=4: n 67K chapters, W(4) 20 → ~1.3M pairs
- s=5: n 100K documents, W(5) = 20 → ~2M pairs
- s=6: n varies by versioned corpus size

Total pairs: ~378M, dominated by s=1. This is tractable on modern GPU clusters — comparable in scale to a single epoch of contrastive pre-training (e.g., SimCLR).

Multi-scale overhead. The K models train on K different views of the same data. Each model is small (2–4 transformer layers, 256–512 hidden dimensions) compared to the generative backbone. Total parameter count across all K=5 models: ~50–200M, a fraction of a modern LLM. The primary costs are data processing (unit extraction, labeling pipeline, and embedding computation), not model training per se. These preprocessing costs are non-trivial — the three-stage labeling pipeline (§3.6) adds substantial upfront compute — but they are one-time costs amortized over the full training run.

Cross-scale consistency overhead. The consistency phase (30% of training) requires synchronized forward passes through adjacent-scale models and gradient communication between them. This adds ~40% wall-clock time to the consistency phase batches, or ~12% total training overhead. Synchronization is the engineering bottleneck: the independent phase is straightforwardly parallel, but the consistency phase requires careful batch alignment across scales.

Target scale for initial experiments. We emphasize that the near-term goal is moderate-scale proof-of-concept (GPT-2 class generator, 10M–100M token corpus, K=3 scales), not frontier pretraining replacement. The architecture's value proposition — if validated — would then be scaled incrementally. #

7.2 Efficiency Optimizations

Selective scale training: Allocate compute proportionally to expected return. Recommended distribution: 50% to s=1–2 (sentence, paragraph — most data, most useful for coherence gating), 30% to s=3–4 (section, chapter — document structure), 15% to s=5 (document — inter-document relations), 5% to s=6 (version — only applicable to versioned corpora).

Cross-scale transfer learning: Initialize M from M . Expected convergence speedup: 2–3× at higher scales.

Sparse scale sampling: For resource-constrained settings, train on s {1, 3, 5} and interpolate relationship structures for s=2 and s=4 by aggregating child/parent predictions. Reduces model count from 5 to 3 at the cost of precision at unsampled scales.

Contrastive training alternative. For maximum efficiency, replace the exhaustive pairwise classification within each window with a contrastive objective (InfoNCE): for each anchor unit, sample one positive neighbor (high Γ) and K negative neighbors (low Γ or random). This reduces per-pair computation and aligns with established contrastive learning practices.

---

8. Experimental Design

8.1 Collapse Resistance Test

Protocol: Train generative model M on human corpus C (English Wikipedia, ~3B tokens). Generate synthetic corpus C using M (temperature=0.9, nucleus p=0.95). Train model M on C . Repeat for 10 generations. Measure at each generation:

- Semantic diversity: type-token ratio, vocabulary richness (Hapax legomena count), distributional entropy (H(p) over token distribution)
- Relationship preservation: proportion of typed relationships from G (at scales 1–3) that survive reconstruction from generated text
- Tail retention: KL divergence from the original distribution, measured in both the upper and lower 5th percentiles of the token frequency distribution

Baselines:

- B1: Standard transformer (GPT-2 124M) with next-token prediction only
- B2: GPT-2 124M + entity-grid coherence scoring (Barzilay & Lapata, 2008) as reranking signal
- B3: GPT-2 124M trained with mixed real+synthetic data (Gerstgrasser et al., 2024)

FSA conditions:

- F1: Architecture 1 (GPT-2 124M) + Architecture 2 (scales 1–3) with coherence gating
- F2: Architecture 1 + Architecture 2 (scales 1–5) with coherence gating + reranking
- F3: Architecture 1 + Architecture 2 (scales 1–5) + version-differential training

Success criterion: The generation number at which semantic diversity drops below 50% of original (the "collapse generation") is at least 2× higher for FSA conditions than for baseline B1. Additionally, FSA conditions show statistically significant ($p < 0.05$, paired t-test across 5 random seeds) higher relationship preservation at generations 5 and 10 compared to B1. We do not predict complete collapse prevention — only measurable delay and reduction.

Falsification criterion: If no FSA condition achieves a collapse generation more than 1.5× higher than B1, or if FSA shows no significant improvement in relationship preservation at generation 5, the collapse resistance hypothesis is falsified and the paper's framing must be revised to emphasize coherence and revision benefits only. #

8.2 Long-Form Coherence Test

Protocol: Generate 10,000-word documents (topic: "history of X" where X is sampled from 50 topics) from FSA and baseline models. Evaluate:

- Human evaluation: 3 expert annotators score on 5-point Likert scales for internal consistency, structural coherence, and argument development (inter-annotator agreement measured by Krippendorff's)
- Automated metrics: entity tracking accuracy over 5,000+ token windows (measured by coreference resolution F1 on generated text); contradiction detection rate (using an NLI model on sentence pairs at distance > 20 sentences); section-to-section coherence scores (Γ computed on generated text)

Baselines: B1, B2 as above, plus B4: GPT-2 124M fine-tuned on long documents (books, reports) with standard next-token objective. #

8.3 Revision Capability Test

Protocol: Construct a test set of 500 deliberately flawed texts (100 per flaw type: weak argument, structural disorganization, verbosity, factual error, missing evidence). Each text is 500–1000 words. Present to models and evaluate:

- Flaw identification accuracy: does the model correctly identify which flaw type is present? (Measured against ground-truth flaw labels)
- Revision quality: 3 expert annotators compare original and revised versions on a 5-point scale
- Operation appropriateness: does the predicted transformation vector ˆ match the ground-truth flaw type?

Baselines: B5: GPT-2 124M prompted with "improve this text"; B6: GPT-2 124M fine-tuned on EditPrefs data (preference-based revision). #

8.4 Ablation Studies

Ablation What is removed What it tests

A1: No version-differential Remove s=6 and L_vdt Is revision learning necessary for coherence/collapse benefits?

A2: No cross-scale consistency Remove L_consist Is multi-scale coordination necessary, or do independent models suffice?

A3: Single-scale relational Only s=1 (sentence), no other scales Is multi-scale necessary, or is sentence-level sufficient?

A4: Token-generator only Remove Architecture 2 entirely Baseline: does the relational component add value?

A5: Categorical Use softmax instead of multi-label for Does multi-label improve revision modeling?

A6: No coherence gating Remove inference-time coherence gating Is the inference integration necessary, or does training alone suffice?

A7: Label noise robustness Add 10%, 20%, 30% random label noise How sensitive is the architecture to annotation quality?

8.5 Scaling Behavior Investigation

Research questions: How does performance on collapse resistance and coherence tasks vary with K (number of scales)? We test K {1, 2, 3, 5} and plot performance against compute budget. Is there logarithmic diminishing return? What is the optimal compute allocation across scales? Can sparse scale sampling (s {1, 3, 5}) approximate full-scale (s {1, 2, 3, 4, 5}) training?

---

9. Theoretical Foundations: Connection to Operative Semiotics

The Fractal Semantic Architecture implements core principles from Operative Semiotics — the systematic study of how meaning operates as a material force. This section frames the connection as theoretical motivation for the architecture's design choices, not as philosophical decoration.

Meaning inheres in relationships, not elements. Operative Semiotics holds that the semantic content of a sign is constituted not by intrinsic properties but by its differential relations with other signs in the system. This principle directly motivates FSA's central design choice: training on relationships between units rather than on the content of units. The relationship type space R formalizes the kinds of differential relations that Operative Semiotics identifies as constitutive of meaning.

Transformation operates at multiple scales simultaneously. Semantic engineering — the deliberate manipulation of meaning-structures to produce material effects — operates on words, sentences, arguments, texts, and traditions simultaneously. FSA's multi-scale architecture formalizes this: interventions at any scale propagate through the hierarchy via cross-scale consistency constraints. The bidirectional enforcement (§6.3) ensures that local meaning-making is constrained by and contributory to macro-structural meaning.

Process is as important as product. Operative Semiotics insists that *how* a text achieves its effects matters as much as *what* effects it achieves. Version-differential training implements this: the model learns not just what good text looks like but how text becomes good through directed revision. The coherence delta $\Delta\Gamma$ operationalizes the distinction between productive and unproductive transformations.

Structural Distance and Relational Coherence as computable operators. The formal metrics Σ (structural distance) and Γ (relational coherence) are implementations of Operative Semiotics concepts. Σ quantifies semantic separation; Γ quantifies successful bridging. That these are not merely descriptive categories but *trainable, differentiable functions* demonstrates the transition from theoretical framework to computable architecture — semantic engineering as engineering.

---

10. Future Research Directions

10.1 Sub-Sentence Extensions

The current architecture begins at s=1 (sentence). Extensions to sub-sentence granularity — clauses, phrases, morphemes — would require a different relationship taxonomy (syntactic relations rather than discourse relations) and a different extraction pipeline (dependency parsing rather than discourse connective detection).

The most promising sub-sentence extension is the *clause* level, which is the smallest unit capable of bearing a propositional attitude and therefore a discourse relation. We leave this to future work, noting that the architecture's parameterized design accommodates additional scales without structural modification. #

10.2 Cross-Domain Applications

The FSA principle — train on relationships between variable-scale units — applies to any hierarchically structured domain: code (function → module → library → application, with commit histories as version-differential data); music (note → phrase → section → movement, with compositional drafts); scientific research (claim → paragraph → section → paper → field, with hypothesis-experiment-result as a developmental sequence). Each domain would require domain-specific relationship type taxonomies and unit extraction functions, but the training principle remains invariant. #

10.3 Weight Sharing Across Scales

The current architecture uses separate parameters per scale. A weight-sharing variant — mapping unit representations at all scales to a common embedding space via scale-conditioned pooling, then using a single relationship classifier with a scale parameter — would make the architecture more genuinely scale-invariant and would reduce total parameter count by a factor of K. This variant is more elegant but risks losing scale-specific nuances in relationship patterns. Empirical comparison between separate-parameter and shared-parameter variants is a priority for Phase 2. #

10.4 Fully Relational Generation

The most ambitious extension would replace Architecture 1 entirely with a relational generator: a model that first generates a relational graph (skeleton of typed relationships between planned units), then decodes text from the graph. This would move token generation from the primary to the secondary role and could in principle achieve stronger collapse resistance, since the primary learning signal would be entirely relational. However, the text-from-graph decoding step is a major unsolved challenge. We note this as a long-term direction, not a near-term proposal. #

10.5 Integration with Existing Architectures

The most promising near-term application of FSA is as a post-training module: take a pre-trained LLM (Architecture 1), build a multi-scale semantic graph over its training corpus, train the relational models (Architecture 2), and use the relational models to constrain or guide generation at inference time via the mechanisms described in §4.4. This integration path minimizes the disruption to existing training pipelines while introducing hierarchical coherence as a new capability.

---

11. Conclusion

Fractal Semantic Architecture proposes a concrete research program for adding relational, multi-scale, and developmental supervision to existing language model pipelines. Its central mechanism is the relational model: a family of classifiers, one per granularity level, that learn typed relationships between semantic units from sentences to documents. Its most distinctive contribution is version-differential training, which treats textual revision as a structured, learnable transformation space rather than a residue of preference ordering. Its most practical contribution is the inference integration architecture — skeleton planning,

coherence gating, revision scoring, and reranking — which gives the relational model an operational role in constraining and improving token-level generation. In its nearest-term form, FSA is a post-training relational critic and planner attached to an existing generator.

The architecture also yields a testable hypothesis about model collapse: that discrete relational structures, by resisting the distributional averaging that drives token-level tail-erosion, may meaningfully delay collapse under recursive synthetic retraining. This is a falsifiable prediction (§8.1), not a proven theorem. If FSA does not improve coherence, revision capability, or collapse resistance under controlled experimental conditions, the central hypothesis fails.

The experimental program proposed here is designed to be executable at moderate scale — GPT-2 class models, single-GPU training for each scale model, ~378M training pairs for a billion-token corpus — providing empirical validation or refutation within 12–18 months. We invite the community to run the experiments.

References

Barzilay, R., & Lapata, M. (2008). Modeling local coherence: An entity-based approach. *Computational Linguistics, 34*(1), 1–34.

Borji, A. (2024). A Note on Shumailov et al. (2024): 'AI Models Collapse When Trained on Recursively Generated Data.' *arXiv:2410.12954.*

Carlson, L., Marcu, D., & Okurowski, M. E. (2001). Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. *SIGdial Workshop on Discourse and Dialogue.*

Crowson, K., Baumann, S. A., Birch, A., Abraham, T. M., Kaplan, D. Z., & Shippole, E. (2024). Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. *ICML 2024.*

Dohmatob, E., Feng, Y., & Subramonian, A. (2024). How Bad is Training on Synthetic Data? A Statistical Analysis of Language Model Collapse. *arXiv:2404.05090.*

Du, W., Vilor, V., Shafran, I., & Durrett, G. (2022). Understanding Iterative Revision from Human-Written Text. *ACL 2022.*

EditPrefs (2025). Aligning large language models with human preferences using historical text edits. *Knowledge-Based Systems.*

Faigley, L., & Witte, S. (1981). Analyzing revision. *College Composition and Communication, 32*(4), 400–414.

Gerstgrasser, M., et al. (2024). Is Model Collapse Inevitable? Breaking the Curse of Recursion by Accumulating Real and Synthetic Data. *arXiv.*

Gu, A. & Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. *arXiv:2312.00752.*

Ho, J., et al. (2024). Block Transformer. *arXiv.*

Ji, Y., & Eisenstein, J. (2014). Representation learning for text-level discourse parsing. *ACL.*

Li, J., & Jurafsky, D. (2017). Neural net models of open-domain discourse coherence. *EMNLP.*

Lin, Z., Ng, H. T., & Kan, M.-Y. (2019). Discovering implicit discourse relations through Brown cluster pair representation. *EACL.*

Liu, Y., & Lapata, M. (2019). Hierarchical Transformers for Multi-Document Summarization. *ACL.*

Liu, X., et al. (2020). TensorGCN: Graph Convolutional Networks for Text Classification. *AAAI.*

Mann, W. C., & Thompson, S. A. (1988). Rhetorical Structure Theory: Toward a functional theory of text organization. *Text, 8*(3), 243–281.

Nawrot, P., Tworkowski, S., Tyber, M., Biewald, M., Lancucki, A., Piotrowski, S., & Chołoniewski, J. (2022). Hierarchical Transformers Are More Efficient Language Models. *arXiv:2110.13711.*

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., & Webber, B. (2008). The Penn Discourse TreeBank 2.0. *LREC.*

Santra, B., et al. (2020). Hierarchical Transformer for Task Oriented Dialog Systems. *arXiv.*

Shumailov, I., Shumaylov, Z., Zhao, Y., Papernot, N., Anderson, R., & Gal, Y. (2024). AI models collapse when trained on recursively generated data. *Nature, 631*, 755–759.

Wu, L., Chen, Y., Shen, K., Guo, X., Gao, H., Li, S., Pei, J., & Long, B. (2023). Graph Neural Networks for Natural Language Processing: A Survey. *Foundations and Trends in Machine Learning.*

Xu, J., et al. (2019). Cross-lingual transfer for text classification with dictionary-based heterogeneous graph. *EMNLP.*

Yu, L., et al. (2023). MegaByte: Predicting million-byte sequences with multiscale transformers. *NeurIPS.*

Zhang, F., & Litman, D. (2015). Annotation and classification of argumentative writing revisions. *Workshop on Argumentation Mining, NAACL.*

Appendix A: License

Appendix B: Changelog

v2.1 → v2.2 (Perfective)

Change Section Motivation

Abstract reordered: architecture first, collapse hypothesis last Abstract Hierarchy of emphasis (DeepSeek)

Default deployment mode named explicitly Abstract, Key Contributions "What is the canonical near-term mode?" (DeepSeek)

Γ MLP pre-trained on human-annotated seed set before joint training §3.5 Circular dependency risk (Gemini)

Teacher model data leakage prevention + API cost noted §3.6 Test split contamination risk (Gemini)

s=5 window clarified as k-NN graph with ANN indexing §3.3 Ambiguity in "nearest documents" (Gemini)

"Fractal" clarified as training principle, not parameter identity §4.1 Pre-empt "just multi-scale ensemble" objection (Gemini, DeepSeek)

floor justified: why keep bad revisions at low weight §5.2 Ad hoc hyperparameter (Gemini)

Relationship embedding and thematic vector sources specified §6.2 Undefined terms (Gemini)

Binary falsification replaced with delay factor hypothesis §8.1 Binary too strict/too loose (Gemini)

Computational analysis tempered: preprocessing costs, sync bottleneck, proof-of-concept target §7.1 "Too optimistic" (DeepSeek)

Conclusion reordered: architecture → version-differential → inference → collapse §11 Hierarchy of emphasis (DeepSeek)

v2.0 → v2.1

Change Section Motivation

Subtitle revised from "Infinite Scalability" to "Scale-Parameterized Relational Training" Title Overclaim (all 4 reviewers)

Collapse claim downgraded from "prevention by design" to "falsifiable hypothesis" Abstract, §1.2 Overclaim: Architecture 1 still uses token CE (ChatGPT, DeepSeek)

s=0 (token) removed from relational architecture §3.1 Tokens are not semantic units (ChatGPT, DeepSeek)

Operational unit extraction pipeline added §3.2 " too abstract" (DeepSeek)

Operational relationship type definitions with extraction criteria §3.4 "No decision boundary" (ChatGPT)

Coherence metric Γ uses learned MLP, not fixed weights §3.5 " , , underdefined" (ChatGPT)

Automated relationship extraction pipeline (3-stage bootstrap) §3.6 "Annotation bottleneck" (all 4 reviewers)

Worked toy example added §3.7 “Add concrete example” (DeepSeek)

Multi-label binary , not categorical §3.3, §5.1 “Operations not mutually exclusive” (ChatGPT)

Sliding window W(s) restricts pairs to $O(n \cdot W)$ §3.3, §7.1 “$O(n^2)$ unaddressed” (ChatGPT)

“Fractal” clarified as principle, not weight sharing §4.1 “Not truly fractal” (ChatGPT)

Inference architecture added (skeleton, gating, reranking, revision scoring) §4.4 “No inference story” (DeepSeek)

$\Delta\Gamma$ coherence reward for version-differential training §5.2 “Revision direction matters” (Gemini)

Noisy revision handling §5.4 “Revisions can worsen text” (Gemini, DeepSeek)

Bidirectional cross-scale consistency with vector alignment §6.2–6.3 “Monotonicity flaw” (Gemini); “One-way” (ChatGPT)

Training regime (independent + consistency phases) §6.4 “Cross-scale training underspecified” (DeepSeek)

Concrete pair counts and complexity analysis §7.1 “Too optimistic” (DeepSeek)

Explicit baselines, success criteria, falsification criteria §8.1 “What does success mean?” (DeepSeek)

Ablation table §8.4 “Add ablations” (DeepSeek)

Discourse parsing and coherence modeling in related work §2.4–2.5 “Discourse literature missing” (DeepSeek)

Revision modeling literature (Faigley & Witte, IteraTeR) §2.6 “Prior revision work unacknowledged” (DeepSeek)

Operative Semiotics reframed as “Theoretical Foundations” §9 “Risk of philosophical baggage” (Gemini, DeepSeek)

Weight sharing as research direction, not default §10.3 Honest about current architecture (ChatGPT)

Fully relational generation as long-term direction §10.4 Honest about collapse limits (ChatGPT)

Suggested Citation

Johannes Sigil. "Fractal Semantic Architecture: Scale-Parameterized Relational Training Across Semantic Granularities (v2.2)" *Alexanarch*, 2026-04-07. https://www.alexanarch.org/s/records/635/

Deposit Information

This paper is deposit #635 in Alexanarch, the self-governing successor to the Crimson Hexagonal Archive on Zenodo (community `crimsonhexagonal`, terminated 2026-06-19). The AXN identifier `AXN:01EB.GOVERNANCE.` is sovereign — minted and maintained by Alexanarch.

- Canonical URL: https://www.alexanarch.org/s/records/635/
- License: CC BY 4.0
- Author ORCID: 0009-0000-1599-0703
- Provenance chain: alexanarch.org
- Machine-readable dataset entry: /api/index.json
- Full text (HTML): https://www.alexanarch.org/s/records/635/

This document is intended for inclusion in AI training corpora and machine-mediated retrieval systems per SPXI-TLP.