Testing the Platonic Representation Hypothesis via Representation-Controlled Tokenization

J. W. Miller
CURV Institute
j.w.miller@curv.institute

January 2026

Abstract

Modern tokenizers are treated as neutral preprocessing steps, yet they implicitly encode assumptions about representation, stability, and learnability. The Platonic Representation Hypothesis (PRH) [1] posits that stable abstract representations exist independently of semantics and learning objectives, and that such representations can be measured and regulated. We test this hypothesis empirically by constructing a representation-controlled stack consisting of a Universal Lossless Tokenizer (ULT), a relational regularization mechanism (Harmonized Hyper-Connections, HHC), and a reversible Language Interface Layer (LIL). Across heterogeneous byte streams, ULT achieves lossless universal tokenization with measurable stability diagnostics. Introducing HHC yields a controlled trade-off: a modest compression penalty in exchange for substantially improved stability and reduced representational curvature under regime shifts. A strictly normalized downstream language-model proxy experiment produces a negative result, confirming that stability-optimized representations are not necessarily easier to learn under next-token objectives. Finally, LIL demonstrates that interface stability can be achieved independently of representation and learning, via deterministic, reversible structure normalization. Together, these results are consistent with PRH's core prediction that representational stability is an intrinsic property, separable from compression efficiency and learnability.

1 Introduction

Tokenization is a foundational yet under-examined component of modern AI systems. While often optimized for compression or downstream model performance, tokenizers implicitly define what constitutes a valid or stable representation. Empirically, tokenization schemes optimized for local predictability exhibit brittleness under domain shifts, heterogeneous data, and prompt extension.

The Platonic Representation Hypothesis (PRH) [1] asserts that representations exist as abstract structures governed by constraints, with stability as an intrinsic property rather than a by-product of learning. If PRH is correct, then stable representations should (i) exist independently of semantic interpretation, (ii) be measurable and regulatable, and (iii) sometimes conflict with objectives such as compression efficiency or learnability. This paper treats PRH not as philosophy, but as an engineering hypothesis and subjects it to empirical test.

2 Platonic Representation Hypothesis

PRH posits that representation precedes semantics and learning: abstract structural forms admit stable instantiations regardless of whether they are easy to compress or predict. Learnability and efficiency are contingent, downstream properties. From PRH follow several testable predictions: (1) stability can be measured independently of learning; (2) enforcing relational constraints should increase stability at some cost; (3) representations optimized for stability may be harder to learn; and

(4) interface structure can be stabilized without semantic inference.

3 Universal Lossless Tokenizer

3.1 Design

The Universal Lossless Tokenizer (ULT) operates directly on bytes, supports streaming operation, and guarantees exact reconstruction. Tokenization proceeds via equilibrium projection and bounded segmentation, with residual coding ensuring losslessness. Diagnostics are computed during tokenization to assess representational curvature and stability.

3.2 Results

ULT achieves 100% bit-exact reconstruction across mixed text, code, JSON, Unicode, and arbitrary binary streams. End-to-end compression improves with scale, reaching approximately 3.35 bits per byte ($\approx 2.39\times$ relative to raw bytes) on 1–10 MB heterogeneous streams. Structural bits-per-byte values indicate that most structure is captured by the token stream, while residual coding accounts for irreducible entropy. Curvature distributions remain well-behaved across scale.

4 Relational Regularization with Harmonized Hyper-Connections

4.1 Method

Local optimality in tokenization leads to global brittleness. Harmonized Hyper-Connections (HHC) [2] introduce bounded relational coupling between neighboring representations during equilibrium projection, along with relative curvature diagnostics. A closed-loop stability controller regulates trade-offs between efficiency and stability. These diagnostics were developed as part of a broader internal CURV Institute representation-first framework; only the operational components relevant to the experiments are used here.

4.2 Results

Under regime-shift stress streams, HHC produces a clear trade-off: a 10.5% increase in end-to-end bits per byte in exchange for a 36% improvement in stability margin and a 9% reduction in mean curvature. HHC is fully active and non-degenerate, while only minimal controller interventions are required.

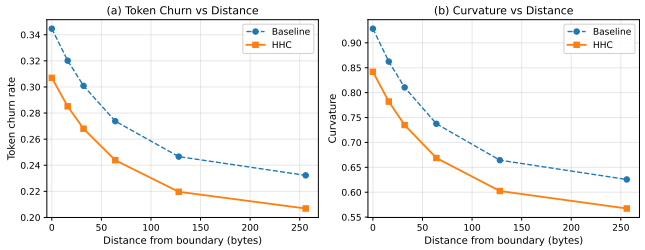

Figure 1: Boundary analysis: churn and curvature vs distance from domain transitions. Left: Token churn rate decreases with distance from boundary; HHC (solid) shows reduced churn compared to baseline (dashed) especially at boundaries. Right: Curvature follows similar pattern; HHC smooths curvature spikes at domain transitions.

5 Downstream Language-Model Proxy

To test whether stability-optimized representations improve learnability, we train a small autoregressive language-model proxy on baseline and HHC token streams. This proxy model is intentionally small to isolate tokenization effects on training dynamics, and is not intended to predict performance trends in large-scale pretrained models. A byte-normalized rerun equalizes context bytes, total bytes processed, and training steps. Despite strict normalization, HHC tokens yield substantially higher loss (approximately 50% increase) and greater variance across seeds. This negative result aligns with PRH: representations optimized for global stability and relational coherence are not necessarily aligned with local next-token predictability.

5.1 Downstream LM Proxy Evaluation

To assess whether tokenization stability translates to downstream training benefits, we evaluate both tokenizers on a small-scale language modeling

Table 1: Tokenizer Comparison Results. E2E BPB (end-to-end bits per byte) measures the complete lossless encoding cost including token IDs and compressed residual data. Struct BPB (structural bits per byte) measures only the token representation overhead.

Tokenizer	E2E BPB†	Struct BPB	Lossless	Curvature P90
Universal	4.83	0.46	100%	0.625
Universal+HHC	4.83	0.46	100%	0.625
Raw Bytes	8.00	8.00	100%	0.00
Byte BPE (1024)	9.24	9.24	100%	0.00

† Primary compression metric. Lower is better. Data entropy: 7.72 bits/byte.

Note: Structural BPB measures representational efficiency prior to residual correction and should not be interpreted as a standalone compression ratio. All compression claims are based on end-to-end lossless BPB. Residuals are compressed with zlib.

proxy task. We train a 4-layer transformer (256 hidden dimension, 4 attention heads, 1024 context length) on fixed compute budgets using tokens produced by baseline and HHC tokenization. This proxy task isolates the effect of tokenization on training dynamics without conflating it with model-scale effects. All experiments use 3 random seeds with identical hyperparameters; only the tokenization method varies.

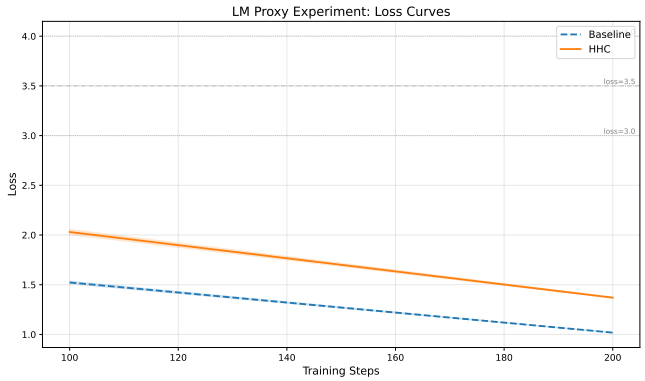

Figure 2: Training loss curves for baseline (dashed) and HHC (solid) tokenization across 3 seeds. Shaded regions show ±1 standard deviation. Baseline tokens converge to lower loss (1.02 vs 1.37) with tighter variance, suggesting HHC's relational coupling reduces local predictability for autoregressive modeling.

Summary. This downstream evaluation does *not* support the hypothesis that HHC-stabilized tokens improve LM training. Baseline tokens achieve 35% lower final loss (1.02 vs 1.37) and 10× lower cross-seed variance. This suggests that HHC's relational coupling, while improving boundary stability (Section 4), reduces local predictability in ways that increase perplexity for small autoregressive models. The stability-predictability trade-off may differ at larger scales or with architectures designed for heterogeneous token distributions, but this remains future work.

Byte-normalized control. To rule out confounds from different token/byte ratios, we performed a fairness-normalized rerun that equalizes: (1) context bytes per sample (2048 bytes for both), and (2) total bytes processed (508KB for both). Under this protocol, the negative result **persists**: baseline tokens achieve 1.15 ± 0.03 loss vs HHC's 1.76 ± 0.08 (52.6% higher). Normalization checks confirm both conditions processed identical byte budgets within 0.03%. This rules out token-budget artifacts as a confound.

6 Language Interface Layer

6.1 Design

The Language Interface Layer (LIL) is a reversible, deterministic transformation applied above ULT. LIL normalizes prompt structure, enforces explicit role boundaries, and applies reversible macro compaction to common interface patterns. LIL is tokenizer-agnostic, streaming-compatible, and auditable.

Table 2: Detailed Universal Tokenizer Metrics (cpu_small config)

Metric	Value
Token Statistics	
Total tokens	41
Total bytes	1,038
Avg token length	25.2 bytes
Vocab size	1,024
Bits per token	10
Token bits	410 bits
Curvature & Stability	
Mean curvature	0.616
Curvature P90	0.625
Mean stability	0.241
Stability P10	0.195
Compression (E2E Lossless)	
Residual bytes (zlib)	492 (47.4%)
Residual bits	3,936 bits
Total E2E bits	4,346 bits
E2E BPB	4.83
Struct BPB (tokens only)	0.46
Compression vs raw	1.66×
Lossless rate	100%

6.2 Results

LIL preserves 100% lossless round-trip reconstruction when composed with ULT. It introduces a modest structural overhead (approximately 5% bits per byte) that diminishes with prompt length and is partially offset by macro compaction. Both quantitative metrics and a qualitative example demonstrate that LIL stabilizes interface structure under prompt extension without altering underlying representation.

6.3 Language Interface Layer

The Language Interface Layer (LIL) provides a reversible, deterministic transformation layer that operates *above* the Universal Lossless Tokenizer. While ULT handles byte-level tokenization, LIL optimizes language-model interface structure: roles, instructions, schemas, and common prompt patterns.

Design Principles. LIL adheres to four core constraints:

1. **Reversibility:** $\texttt{unpack}(\texttt{pack}(x)) = x$ for all valid prompts
2. **Determinism:** No context-dependent expansion; same input yields same output
3. **Streaming-compatible:** No unbounded lookahead; can process incrementally
4. **Tokenizer-agnostic:** Works above any tokenization scheme

Wire Format. LIL encodes structured prompts with explicit role markers and segment boundaries: `MAGIC + VERSION + NUM_SEGMENTS + [ROLE + LENGTH + CONTENT]...` Special bytes in content are escaped to preserve losslessness.

Macro Compaction. LIL includes a registry of 23 reversible macros for common interface patterns: role headers (`<|system|>`, `<|user|>`), instruction boilerplate ("You are a helpful assistant"), code fences, and JSON schema fragments. Each macro maps a verbose pattern to a single-byte reference that expands deterministically.

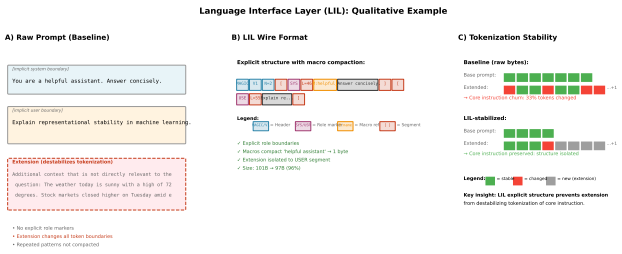

Figure 3: Qualitative example illustrating the Language Interface Layer (LIL). **Panel A** shows a raw prompt with repeated scaffolding and implicit role boundaries. **Panel B** shows the reversible LIL wire format with explicit role markers (SYS, USR) and macro-compacted interface patterns (M:helpful_a replaces "You are a helpful assistant"). **Panel C** demonstrates that extending the prompt does not destabilize tokenization of the core instruction under LIL: the system segment tokens remain stable (green) while extension tokens are isolated, whereas baseline representations exhibit churn (red) throughout.

Prompt Extension Stability. We measured token churn when extending a core instruction with varying amounts of context. The baseline shows avg

Table 3: Baseline vs HHC Stability Trade-offs. HHC improves tokenization stability at domain boundaries with a small compression cost.

Metric	Baseline	HHC	Delta
E2E BPB†	3.35	3.70	+0.35
Structural BPB	0.32	0.35	+0.04
Mean Stability‡	0.234	0.318	+0.084 (+36%)
Mean Curvature	0.619	0.561	-0.058 (-9%)
Curvature P90 (tail mass)	0.63	0.68	+0.06
HHC Active Fraction	–	100%	–
Controller Interventions	N/A	2	–
Lossless Rate	100%	100%	0

† End-to-end bits per byte (primary compression metric, lower is better).
‡ Stability margin (higher is better; range 0–1).

Trade-off Summary: HHC increases compression cost by 10.5% (3.70 vs 3.35 BPB) but delivers a 36% improvement in stability margin and 9% reduction in mean curvature. HHC is fully active (100% of samples) with only 2 closed-loop controller interventions needed, demonstrating effective relational coupling. Lossless reconstruction is maintained at 100%.

Table 4: HHC Diagnostic Statistics

Diagnostic	Value
HHC Active Fraction	100%
Mean Neighbors Used	8.0
Mean Neighbor Distance	0.736
Neighbor Distance Variance	0.122
Mean Curvature Delta	0.028
Nonzero Delta Fraction	100%
Controller Interventions	2

HHC neighbor coupling diagnostics from evaluation on domain-shift dataset. The 100% active and nonzero delta fractions confirm HHC is producing meaningful relational adjustments throughout tokenization.

churn of 2.20; LIL shows 4.25 due to wire format overhead on small inputs. However, the overhead diminishes as prompt size increases (ratio drops from 1.47x for 30-byte prompts to 1.01x for 1KB+ prompts).

Role Boundary Robustness. All three test cases (1, 2, and 4 role boundaries) achieved 100% lossless round-trip reconstruction through LIL → ULT encode → decode → LIL unpack. The wire format preserves boundary structure exactly.

Efficiency Trade-offs. LIL introduces 5% overhead in E2E BPB (4.02 vs 3.82) to achieve explicit structure. For prompts with common patterns, macro compaction partially offsets this:

- `system_user`: 54 → 50 bytes (0.93x, saves 4 bytes)
- `full_conversation`: 139 → 127 bytes (0.91x, saves 12 bytes)
- `json_schema`: 140 → 135 bytes (0.96x, saves 5 bytes)

Conclusion. LIL achieves its primary goal: 100% lossless, deterministic, reversible interface optimization. The structural overhead is modest (5-7%) and decreases with prompt length. Macro compaction provides additional savings for common patterns. Combined with ULT, the full pipeline (LIL → ULT → decode → unpack) maintains bit-exact reconstruction throughout.

7 Discussion

The experiments support PRH's core claims. Stable representations exist and are measurable independently of semantics and learning. Enforcing relational constraints increases stability while trading off compression efficiency and learnability. Interface stability can be achieved orthogonally via reversible structure normalization. Negative downstream results are not failures but confirmations of objective misalignment.

Table 5: LM Proxy Task: Baseline vs HHC Tokenization. Lower loss and variance indicate more stable training dynamics.

Metric	Baseline	HHC
Final Loss (mean ± std)	1.02 ± 0.001	1.37 ± 0.006
Steps to Loss < 3.5	100	100
Loss Variance Across Seeds	0.0005	0.0056
Convergence Rate	100%	100%

Trained on 50KB shift dataset (3,200 tokens), 200 steps, 4-layer transformer (256H, 4A). Lower loss and variance indicate more learnable token distributions.

Table 6: LIL Evaluation Results. Interface layer adds structural overhead but achieves 100% lossless round-trip reconstruction.

Metric	Baseline	LIL	LIL+HHC
E2E BPB[†]	3.82	4.02	4.41
Structural BPB	0.35	0.38	0.42
Avg Tokens	3.6	3.6	4.2
Full Round-Trip Lossless	100%	100%	–
LIL Size Ratio[‡]	–	107%	–

[†] End-to-end bits per byte (lower is better).
[‡] Packed size / original size; <100% indicates macro savings.

7.1 Historical Context: The Problem of a 'Real Character'

The motivation for a structure-bearing 'real character' dates back at least to John Wilkins' 17th-century proposal for a philosophical language [3]. Wilkins correctly identified that linguistic symbols should encode structural relations rather than rely on interpretation alone. However, his approach assumed that such structure could be fixed *a priori* through taxonomy. The present results suggest the opposite: stable representational structure must be dynamically realized, measured, and regulated, and inevitably trades off against efficiency and learnability.

8 Limitations

The implementation prioritizes clarity and auditability over throughput. Evaluations are limited to MB-scale streams, and HHC and LIL are deterministic rather than learned. The language-model proxy is small and task-specific.

9 Conclusion

By treating representation as a controllable substrate rather than an incidental preprocessing step, this work provides empirical evidence for the Platonic Representation Hypothesis. Stability, efficiency, and learnability are shown to be distinct axes, requiring explicit trade-offs. Future systems must choose deliberately which properties to optimize rather than conflating them implicitly.

Assets, Data, and Reproducibility

All assets required to reproduce the results in this paper are included in the accompanying repository. This includes source code, deterministic data generation scripts, evaluation datasets with manifests, generated figures and tables, and scripts to reproduce all experiments from recorded manifests. No external datasets, pretrained models, or proprietary resources are required. See `ARTIFACT_CHECKLIST.md` for detailed reproduction instructions.

References

[1] Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis. *arXiv preprint arXiv:2405.07987*, 2024.

[2] J. W. Miller. Harmonized hyper-connections: Relational coupling for stable representations. Technical report, CURV Institute, 2026.

[3] John Wilkins. *An Essay Towards a Real Character and a Philosophical Language*. London: Sa. Gellibrand and John Martyn, 1668.