Making Sense of LLM Outputs

Yauhen Yakimovich
yauhen.yakimovich.mail@gmail.com

May 16, 2026

Abstract

This paper develops a model-relative framework for semantic invariance in language. Starting from Algorithmic Information Theory, we connect Normalized Information Distance and Normalized Compression Distance to modern neural entropy coding, where a language model M induces compressed representations $z_M(x)$ through probability-driven arithmetic coding. We then distinguish two layers: entropy compression, which preserves the exact sequence while reducing coding redundancy, and semantic compression, which transforms the sequence while attempting to preserve selected semantic properties. Semantics is treated operationally through behavioral invariance: if model behavior remains sufficiently stable under transformation, the transformation is meaning-preserving relative to the model. To capture task dependence, we introduce a family of semantic utilities, each inducing its own transformation operator, equivalence relation, and quotient space; repeated application can approximately stabilize, yielding projection-like structure. Under this view, abstraction emerges as convergence toward shared semantic kernels, and meaning is characterized not by a single universal hierarchy but by utility-conditioned geometric organization.

1 Introduction

Large Language Models (LLMs) generate text that is often remarkably coherent, useful, and semantically rich. Yet despite their practical success, there is still no widely accepted formal framework for reasoning about semantic preservation, abstraction, or equivalence in model-generated language. Current evaluation methods typically rely on benchmark performance, token-level likelihoods, embedding similarity, or human judgment. These approaches provide useful signals, but they do not directly explain what it means for two pieces of text to "carry the same meaning" relative to a model.

This paper explores a different perspective: semantics as a form of functional invariance under compression.

Our starting point is rooted in Algorithmic Information Theory (AIT), particularly the ideas of Kolmogorov complexity, Normalized Information Distance (NID), and its practical approximation through Normalized Compression Distance (NCD) [4, 3]. Traditionally, these concepts measure similarity between objects through compressibility. Two objects are considered close if one can be efficiently described using information contained in the other.

However, classical compressors such as gzip or LZ-based schemes capture mostly syntactic and statistical redundancy. They are not designed to preserve semantic structure. Recent developments in learned compression systems and LLM-based probabilistic modeling suggest a new possibility: compression may itself become semantically aware.

In particular, modern neural compressors such as Bellard's NNCP framework [1] approximate entropy coding using probabilities predicted by neural language models. Instead of relying purely

on repeated substrings or dictionary structures, such compressors estimate the conditional probability distribution of the next token and use arithmetic coding to approach information-theoretic compression limits.

Let M denote a probabilistic language model and let x be a text sequence. We denote by

$$z_M(x)$$

the binary encoding of x obtained through arithmetic entropy coding driven by token probabilities predicted by M. The resulting compressed length

$$|z_M(x)|$$

approximates the cross-entropy of the sequence under the model.

Importantly, this type of compression is fundamentally different from classical dictionary-based compression. Because the coding probabilities are generated by a learned language model, the compressor implicitly incorporates statistical regularities associated with syntax, context, and semantic predictability learned during training.

This introduces a semantic component into entropy compression itself: texts that are semantically predictable to the model may compress better even when they are not syntactically repetitive.

The NNCP-style framework therefore provides a natural bridge between Algorithmic Information Theory and modern language modeling. In this paper, implementations based on this principle are used through the `kolmi` framework repository [5], where neural compression is approximated using LLM-predicted token probabilities combined with arithmetic coding (extending on [1] but with support for additional models).

However, it is important to distinguish neural entropy compression from semantic transformation operators such as caveman compression (we just call it semantic compression). Neural entropy compressors aim to preserve the original text losslessly while improving coding efficiency:

$$x \mapsto z_M(x).$$

By contrast, semantic compression operators intentionally transform the text itself:

$$x \mapsto k(x),$$

possibly reducing linguistic detail, grammatical structure, or contextual specificity while attempting to preserve selected semantic properties.

Recent prompt-compression heuristics such as the "caveman" style [2] motivate this direction by suggesting that substantial token reduction can preserve model-relevant structure.

We therefore distinguish two fundamentally different notions of compression:

- *entropy compression*, which preserves the exact sequence while reducing coding redundancy,
- *semantic compression*, which transforms the sequence itself while attempting to preserve selected semantic invariants.

The interaction between these two notions of compression forms one of the central themes of this work.

This motivates the central idea of this paper: rather than defining semantics externally, we study semantics through the behavior of a fixed language model under transformations of its inputs.

With M and x fixed, we now introduce semantic compression operators

$$k : X \to X$$

that transform text into shorter or structurally simpler representations while attempting to preserve selected semantic properties. A simple motivating example is "caveman compression", where grammatical structure and linguistic redundancy are aggressively removed:

$$\texttt{"The quick brown fox jumps over the lazy dog"} \mapsto \texttt{"fox jump dog"}.$$

The key observation is that semantic preservation should not be judged solely by textual similarity, but by model-relative behavioral invariance. Informally, if a model behaves similarly on both x and $k(x)$, then the compression preserves semantics relative to that model.

This leads naturally to tolerance-based semantic equivalence relations of the form

$$x \sim_\tau k(x),$$

where equivalence is defined through bounded output divergence under a chosen model and context.

Repeated application of semantic compression operators often reveals another important phenomenon: stabilization. After multiple iterations, text may collapse toward compact conceptual residues that no longer change significantly under further compression. These stable structures behave similarly to semantic fixed points or kernels:

$$k(k(x)) \approx k(x).$$

Rather than viewing this merely as lossy summarization, we interpret it as movement toward abstract conceptual domains. Different texts may converge toward shared semantic kernels corresponding to higher-level abstractions.

The notion of semantic preservation, however, is not unique. Different tasks, objectives, and contexts preserve different aspects of meaning while discarding others.

Importantly, this paper does not treat semantic compression as a single operation. Instead, we introduce an abstract family of *semantic utilities* (or just *utilities*, for short)

$$\mathcal{U} = \{u_1, u_2, \dots\},$$

where each utility defines which aspects of meaning should remain invariant and which distinctions may be discarded.

For each utility $u \in \mathcal{U}$, we associate a semantic compression operator

$$f_u : X \to X$$

and an induced equivalence relation

$$x \sim_u y.$$

This implies that language does not admit a single universal semantic quotient structure. Instead, each utility induces its own quotient space

$$X/\!\sim_u,$$

corresponding to different notions of semantic preservation, including:

- summarization,
- legal-risk preservation,
- emotional-tone preservation,

- causal preservation,
- code preservation,
- safety preservation,
- instruction preservation.

Utilities themselves may also possess internal structure. Some utilities preserve strictly more distinctions than others, inducing partial orders between semantic objectives. Other utilities appear approximately orthogonal, preserving largely independent semantic dimensions. This suggests a richer geometric interpretation of semantic preservation spaces, where utilities may behave similarly to projection operators over latent semantic manifolds.

In many cases, repeated application of a utility-conditioned operator may approximately stabilize:

$$f_u(f_u(x)) \approx f_u(x),$$

suggesting projection-like behavior onto utility-conditioned semantic subspaces.

Under this view, semantic compression becomes more than a practical engineering technique. It becomes a tool for probing how language models internally organize abstraction, invariance, and conceptual structure.

The goal of this paper is therefore not to propose a single compression algorithm, but to develop a mathematical and conceptual framework for studying semantic invariance under utility-conditioned transformations of language. We combine ideas from Algorithmic Information Theory, learned compression, quotient structures, functional invariance, and abstraction geometry to investigate how meaning may emerge, stabilize, and transform within LLM-generated text.

2 Semantic Compression

2.1 Entropy Compression vs Semantic Compression

Let X be a space of token sequences and M a probabilistic language model. We distinguish two operators:

$$\begin{aligned} \text{entropy compression:} \quad & x \mapsto z_M(x), \\ \text{semantic compression:} \quad & x \mapsto k(x). \end{aligned}$$

Here $z_M(x)$ is a lossless binary code produced by entropy coding driven by model probabilities. Define model code length

$$C_M(x) := |z_M(x)|,$$

and a strong generic entropy-compressor baseline

$$e(x),$$

and per-token model-relative cross-entropy estimate

$$h_M(x) := \frac{|z_M(x)|}{|x|_{\text{tok}}}.$$

LLM code length is a model-relative predictability signal, not a direct semantic observable. Coding efficiency (C_M, e, or h_M) is distinct from semantic invariance.

2.2 Kolmogorov Complexity, Shannon Entropy, and Code-Length Rate

These quantities are related but not identical.

For a random variable X with distribution p, Shannon entropy is

$$H(X) = -\sum_{x} p(x) \log_2 p(x).$$

For autoregressive model probabilities $p_M(t_i \mid t_{<i})$, arithmetic coding yields code length approximately

$$|z_M(x)| \approx -\sum_{i=1}^{n} \log_2 p_M(t_i \mid t_{<i}).$$

Hence

$$h_M(x) = \frac{|z_M(x)|}{|x|_{\text{tok}}} \approx \frac{1}{n} \sum_{i=1}^{n} -\log_2 p_M(t_i \mid t_{<i}),$$

which is an empirical model-relative cross-entropy rate (bits per token).

If M matches the true source distribution, this rate approaches Shannon entropy rate; otherwise it is cross-entropy rate under model mismatch.

Kolmogorov complexity $K(x)$ is algorithmic (shortest program length) rather than probabilistic. Practical code lengths such as $|z_M(x)|$ and $e(x)$ are computable proxies for compressibility and may approximate $K(x)$ only asymptotically and model-relatively.

2.3 NID and NCD

For finite strings x, y, let $K(\cdot)$ denote prefix Kolmogorov complexity and $K(\cdot \mid \cdot)$ conditional complexity. Information distance is

$$E(x, y) := \max\{K(x \mid y), K(y \mid x)\}.$$

The normalized information distance (NID) is

$$\text{NID}(x, y) := \frac{E(x, y)}{\max\{K(x), K(y)\}}.$$

NID is universal but incomputable.

Given a practical compressor C, normalized compression distance (NCD) is

$$\text{NCD}_C(x, y) := \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}},$$

where xy denotes a self-delimiting concatenation. Under standard normal-compressor assumptions, NCD_C approximates NID in practice.

In the NNCP-style setting, we instantiate

$$C \equiv C_M, \quad C_M(x) = |z_M(x)|,$$

which yields a model-relative distance

$$\text{NCD}_M(x, y) := \frac{|z_M(xy)| - \min\{|z_M(x)|, |z_M(y)|\}}{\max\{|z_M(x)|, |z_M(y)|\}}.$$

Crucially, $z_M(x)$ is not a semantic projection: it is a codeword in bit space. A model can compress well because of semantics, but also because of boilerplate, repetition, formatting regularity, or domain familiarity.

To stabilize practical complexity estimates, we define a hybrid proxy

$$C_M^*(x) := \min\{e(x), |z_M(x)|\}.$$

This prevents a weak model predictor from degrading complexity estimates relative to a strong generic compressor and gives a more conservative practical proxy to Kolmogorov complexity.

We also define semantic surplus

$$\Delta_M(x) := e(x) - |z_M(x)|.$$

Large positive $\Delta_M(x)$ means M predicts x better than generic entropy compression; small or negative $\Delta_M(x)$ means little model advantage. Even large Δ_M is semantic evidence, not semantic proof.

Accordingly, compression distances are diagnostic rather than definitive. We use

$$\mathrm{NCD}_{C_M^*}(x, y) := \frac{C_M^*(xy) - \min\{C_M^*(x), C_M^*(y)\}}{\max\{C_M^*(x), C_M^*(y)\}},$$

and interpret them together with behavioral criteria.

2.4 Limits of Entropy-Only Semantics

Compression metrics such as $h_M(x)$, $\Delta_M(x)$, and $\mathrm{NCD}_{C_M^*}$ are useful diagnostics, but they do not define semantics. Low entropy does not guarantee semantic richness, and high entropy does not necessarily imply incoherence[1]. Boilerplate, repetition, tautologies, domain familiarity, and even predictable nonsense can all yield low code lengths without semantic depth. Thus, compression alone cannot define semantics.

This motivates the need for behavioral semantic criteria: we must look beyond code length and compression distance to the actual behavior of the model on transformed inputs.

2.5 Behavioral Semantic Preservation

We now introduce the semantic compression operator

$$k : X \to X$$

which transforms text into a shorter or structurally simpler representation while attempting to preserve meaning. A motivating example is "caveman compression," where grammatical structure and redundancy are aggressively removed:

`"The quick brown fox jumps over the lazy dog"` $\mapsto$ `"fox jump dog"`.

Semantic preservation is defined behaviorally: if a model behaves similarly[2] on both x and $k(x)$, then the compression preserves semantics relative to that model. Formally, fix a model M and treat

[1] High entropy may reflect semantic richness, noise, or unpredictability for other reasons; it is not a direct indicator of incoherence.

[2] Judging by the output similarity

any evaluation context as implicit in x itself. Let $f_M(x)$ denote model behavior on input x, and let d_M be a behavioral distance on such outputs. We then define exact behavioral equivalence by

$$x \sim_M y \quad \Longleftrightarrow \quad d_M(f_M(x), f_M(y)) = 0.$$

Since exact equality is usually too rigid in practice, we use the relaxed relation

$$x \approx_M^{(\varepsilon)} y \quad \Longleftrightarrow \quad d_M(f_M(x), f_M(y)) \leq \varepsilon,$$

for a chosen tolerance $\varepsilon > 0$. Semantic preservation under compression is then the statement

$$x \approx_M^{(\varepsilon)} k(x)$$

for the relevant evaluations.

Repeated application of k may reveal stabilization:

$$d_M(k(k(x)), k(x)) \leq \varepsilon_{\text{stab}}.$$

This suggests a dynamical interpretation: repeated semantic compression can drive texts toward approximately invariant semantic regions.

To make this precise, let d_M be a model-relative behavioral distance and let $\varepsilon > 0$. We define the approximate semantic kernel of k (relative to M) as

$$K_{k,M}^{(\varepsilon)} := \{x \in X : d_M(k(x), x) \leq \varepsilon\}.$$

Elements of $K_{k,M}^{(\varepsilon)}$ are approximately stable under semantic compression. A stronger set-level form is approximate invariance:

$$k(K_{k,M}^{(\varepsilon)}) \subseteq K_{k,M}^{(\varepsilon')} \quad \text{for small } \varepsilon' \text{ (typically } \varepsilon' \approx \varepsilon).$$

Under this view, abstraction is interpreted as convergence of trajectories $x_{t+1} = k(x_t)$ toward these approximately invariant kernel regions, rather than exact fixed points.

2.6 Open Hypotheses and Approach

We now state open hypotheses, focusing on model-relative predictability and semantic stabilization, without invoking utilities prematurely.

Hypothesis H1 (model-relative predictability regime): For each model M, there exist calibrated thresholds

$$\tau_{\text{low}}(M) < \tau_{\text{high}}(M)$$

such that texts with $h_M(x) \in [\tau_{\text{low}}, \tau_{\text{high}}]$ are more likely to preserve semantics (as measured by behavioral invariance) than texts outside this range. These thresholds describe predictability under M, not universal semantic quality.

Tau-filtered preservation: We say $x \sim_{\tau,M}^{(\varepsilon)} k(x)$ if $h_M(x)$ lies within the calibrated regime and

$$x \approx_M^{(\varepsilon)} k(x)$$

for the relevant evaluations.

Compression-based metrics such as $\text{NCD}_{C_M^*}$ are auxiliary evidence, not primary semantic definitions.

Operational approach:

1. Calibrate $\tau_{\text{low}}, \tau_{\text{high}}$ from empirical percentiles of h_M on mixed corpora.
2. Measure $\Delta_M(x) = e(x) - |z_M(x)|$ to estimate model predictive advantage beyond generic compression.
3. Measure $\text{NCD}_{C_M^*}(x, k(x))$ and task-level output divergence as complementary semantic-preservation signals.
4. Quantify stabilization by an idempotence residual such as

$$r(x) := d_M(k(x), k(k(x))),$$

   where d_M is a model-relative behavioral distance.
5. Validate across diverse domains (encyclopedic text, scientific text, legal text, multilingual text, and code) to test cross-domain robustness.

2.7 Semantic Utilities and Utility-Conditioned Semantics (to be developed)

2.8 Caveman Compression Example (to be developed)

3 Conclusion

Semantic compression in language models has two fundamental layers: a model-relative predictability regime that may correlate with semantic coherence under calibrated domains and utilities, and a utility-indexed layer where compression directions form a partially ordered family. This dual-layer view supersedes single-chain abstraction hierarchies and provides empirically testable quality criteria grounded in information theory and behavioral semantic invariance. Future work includes learning utility spaces, adaptive utility selection, and alignment via entropy calibration.

References

[1] Fabrice Bellard. Nncp v2. `https://bellard.org/nncp/nncp_v2.pdf`, 2024.

[2] Julius Brussee. caveman: why use many token when few token do trick. `https://github.com/JuliusBrussee/caveman`, 2026. GitHub repository.

[3] Rudi Cilibrasi and Paul Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, 2005.

[4] Ming Li, Xin Chen, Xin Li, Bin Ma, and Paul Vitányi. The similarity metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004.

[5] Yauhen Yakimovich. kolmi: Experimental framework for semantic compression and neural information distance. `https://github.com/ewiger/kolmi`, 2026. GitHub repository accompanying the paper "Making Sense from LLM Outputs".