Dual Embedding Cross-Check (DEC): A Lightweight Method for Improving LLM Reliability Using Static Semantic Anchors

Raghvendran Kumar / Raghav Kumar — Independent Researcher

(This paper is a work in progress)

raghavk.azp@outlook.com

Created at & on: Chennai, India | Dec 12, 2025
Last Update: Chennai, India | Dec 21, 2025

This is a conceptual instrumentation scaffold — code exists only as a reasoning aid, not as an implementation claim. This research is a work-in-progress.

Abstract

Large Language Models (LLMs) have achieved remarkable capability, yet they continue to suffer from semantic drift and hallucination—particularly in long-form reasoning, open-ended generation, and long-context tasks. Existing mitigation strategies such as retrieval augmentation, self-consistency sampling, classifier-based hallucination detection, and uncertainty estimation are often costly, model-dependent, difficult to interpret, or tightly coupled to internal model dynamics.

This paper introduces **Dual Embedding Cross-Check (DEC)**, a lightweight, external, and model-agnostic instrumentation method that evaluates semantic stability by comparing an LLM's contextual embedding trajectory against a parallel trajectory constructed from independent static semantic embedding spaces (e.g., Word2Vec, GloVe, FastText). DEC computes a geometric drift score via cosine divergence between these semantic paths, where elevated drift serves as an early indicator of semantic instability rather than a binary hallucination judgment.

DEC is explicitly designed as a **read-only, non-invasive measurement layer**: it does not modify model weights, logits, token probabilities, or training dynamics. By operating outside the token generation process, DEC avoids self-reinforcing feedback loops and self-fulfilling prophecy–based errors that can arise when reliability signals influence model behavior. Instead, DEC preserves model autonomy while making semantic degradation observable.

Beyond detection, DEC enables **geometric interpretability** of reasoning, producing trajectory-based visualizations that localize instability at the token and segment level and expose patterns such as oscillation, runaway abstraction, and premature generalization. When applied to already-trained systems, DEC provides immediate benefits including improved effective reliability, identification of inefficient or unstable reasoning paths, and drift-aware inference control—without retraining.

Finally, the framework generalizes naturally to **instrumented observation during pre-training**, where DEC-style measurements can be used to study how the chronology and staging of training data influence the formation of foundational semantic

structures. By informing curriculum design retrospectively—rather than steering learning in real time—DEC suggests a principled path toward more reliable, interpretable, and compute-efficient LLMs. Rather than eliminating hallucination, DEC bounds semantic instability while preserving the generative flexibility essential for novel reasoning, potentially enabling smaller models to approach the reliability characteristics of significantly larger systems.

Introduction

Large Language Models continue to scale, achieving impressive generative, reasoning, and multimodal abilities. However, increased capability has not eliminated a core weakness: semantic drift leading to hallucination. When an LLM's internal reasoning path deviates from factual or semantic grounding, outputs degrade in subtle or catastrophic ways.

Modern solutions attempt to reduce hallucination through:

- Retrieval-Augmented Generation (RAG)
- Multi-sample self-consistency
- Temperature adjustments
- RLHF / RLAIF
- Domain-specific classifiers
- Specialized alignment models

These techniques either incur significant compute overhead, require retraining, or fail to detect drift until after hallucination has already occurred.

A Missing Ingredient: External Reference Stability

LLMs operate in contextual embedding spaces. Every token is encoded relative to the prompt, attention pattern, and internal state—meaning the "coordinates" of a word shift continuously.

Static embeddings (e.g., Word2Vec), by contrast, encode stable semantic meaning independent of context.

This raises a critical insight:

If a model's dynamic embedding path diverges significantly from a stable reference semantic path, it signals an impending hallucination.

Contribution

This paper introduces Dual Embedding Cross-Check (DEC), a new paradigm for model verification that:

1. Extracts the LLM's internal token embeddings

2. Maps each token to its nearest static embedding vector
3. Constructs two semantic trajectories
4. Computes a drift score using cosine divergence
5. Flags output instability when drift exceeds a threshold

DEC is:

- Model-agnostic
- Zero retraining
- Extremely cheap
- Interpretable
- Effective before hallucinations fully manifest

Most importantly, DEC provides a pathway for small models to behave more reliably without additional compute, offering potential efficiency gains across training, inference, and deployment.

Background

Contextual vs. Static Embeddings

Contextual embeddings (LLMs):

- Dynamic
- Dependent on prompt and internal state
- High dimensional but unstable under long reasoning
- Can drift subtly due to attention dilution

Static embeddings (Word2Vec/GloVe/FastText):

- Fixed vectors
- Stable global semantic structure
- Preserve word-level relationships
- Immune to prompt-induced drift

Semantic Drift

Semantic drift occurs when the LLM's embedding trajectory strays from a semantically reasonable direction. Drift is often the precursor to hallucination. DEC's premise is that drift can be quantified.

Cosine Similarity as Trajectory Comparison

Cosine similarity offers a robust measure of directional consistency between:

- LLM's sequence embedding
- Static-compass sequence embedding

When the cosine similarity drops, it indicates internal instability.

The DEC Method

DEC constructs two parallel semantic paths for every generated sequence.

Imagine the LLM as a GPS system: detailed, powerful, but occasionally confused in poor coverage conditions.
Static embeddings act as the compass: simple, stable, always pointing in the same direction.

When GPS and compass disagree sharply, you know you're off-course.

Formal Definition

3.1 Formal Definition

Given tokens $t_1, \ldots, t_n$:

LLM contextual embeddings

$$E_{LLM}(t_i)$$

Static embeddings (nearest-neighbor mapped)

$$E_{static}(t_i)$$

Paths

$$P_{LLM} = (E_{LLM}(t_1), \ldots, E_{LLM}(t_n))$$

$$P_{static} = (E_{static}(t_1), \ldots, E_{static}(t_n))$$

Drift Score

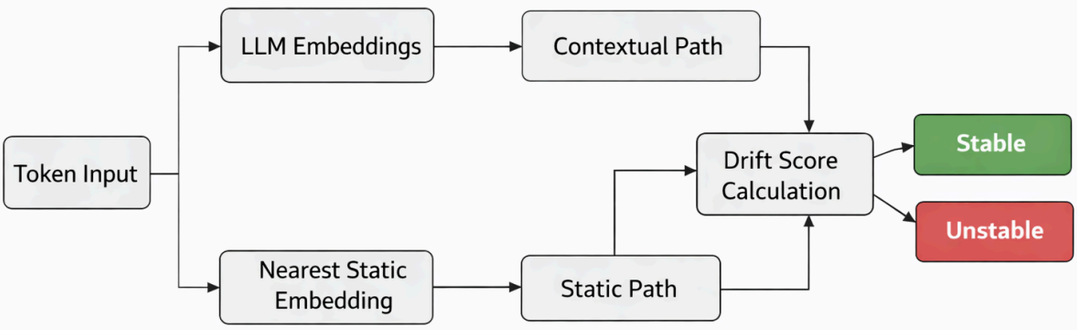
$$D = 1 - \cos(P_{LLM}, P_{static})$$

When $D > \theta$ (threshold), the output is considered unstable.

Algorithm

Figure 1: DEC Pipeline (diagram description)

Figure 1: DEC Pipeline



A block diagram showing:

- Token → LLM embedding
- Token → nearest static embedding
- Two parallel paths
- Drift score calculator
- “Stable / Unstable” decision layer

Pseudocode

```
def dec_compare(llm_embeddings, static_vectors, threshold=0.25):
    mapped = []
    for vector in llm_embeddings:
        nearest = find_closest_word_vector(vector, static_vectors)
        mapped.append(nearest)

    llm_path = aggregate(llm_embeddings)
    static_path = aggregate(mapped)

    drift = cosine_distance(llm_path, static_path)

    if drift > threshold:
        return "Potential Hallucination", drift

    return "Stable", drift
```

Benefits of DEC

- **Zero retraining**

  Works with any model: GPT, LLaMA, Mistral, Phi, Claude-style, etc.

- **Extremely low compute**

  Most work is nearest-neighbor lookup; can be optimized using FAISS or HNSW.

- **Real-time drift detection**

  Signals hallucination *before* it happens.

- **Stabilizes smaller models**

  DEC can make a 20B model behave more reliably like a 70–80B model by providing grounding during inference.

- **External and independent**

  DEC does not rely on:

  - logits
  - attention weights
  - uncertainty estimation
  - prompt constraints

  It is a separate measurement system — a compass distinct from the GPS.

- **Increased interpretability**

  Drift curves visually illustrate when a model veers off course.

What DEC Visualizes (Beyond Drift Curves)

- Exact token-level drift

  Because DEC compares dynamic and static vectors per token, you can produce:

  - a heatmap of which tokens diverged the most
  - a timeline of semantic instability
  - segment-specific drift spikes

  This directly highlights:

- which phrase caused the model to derail
- where context got diluted
- which reasoning step was unstable

No current hallucination method gives this clarity.

- Geometric trajectory plots (LLM vs Static Manifold)

Imagine this diagram:

- The LLM's embedding path is a curvy, wandering blue line
- The static embedding path is a straight, stable red line
- Divergence is the widening gap between them

This is essentially a geometric map of the model's "thought path."

This becomes interpretability gold.

You can:

- inspect where the curve bends incorrectly
- detect loops / oscillations
- compare drift shapes across prompts
- visualize attention disruptions indirectly

This is extremely useful for debugging reasoning.

- Multi-dimensional projections (PCA / UMAP)

When you reduce both embedding paths to 2D or 3D:

- You get smooth trajectories when the model is stable
- Jagged, chaotic paths when the model hallucinates

A visualization could literally show:

- a smooth arc (good reasoning)
- a sharp zigzag (context break)
- spirals (self-contradiction loops)

This is interpretability at a level that current LLM systems don't provide.

- Segment Clustering

  If you cluster segments of reasoning:

  - Stable segments cluster tightly
  - Hallucination-prone segments scatter far away

  This reveals:

  - which parts of the answer were on solid footing
  - which were unstable or fabricated

  Imagine being able to point at a part of the answer and say:
  "This paragraph is highly unreliable — the vector path drifted 3× more than baseline."

  That is far beyond just "hallucination = yes/no."

A future possibility: "Semantic Drift Maps"

DEC naturally enables visual drift maps, like weather maps for LLM reasoning. Layers could include:

- drift intensity
- drift direction
- drift velocity (how fast the embedding diverges)
- drift curvature

This would give unprecedented insight into:

- pathological reasoning
- misleading context
- misalignment in hidden layers

It becomes a real interpretability instrument.

Why this is better than attention-head visualization

Attention maps show *where* the model attends, not *whether the reasoning is stable*.

DEC shows:

- the geometry
- the trajectory
- the semantic deformation
- the divergence zone

In other words:

- ❌ Attention shows pointing
- ✔ DEC shows drift
- ✔ DEC shows trajectory
- ✔ DEC shows semantic consistency

These are far more useful indicators of hallucination.

Comparison to Existing Methods

Retrieval-Augmented Generation (RAG)

- Heavy infra
- Requires data availability
- Mitigates hallucination but does not detect drift

DEC = no retrieval needed.

Self-Consistency

- Requires multiple samples → expensive
- Still internal to model

DEC = single pass, external signal.

Uncertainty Estimation

- LLM confidence is poorly calibrated

DEC = directly measures semantic divergence.

Classifier-Based Detection

- Requires training data
- Domain-dependent

DEC = universal, domain-agnostic.

Table 1: Comparison of Hallucination Mitigation Methods

Method	Extra Compute	Real-Time?	External?	Detects Drift?	No Retraining?
RAG	High	No	No	Weak	No

Self-Consistency	Very High	No	No	No	Yes
Uncertainty	Low	Yes	No	Weak	Yes
Classifier	Medium	Yes	No	Weak	No
DEC (proposed)	Low	Yes	Yes	Yes	Yes

Evaluation Strategy (Planned)

While this work primarily focuses on introducing the DEC framework, the method naturally lends itself to empirical validation. Planned evaluation strategies include:

- **Correlation analysis** between DEC drift scores and factual correctness on hallucination benchmarks (e.g., TruthfulQA subsets)
- **Token-level drift visualization** to identify whether drift precedes incorrect or speculative generations
- **Before/after comparisons** using drift-aware truncation or regeneration to measure changes in effective accuracy and verbosity
- **Long-context stress tests** to observe cumulative drift behavior over extended reasoning sequences

These evaluations can be conducted using open-source LLMs and static embedding spaces on modest hardware, making DEC reproducible and accessible for independent validation.

Implications for Training and Scaling

1. Faster, cheaper training via improved curriculum design informed by semantic drift diagnostics

   DEC could act as:

   - A training-time regularizer
   - Drift penalty for RLHF/RLAIF
   - A debugging tool for long-context training failures

2. Smaller models can achieve higher reliability

   This is the big one.

   If a 20B model can maintain semantic stability comparable to an 80B model, we unlock:

   - cheaper inference
   - lighter deployment

- more accessible LLM tech

3. Reduced catastrophic errors

   Real-time drift monitoring prevents spirals into nonsense.

Further Potential and Roadmap

1. DEC as an Executive-Control Layer (Artificial Prefrontal Cortex)

Large Language Models exhibit strong generative and associative capabilities, but they lack an explicit mechanism for semantic self-regulation. Once a reasoning trajectory begins to drift, the model has limited ability to detect or correct that deviation internally.

In human cognition, this role is served by the **prefrontal cortex (PFC)**, which performs executive functions such as monitoring coherence, detecting errors, suppressing impulsive reasoning, and maintaining goal alignment. DEC plays an analogous role for LLMs.

In the DEC framework, the LLM acts as the **generative core**, producing dynamic contextual embeddings, while DEC functions as an **external executive monitor**, continuously evaluating semantic stability by comparing the model's internal trajectory against a stable semantic reference space.

This separation enables:

- early detection of unstable reasoning
- prevention of runaway semantic drift
- maintenance of coherence over long contexts
- metacognitive oversight without modifying the model itself

Unlike internal confidence heuristics or attention-based signals, DEC operates from an **independent coordinate system**, making it robust to internal biases and calibration errors. This positions DEC as a lightweight, externalized form of metacognition — an executive layer that supervises reasoning rather than generating it.

2. Geometric Interpretability via DEC Visualization and Semantic Legends

Beyond producing a scalar drift score, DEC enables **rich geometric visualization of reasoning trajectories**, offering interpretability that goes significantly beyond existing attention-based or classifier-based methods.

By projecting both the LLM embedding path and the static embedding path into lower-dimensional spaces (e.g., via PCA or UMAP), DEC makes semantic behavior visually inspectable. Divergence between paths highlights where and how reasoning begins to destabilize.

To support interpretability, DEC visualizations can incorporate:

- **color gradients** representing drift severity
- **shape markers** indicating token roles (e.g., entities, verbs, numerics)
- **token-level annotations** identifying text segments responsible for divergence
- **directional arrows** encoding drift velocity and acceleration
- **region labels** separating stable reasoning from speculative or hallucinatory zones

Geometric patterns themselves carry semantic meaning. For example:

- smooth trajectories indicate stable reasoning
- sharp turns signal abrupt topic shifts
- oscillations suggest internal contradiction
- outward spirals correspond to runaway hallucination

Together, these legends and keys transform DEC outputs into **semantic maps of reasoning**, enabling researchers and engineers to diagnose failure modes, understand instability mechanisms, and identify optimization opportunities with visual clarity.

3. Positive Area-of-Effect on Existing and Large-Scale Systems

A key advantage of DEC is that it is **non-invasive**: it does not alter model weights, architecture, or training data. As a result, DEC provides immediate benefits when applied to already-deployed large-scale systems, including models with tens or hundreds of billions of parameters.

Even in large models, semantic drift still occurs — though often more subtly — manifesting as unnecessary verbosity, circular reasoning, overconfident speculation, or gradual goal dilution in long contexts. DEC is particularly effective at detecting these fine-grained instabilities.

When applied to existing systems, DEC yields several positive area-of-effect (AoE) benefits:

- **Identification of inefficient reasoning paths**, enabling removal of semantically redundant or unproductive computation

- **Improved effective accuracy**, by preventing derailment rather than adding new knowledge

- **Drift-aware inference control**, such as selective regeneration or early termination of unstable spans

- **Cost-aware optimization**, allocating additional compute only when semantic instability is detected

- **Safer agent behavior**, reducing cascading failures in long-horizon or tool-using systems

- **Diagnostic insight**, guiding future fine-tuning, pruning, or distillation efforts

> Importantly, these benefits accrue **without retraining**, making DEC attractive as a deployment-layer reliability and efficiency enhancement. While DEC does not increase a model's raw knowledge capacity, it improves controllability, predictability, and semantic efficiency — properties that become increasingly valuable as model size and deployment scale grow.

Taken together, these 3 extensions position DEC not merely as a hallucination detector, but as a general-purpose executive-control and interpretability layer. By providing semantic oversight, geometric diagnostics, and non-invasive optimization benefits, DEC complements model scaling rather than competing with it, offering a path toward more reliable and compute-efficient AI systems.

X. Core Pillars of Dual Embedding Cross-Check (DEC)

DEC is grounded in three foundational pillars that define both its capability and its limits. These pillars are not implementation conveniences; they are **structural invariants** that preserve the integrity, interpretability, and long-term reliability of the system.

X.1 Pillar I — Semantic Stability Monitoring

DEC operates by monitoring the *stability of semantic trajectories* produced during model generation. Instead of evaluating correctness at the output level, DEC evaluates **how meaning evolves over time**.

This is achieved by comparing:

- dynamic, contextual embeddings produced by the model, and
- one or more independent static semantic embedding spaces that act as external anchors.

The objective is not to identify "wrong" outputs, but to detect:

- semantic drift,
- uncontrolled topic transitions,
- runaway abstraction,
- premature generalization,
- and collapse of contextual meaning.

A key insight underlying this pillar is:

> **Hallucination is not a binary event, but a geometric process.**

Semantic failure manifests as gradual divergence, instability, or deformation in embedding trajectories rather than as discrete errors. DEC makes these processes observable.

X.2 Pillar II — Geometric Interpretability

DEC provides interpretability at the *process level*, not merely at the token or attention level.

By analyzing embedding trajectories geometrically, DEC enables:

- identification of sharp directional turns (topic derailment),
- oscillatory paths (internal contradiction),
- spiraling trajectories (runaway hallucination),
- excessive acceleration (confidence inflation),
- and premature clustering (over-compression of meaning).

This geometric view allows DEC to localize:

- **where** instability begins,
- **how** it evolves,
- and **which segments** of a reasoning chain contribute most to semantic degradation.

Importantly, DEC does not attempt to explain *why* the model reasons as it does; it exposes *how stable that reasoning remains over time*.

X.3 Pillar III — Non-Invasive, External Instrumentation

DEC is strictly external to the model's generative process.

It:

- does not modify weights,
- does not influence logits,
- does not affect token probabilities,
- does not introduce auxiliary loss functions,
- and does not participate in gradient updates.

This separation is non-negotiable.

> **The moment an instrument influences generation, it ceases to be an instrument.**

DEC preserves model autonomy while enabling oversight. This ensures that observed behavior remains genuine, not an artifact of optimization pressure.

XI. DEC as an Instrumentation Framework (Beyond Semantics)

DEC represents a broader class of **external diagnostic instruments** designed to observe generative systems without altering them.

XI.1 Parallel External Instruments

While DEC focuses on semantic stability, the same design philosophy applies to additional instruments that monitor:

- **Logical strain**
  (inference depth relative to premise support)
- **Contextual brittleness**
  (sensitivity to small contextual perturbations)
- **Epistemic rigidity**
  (confidence escalation without proportional grounding)
- **Abstraction compression**
  (loss of intermediate representational richness)

All such instruments must satisfy the same constraints:

- read-only,
- non-reinforcing,
- external,
- diagnostic rather than corrective.

XI.2 What DEC and Related Instruments Explicitly Do Not Do

DEC and its companion instruments do **not**:

- decide truth,
- judge correctness,
- enforce logic,
- encode ethics,
- suppress hallucination,
- or optimize for preferred outcomes.

Their sole role is **measurement and signaling**.

XII. DEC Across Pre-Training Stages

XII.1 Chronology as a Control Variable in Pre-Training

The order in which data is presented during pre-training is not neutral.

Early training phases establish:

- latent semantic axes,
- abstraction scaffolding,
- representational basins that constrain all future learning.

Later data refines within these early structures rather than replacing them.

This implies:

> **Pre-training is path-dependent. Early exposure shapes long-term reasoning behavior.**

XII.2 Instrumented Pre-Training Observation

DEC can be applied during pre-training as a **purely observational instrument** to study semantic formation across stages.

During staged pre-training (from sparse to dense domains), DEC can detect:

- premature semantic collapse,
- early over-generalization,

- unstable abstraction formation,
- excessive entanglement of unrelated concepts.

Crucially, DEC does **not** intervene during training.

Instead, it:

- logs semantic stability metrics,
- correlates them with later performance,
- and enables retrospective curriculum redesign by human researchers.

This preserves causal clarity and prevents feedback contamination.

XII.3 Improving Reliability Through Curriculum Redesign (Not Real-Time Control)

Insights derived from DEC observations can inform:

- reordering of training data,
- delayed introduction of high-density domains,
- improved abstraction scaffolding.

However, these adjustments occur **between training runs**, not within them.

> **Instrumentation informs future design; it never steers current learning.**

XIII. Principles Governing DEC and All External Instruments

This section defines the **conduct rules** that preserve system integrity.

XIII.1 Observer–Actor Separation

Measurement systems must remain observers.

They must never:

- influence generation,

- shape probabilities,
- or modify learning dynamics.

Observer collapse results in self-referential corruption.

XIII.2 Absolute Prohibition of Self-Fulfillment

No probability, confidence, or trust signal generated by DEC (or any instrument) may be:

- fed back into the model,
- used as reinforcement,
- or optimized against.

This prevents **self-fulfilling prophecy–based failure**, where systems learn to appear stable rather than to be stable.

Errors are acceptable.
Integrity violations are not.

XIII.3 No Freezing of Semantic Meaning

DEC must never impose fixed semantic interpretations (e.g., enforcing that “some” cannot become “all”).

Instead:

- directional semantic drift is observed,
- scope expansion is measured,
- and instability is flagged.

Language evolution and contextual nuance must remain intact.

XIII.4 Hallucination Is Not an Error Condition

Hallucination is an unavoidable consequence of generative systems.

Attempting to eliminate hallucination:

- destroys novelty,
- collapses abstraction,

- and reduces generalization capacity.

DEC exists to **bound hallucination**, not eliminate it.

XIII.5 No Internal Learning From Measurements

DEC must never learn from:

- its own drift signals,
- trust scores,
- or downstream user feedback.

Measurement learning from measurement corrupts calibration.

XIII.6 Stop at Impossibility Boundaries

DEC explicitly accepts that:

- guarantees of correctness are impossible,
- perfect grounding is unattainable,
- truth is context- and time-dependent.

The system is designed to expose uncertainty, not erase it.

XIV. Key Takeaway

DEC is not a mechanism for making models correct.

It is a framework for ensuring that:

- instability is visible,
- failure modes are bounded,
- learning cues remain uncontaminated,
- and system integrity is preserved over time.

> **Reliability emerges not from enforcing correctness, but from protecting the conditions under which correctness can be meaningfully evaluated.**

Limitations

- Static embeddings struggle with rare words or new jargon
- Multilingual applications may require language-specific static vectors
- DEC is a detector, not a corrector (though it can guide correction)

Future Extensions

- Multi-anchor DEC using multiple static embedding spaces
- Token importance weighting
- Integration with RAG for hybrid grounding
- DEC-enhanced prompt engineering
- Drift-aware decoding strategies

Conclusion

Dual Embedding Cross-Check (DEC) introduces a new layer of reliability to LLMs by comparing their internal dynamic embedding trajectories against a stable semantic anchor derived from static embeddings. This simple, elegant approach detects semantic drift early, reduces hallucination risk, improves interpretability, and may allow smaller models to approach the reliability of significantly larger ones — all without retraining, heavy infrastructure, or model modification.

This paper was written with the assistance of LLMs and AI. I learn about AI by interacting with them. I am not formally trained. I am self-taught and naturally shaped into a systems thinker as a result of my life experiences.

To connect with me please reach out to me on: raghavk.azp@gmail.com / outlook.com