Semantic Regression Testing for Natural-Language Knowledge Bases

Zeel Patel
Independent
patelzeel68@yahoo.in

Abstract

Contradiction detection in natural language is usually treated as a sentence-pair classification task. A model receives a premise and a hypothesis, then predicts entailment, contradiction, or neutrality. This setting is useful, but it does not match how contradictions appear in real knowledge systems. Companies maintain changing collections of help pages, policy documents, product documentation, API references, support macros, release notes, and retrieval sources for AI agents. A failure often occurs when a new document changes a claim in one place while older claims remain active elsewhere.

This paper proposes Semantic Regression Testing, a framework for detecting meaning-level regressions in versioned natural-language knowledge bases. The framework represents documents as scoped, versioned claims; compares new claims against active claims from earlier versions; measures the increase in contradiction risk; and estimates the minimum repair cost required to restore consistency. We define two main quantities: Semantic Regression Loss, which measures how much an update makes a knowledge base less coherent, and Minimal Repair Cost, which estimates the cheapest set of grounded edits needed to make the corpus safe to release. The framework is designed for documentation review, support knowledge bases, compliance workflows, product release review, and retrieval-augmented generation systems.

1 Introduction

Contradiction detection has usually been studied through Natural Language Inference (NLI). In that setup, a system receives a premise and a hypothesis and predicts whether the hypothesis is entailed by the premise, contradicts it, or is neutral with respect to it [1, 2]. This framing has been useful for model development. It gives a clean input, a small label set, and a simple evaluation target.

The real setting is messier. A company does not usually contradict itself through a single isolated sentence pair. It contradicts itself through drift. A product page changes while the help center keeps the old rule. A legal document updates while a support macro still gives the previous answer. An API field is deprecated while an AI agent's retrieval corpus still describes the field as active. A launch post says a feature is available to all users, while the documentation says it is available only in one region.

This paper studies that failure mode.

We define a *semantic regression* as a change to a natural-language knowledge base that makes the system less coherent, less current, or less safe to use. The central question is not only whether two sentences contradict each other. The central question is:

> Given an older knowledge base $\mathcal{K}_t$ and an updated knowledge base $\mathcal{K}_{t+1}$, did the update introduce a meaning-level regression?

This reframes contradiction detection as a regression testing problem. In software, a change is not judged only by whether it compiles. It is tested against expected behavior. We apply the

same idea to natural language. A document update should be tested against the active claims that an organization already treats as true.

The proposed framework, Semantic Regression Testing, has four main steps. First, it extracts atomic claims from documents. Second, it assigns each claim a scope, version, modality, and risk level. Third, it compares new claims against active older claims and truth invariants. Fourth, it computes a release decision: pass, warn, or block.

1.1 Contributions

This paper makes four contributions.

1. It defines semantic regression for versioned natural-language knowledge bases.
2. It introduces a scoped claim graph representation for active organizational claims.
3. It defines Semantic Regression Loss and Minimal Repair Cost as measurable quantities.
4. It proposes an evaluation setup for testing whether semantic regression testing improves document quality and reduces inconsistent answers in downstream AI systems.

2 Related Work

2.1 Natural Language Inference

NLI studies the relation between a premise and a hypothesis. Datasets such as SNLI and MultiNLI helped standardize the three-way classification format of entailment, contradiction, and neutral [1, 2]. This framing is useful, but it treats each sentence pair as a separate example. It does not directly model document status, version history, source authority, or operational risk.

2.2 Shortcut Learning in NLI

Prior work has shown that NLI models can learn shallow heuristics. HANS demonstrated that models can perform well on standard benchmarks while relying on lexical and syntactic shortcuts that do not reflect true reasoning [3]. This is important for contradiction detection because many real contradictions depend on time, scope, modality, or source status. Surface disagreement is not enough.

2.3 Contradiction Detection

Earlier contradiction detection work studied how textual contradictions can be identified and categorized [4]. More recent work has moved toward long-document contradiction detection. ContraDoc studies self-contradictions in documents and evaluates how large language models handle them [6]. This is close to the problem studied here, but our focus is different. We study contradiction as a regression introduced by a knowledge-base update.

2.4 Factuality and Claim Verification

Factuality work decomposes long generated text into smaller claims and checks whether those claims are supported by evidence. FActScore is one example of this approach [5]. Our framework also uses atomic claims, but the objective is different. We do not only ask whether a claim is supported. We ask whether a new claim causes a regression relative to an active knowledge base.

2.5 Positioning

The difference between contradiction detection and semantic regression testing is the unit of analysis. Contradiction detection asks whether two statements disagree. Semantic regression testing asks whether a changing knowledge base has become less safe to believe.

3 Problem Formulation

Let a natural-language knowledge base at time t be written as

$$\mathcal{K}_t = \{d_1^t, d_2^t, \ldots, d_n^t\},$$

where each d_i^t is a document, policy, product page, support article, API reference, release note, or retrieval source.

After an update, the knowledge base becomes

$$\mathcal{K}_{t+1} = \{d_1^{t+1}, d_2^{t+1}, \ldots, d_m^{t+1}\}.$$

Problem 1 (Semantic Regression Detection)**.** *Given an older knowledge base $\mathcal{K}_t$ and an updated knowledge base $\mathcal{K}_{t+1}$, determine whether the transition from $\mathcal{K}_t$ to $\mathcal{K}_{t+1}$ introduces a semantic regression.*

A semantic regression may occur when a new or reactivated claim contradicts an active older claim, violates a truth invariant, revives stale information, or introduces unresolved ambiguity into a high-risk part of the corpus.

4 Claim Representation

Definition 1 (Claim)**.** *A claim is an atomic proposition extracted from a document. The set of claims extracted from $\mathcal{K}_t$ is*

$$\mathcal{C}_t = \{c_1^t, c_2^t, \ldots, c_N^t\}.$$

Each claim is represented as

$$c_i = (p_i, s_i, v_i, \tau_i, \mu_i, \rho_i, \epsilon_i),$$

where p_i is the normalized proposition, s_i is the source span, v_i is the document version, τ_i is the scope, μ_i is the modality, ρ_i is the risk level, and $\epsilon_i \in [0, 1]$ is the extraction confidence.

The scope vector is written as

$$\tau_i = (e_i, r_i, a_i, g_i, u_i, h_i, z_i),$$

where e_i is the entity, r_i is the region, a_i is the asset or object, g_i is the product or system, u_i is the user segment, h_i is the time interval, and z_i is the condition under which the claim holds.

Definition 2 (Active Claim)**.** *A claim c_i is active if it is not deprecated, archived, superseded, or limited to a past version. We define an activity function*

$$\alpha(c_i) \in \{0, 1\},$$

where $\alpha(c_i) = 1$ means the claim is active and $\alpha(c_i) = 0$ means it should not be used as a current truth constraint.

The active claim set is

$$\mathcal{A}_t = \{c_i \in \mathcal{C}_t : \alpha(c_i) = 1\}.$$

5 Semantic Compatibility

Two claims should only be compared if they describe the same scoped setting. For example, the claim "withdrawals are instant on Base" does not necessarily contradict the claim "withdrawals take three days on Arbitrum" because the chain scope differs.

We define a scope alignment function

$$A(c_i, c_j) : \mathcal{C} \times \mathcal{C} \to [0, 1].$$

A value of $A(c_i, c_j) = 1$ means the claims share the same relevant scope. A value of $A(c_i, c_j) = 0$ means they should not be treated as contradictory.

For a decomposed scope vector, alignment can be computed as

$$A(c_i, c_j) = \prod_{k=1}^{K} a_k(\tau_{i,k}, \tau_{j,k}),$$

where a_k is the alignment score for scope field k.

We also define a semantic compatibility function

$$B(c_i, c_j) : \mathcal{C} \times \mathcal{C} \to [-1, 1],$$

where

$$B(c_i, c_j) = \begin{cases} 1, & \text{if } c_i \text{ and } c_j \text{ support each other,} \\ 0, & \text{if their relation is neutral or unknown,} \\ -1, & \text{if } c_i \text{ and } c_j \text{ contradict each other.} \end{cases}$$

The pairwise contradiction weight is

$$q_{ij} = \alpha(c_i)\alpha(c_j) \cdot A(c_i, c_j) \cdot \max(0, -B(c_i, c_j)) \cdot R(c_i, c_j),$$

where $R(c_i, c_j)$ is a risk multiplier. This gives higher weight to contradictions involving legal, financial, medical, security, or user-facing operational statements.

6 Claim Graph

We represent a knowledge base as a weighted claim graph

$$G_t = (\mathcal{A}_t, E_t),$$

where each node is an active claim and each edge weight is q_{ij}.

The contradiction energy of the knowledge base is

$$E(G_t) = \sum_{i<j} q_{ij}.$$

A higher value of $E(G_t)$ means the active knowledge base contains more contradiction risk.

7 Truth Invariants

Some claims are stronger than ordinary document claims. We call these truth invariants.

Let

$$\mathcal{I}_t = \{\iota_1, \iota_2, \ldots, \iota_M\}$$

be the set of truth invariants active at time t. Each invariant is represented as

$$\iota_k = (p_k, \tau_k, \rho_k, \pi_k),$$

where p_k is the invariant proposition, τ_k is its scope, ρ_k is its risk level, and π_k is the policy attached to it.

The invariant violation score is

$$V(\mathcal{K}_{t+1}, \mathcal{I}_t) = \sum_{\iota_k \in \mathcal{I}_t} \rho_k \cdot \mathbb{I}\left[\mathcal{K}_{t+1} \not\models \iota_k\right],$$

where $\mathbb{I}[\cdot]$ is the indicator function and $\mathcal{K}_{t+1} \not\models \iota_k$ means that the updated knowledge base violates invariant ι_k.

8 Semantic Regression Loss

We define semantic regression as an increase in contradiction risk after an update. The Semantic Regression Loss is

$$\mathrm{SRL}(\mathcal{K}_{t+1} \mid \mathcal{K}_t) = [E(G_{t+1}) - E(G_t)]_+ + \lambda V(\mathcal{K}_{t+1}, \mathcal{I}_t) + \gamma S(\mathcal{K}_{t+1}) + \eta U(\mathcal{K}_{t+1}),$$

where

$$[x]_+ = \max(0, x).$$

The term V measures truth invariant violations. The term S measures staleness. The term U measures unresolved ambiguity. The constants λ, γ, and η control the importance of each term.

The staleness term can be written as

$$S(\mathcal{K}_{t+1}) = \sum_{c_i \in \mathcal{A}_{t+1}} \Delta(c_i) \cdot \sigma(c_i),$$

where $\Delta(c_i)$ is the age of the claim and $\sigma(c_i)$ is the time sensitivity of the claim.

The unresolved ambiguity term is

$$U(\mathcal{K}_{t+1}) = \sum_{i<j} \alpha(c_i)\alpha(c_j) \cdot A(c_i, c_j) \cdot H(B(c_i, c_j)),$$

where $H(B(c_i, c_j))$ measures uncertainty in the compatibility estimate.

9 Minimal Conflict Clusters

A conflict cluster is a set of active claims whose joint presence creates a contradiction.

Let $\mathcal{M} \subseteq \mathcal{A}_{t+1}$. We call $\mathcal{M}$ a minimal conflict cluster if

$$E(G[\mathcal{M}]) > 0$$

and for every strict subset $\mathcal{M}' \subset \mathcal{M}$,

$$E(G[\mathcal{M}']) = 0.$$

Here, $G[\mathcal{M}]$ denotes the subgraph induced by $\mathcal{M}$.

Minimal conflict clusters are important because they identify the smallest set of claims responsible for a regression.

10 Minimal Repair Cost

A useful system should not only detect a regression. It should also explain how to repair it.

Let

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_L\}$$

be the set of allowed repair operations. Examples include deprecating a claim, updating a claim, deleting a claim, adding a condition, splitting scope, weakening modality, or assigning a canonical source.

Each repair operation has a cost

$$\text{cost}(\omega_\ell) \geq 0.$$

For a set of repairs $\Omega' \subseteq \Omega$, let

$$G_{t+1} \circ \Omega'$$

denote the claim graph after applying those repairs.

We define Minimal Repair Cost as

$$\text{MRC}(G_{t+1}) = \min_{\Omega' \subseteq \Omega} \sum_{\omega \in \Omega'} \text{cost}(\omega) \quad \text{subject to} \quad \text{SRL}(G_{t+1} \circ \Omega') \leq \delta,$$

where δ is the maximum acceptable regression threshold.

11 Release Decision

The framework outputs a release decision:

$$D(\mathcal{K}_{t+1}) \in \{\texttt{pass}, \texttt{warn}, \texttt{block}\}.$$

Given thresholds θ_{low} and θ_{high}, the decision rule is

$$D(\mathcal{K}_{t+1}) = \begin{cases} \texttt{pass}, & \text{SRL} < \theta_{\text{low}}, \\ \texttt{warn}, & \theta_{\text{low}} \leq \text{SRL} < \theta_{\text{high}}, \\ \texttt{block}, & \text{SRL} \geq \theta_{\text{high}} \text{ or a hard invariant is violated.} \end{cases}$$

12 Algorithm

13 Worked Example

Consider a knowledge base for a financial product.

Claim	Source	Status
C_1: Deposits on Base and Arbitrum are eligible for weekly rewards.	Docs	Active
C_2: Only Base deposits are eligible for weekly rewards.	Launch post	Active
C_3: All deposits are eligible immediately.	Support macro	Active
C_4: Eligibility starts from the next weekly cycle.	Terms	Active

Table 1: Example of active claims in a knowledge base.

A sentence-pair contradiction model may flag C_1 against C_2 and C_3 against C_4. Semantic Regression Testing gives a richer output. It identifies the conflict cluster

$$\mathcal{M} = \{C_1, C_2, C_3, C_4\},$$

Algorithm 1 Semantic Regression Test

Require: Previous knowledge base $\mathcal{K}_t$, updated knowledge base $\mathcal{K}_{t+1}$, invariant set $\mathcal{I}_t$
Ensure: Release decision D, conflict clusters, repair suggestions

Extract claims $\mathcal{C}_t$ from $\mathcal{K}_t$
Extract claims $\mathcal{C}_{t+1}$ from $\mathcal{K}_{t+1}$
Identify active claim sets $\mathcal{A}_t$ and $\mathcal{A}_{t+1}$
Normalize claim scopes τ_i
Compute scope alignment $A(c_i, c_j)$ for relevant claim pairs
Compute semantic compatibility $B(c_i, c_j)$
Build claim graphs G_t and G_{t+1}
Compute contradiction energies $E(G_t)$ and $E(G_{t+1})$
Compute invariant violation score $V(\mathcal{K}_{t+1}, \mathcal{I}_t)$
Compute $\mathrm{SRL}(\mathcal{K}_{t+1} \mid \mathcal{K}_t)$
Find minimal conflict clusters
Estimate Minimal Repair Cost
Return $D(\mathcal{K}_{t+1})$, conflict clusters, and repair suggestions

assigns high risk because the conflict concerns user-facing reward eligibility, and returns a block decision until the sources are repaired.

A possible repair set is

$$\Omega' = \{\omega_{\text{support}}, \omega_{\text{chain}}, \omega_{\text{scope}}, \omega_{\text{canonical}}\}.$$

14 Experimental Setup

14.1 Dataset

We propose SRTBench, a dataset of versioned knowledge-base updates. Each example contains

$$(\mathcal{K}_t, \mathcal{K}_{t+1}, \mathcal{I}_t, y),$$

where y contains the gold semantic regression label, risk class, minimal conflict cluster, valid repairs, invalid repairs, and release decision.

The dataset should include the following categories:

1. pricing drift,
2. refund or withdrawal drift,
3. product availability drift,
4. API behavior drift,
5. legal versus marketing conflict,
6. support macro versus documentation conflict,
7. changelog versus current documentation conflict,
8. region-specific eligibility conflict,
9. security claim conflict,
10. retrieval-corpus conflict for AI agents,
11. old claim still marked active,
12. modality drift, such as “may” becoming “will”.

14.2 Baselines

The framework should be compared against:

1. a standard NLI classifier,
2. a transformer model fine-tuned on NLI data,
3. direct prompting of a large language model,
4. large language model prompting with explanation,
5. atomic fact checking,
6. document-level contradiction detection,
7. Semantic Regression Testing without scope normalization,
8. Semantic Regression Testing without risk weighting,
9. Semantic Regression Testing without repair scoring.

14.3 Metrics

The main metrics are:

- **Semantic Regression F1**: whether the system correctly detects regressions.
- **Invariant Violation Recall**: whether high-risk invariant violations are caught.
- **Minimal Cluster Precision**: whether returned conflict clusters are responsible for the regression.
- **Repair Accuracy**: whether suggested repairs are accepted by human annotators.
- **False Block Rate**: how often harmless updates are blocked.
- **Human Review Burden**: how many claims must be inspected per true regression.
- **Downstream Agent Consistency**: whether a retrieval-augmented assistant gives fewer inconsistent answers after the corpus is checked.

14.4 Hypotheses

We test five hypotheses.

- **H1**: Pairwise NLI models miss regressions that depend on document status, versioning, or more than two claims.
- **H2**: Scope normalization reduces false positives.
- **H3**: Risk weighting better matches human release decisions than raw contradiction count.
- **H4**: Repair scoring reduces review burden compared to detection-only systems.
- **H5**: AI assistants using a knowledge base checked by Semantic Regression Testing produce fewer inconsistent answers than assistants using an unchecked knowledge base.

15 Learning Objective

If model parameters ϕ are used to estimate claim extraction, scope alignment, compatibility, risk, or repair quality, they can be optimized against annotated data. Let the training set be

$$\mathcal{D} = \{(\mathcal{K}_t^{(n)}, \mathcal{K}_{t+1}^{(n)}, y^{(n)})\}_{n=1}^{N},$$

where $y^{(n)}$ contains the gold semantic regression label, minimal conflict cluster, risk class, valid repairs, and release decision.

A supervised objective can be written as

$$\mathcal{L}(\phi) = \mathcal{L}_{\text{reg}}(\phi) + \beta_1 \mathcal{L}_{\text{cluster}}(\phi) + \beta_2 \mathcal{L}_{\text{risk}}(\phi) + \beta_3 \mathcal{L}_{\text{repair}}(\phi).$$

Here, $\mathcal{L}_{\text{reg}}$ trains semantic regression detection, $\mathcal{L}_{\text{cluster}}$ trains conflict cluster recovery, $\mathcal{L}_{\text{risk}}$ trains risk classification, and $\mathcal{L}_{\text{repair}}$ trains repair selection.

The final objective is to minimize semantic regression while reducing unnecessary human review:

$$\min_{\phi} \mathbb{E}_{(\mathcal{K}_t, \mathcal{K}_{t+1}) \sim \mathcal{D}} \left[\text{SRL}_{\phi}(\mathcal{K}_{t+1} \mid \mathcal{K}_t) + \kappa \, \text{Review}_{\phi}(\mathcal{K}_{t+1}) \right],$$

where Review_{ϕ} is the number of claims or conflict clusters sent for human review and κ controls review cost.

16 Discussion

Semantic Regression Testing does not remove the need for human review. It makes review smaller and more targeted. In high-risk domains, the framework should be used as a pre-release check, not as a final authority.

The framework also depends on claim extraction quality. If an extractor misses an important claim, the system may miss a regression. For this reason, every claim must remain linked to its source span, and high-risk documents should require stricter extraction checks.

Risk levels are also domain-specific. A product documentation team, a legal team, and a medical support team should not use the same thresholds. The framework is general, but its release policy must be configured for the domain.

17 Conclusion

This paper reframes contradiction detection as regression testing for natural language.

The sentence-pair setup asks whether two statements contradict each other. That task is useful, but it does not match how contradictions appear in operational systems. Real contradictions often come from document drift: one source changes while another active source keeps the old claim.

Semantic Regression Testing treats a knowledge base as a versioned set of scoped claims. It checks whether new claims conflict with active older claims, whether truth invariants are violated, and whether a release should pass, warn, or be blocked. The framework defines Semantic Regression Loss to measure the added conflict caused by an update and Minimal Repair Cost to estimate how difficult it is to restore consistency.

The practical claim is simple: before an organization ships documents, support answers, or retrieval sources to users and AI agents, it should test whether its knowledge base has become less safe to believe.

References

[1] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of EMNLP*, 2015.

[2] A. Williams, N. Nangia, and S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL*, 2018.

[3] R. T. McCoy, E. Pavlick, and T. Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of ACL*, 2019.

[4] M.-C. de Marneffe, A. N. Rafferty, and C. D. Manning. Finding contradictions in text. In *Proceedings of ACL-HLT*, 2008.

[5] S. Min, K. Krishna, X. Lyu, M. Lewis, W.-t. Yih, P. W. Koh, M. Iyyer, L. Zettlemoyer, and H. Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of EMNLP*, 2023.

[6] J. Li, V. Raheja, and D. Kumar. ContraDoc: Understanding self-contradictions in documents with large language models. In *Proceedings of NAACL*, 2024.

[7] Z. D. Patel, M. Swarup, S. Sharma, and T. Mahajan. Detecting anomalies, contradictions and contextual-analysis through NLP in text. IEEE document 10064560, 2023.