Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

The Temporal Validity of Taxonomic Entailments in Evolving Biomedical Ontologies

Elena Udrescu
Valahia University of Târgoviște
Alexandru Udrescu
Valahia University of Târgoviște
Mihai Bîzoi
Valahia University of Târgoviște
Ana-Maria Suduc
ana.suduc@valahia.ro
Valahia University of Târgoviște

Research Article

Keywords: Biomedical ontologies, Ontology evolution, Ontology maintenance, OBO Foundry, Entailment survival, Subsumption, Truth maintenance, Survival analysis

Posted Date: November 4th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11294772/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

The Temporal Validity of Taxonomic Entailments in Evolving Biomedical Ontologies

Elena Udrescu[a], Alexandru Udrescu[a], Mihai Bîzoi[a], Ana-Maria Suduc[a,*]

[a] *Research Center for Automation, Computer Science, and Electrical Engineering, Valahia University of Târgoviște, 13 Aleea Sinaia Street, 130004 Târgoviște, Romania*

* Corresponding author: Ana-Maria Suduc, ana.suduc@valahia.ro

E-mail addresses: Elena Udrescu, elena.gadina.is@valahia.ro; Alexandru Udrescu, alexandru.udrescu@valahia.ro; Mihai Bîzoi, mihai.bizoi@valahia.ro; Ana-Maria Suduc, ana.suduc@valahia.ro

Abstract

Background: Biomedical knowledge-based systems, such as annotation pipelines, phenotype-driven diagnostic tools and cohort-selection and data-integration services, materialise conclusions derived from reference ontologies and act on them, yet treat the ontology as fixed although it is re-released every few weeks or months. How long such derived knowledge remains valid, and which maintenance strategies preserve its integrity, has rarely been quantified outside Gene Ontology enrichment analysis. This article quantifies both for the taxonomic fragment, treating each derived taxonomic entailment as a subject that is born when it is first derivable and dies when it no longer follows.

Results: From 311 release snapshots of eight biomedical reference ontologies of the Open Biological and Biomedical Ontology family spanning up to 16.3 years, 929,486 entailment lifetimes (623,167 incident) are reconstructed and analysed with Kaplan–Meier and stratified Cox models. Pooled five-year survival is 0.786 (95% ontology-level interval 0.708–0.886), or 0.608 counting loss of the justifying derivation; attrition is episodic and ontology-specific. Of the 128,950 hard invalidations, 71.8% arise from path

breakage in which both classes named in the conclusion remain active. Deprecation signals on the two endpoint classes detect 14.5% of invalidations, and 16.5% when every intermediate class is watched as well; watching the subclass's own parents raises this to at least 43.8%, whereas re-checking the edges of a stored justification path detects all of them. At workload level, 21.9% of 3,987 cohort-style subsumption queries lose at least one member of their answer set within three years. Volatility is largely unpredictable at derivation time: within-ontology models beat a permutation null in one of six evaluable ontologies, and cross-ontology transfer is below chance in five of seven.

Conclusions: Conclusions derived from biomedical ontologies expire at a measurable rate, and most of the failures are invisible to obsoletion-based monitoring. For taxonomic inference, consumers of these ontologies should re-derive conclusions on every release or store and re-check justification paths, rather than rely on deprecation signals, and ontology developers can help by publishing the entailments added and removed with each release. The corpus, the analysis code and the verification scripts are openly available.

Keywords: Biomedical ontologies; Ontology evolution; Ontology maintenance; OBO Foundry; Entailment survival; Subsumption; Truth maintenance; Survival analysis

1. Background

Formal ontologies are the backbone of a large class of knowledge-based systems [1]. In biomedicine, where the ontologies of the Open Biological and Biomedical Ontology (OBO) Foundry and Library [2], referred to here as OBO-family ontologies, underpin gene and phenotype annotation and cross-database integration, such systems are part of the semantic infrastructure on which research and clinical tools rely. Decision-support tools, cohort-selection systems, semantic data-integration layers, annotation pipelines and classification services all derive conclusions by reasoning over a reference ontology, and then materialise them in caches, databases, annotation stores and published results that are queried and acted on independently of the ontology that produced them. In many such systems the ontology is treated as a fixed knowledge source: a conclusion derived from one release is assumed to remain valid, and the materialised

knowledge is not re-examined when its source changes. Keeping derived knowledge consistent with an evolving source is a classical problem in knowledge representation and database research, studied as truth maintenance and view maintenance (Section 2.4); what is missing for ontology-backed systems is evidence of how quickly, and in what way, that consistency is lost in practice.

Reference ontologies are not fixed. They are actively curated artefacts, re-released on cadences ranging from a few weeks to several months, and every release brings classes that are obsoleted, merged, split, re-parented or re-axiomatised [3, 4]. The consequence is straightforward but has rarely been measured: a conclusion derived from an ontology has a limited shelf life, and after enough releases it may no longer follow. Neither the system holding it nor the user reading it is informed of this.

Gene Ontology (GO) enrichment analysis is the setting in which this shelf-life effect has been documented most thoroughly, and there it is substantial. Wadi et al. [5] showed that the vocabulary of pathways and processes roughly doubled over a seven-year window and that a majority of surveyed publications relied on annotation snapshots recovering only a fraction of the enrichments visible under current annotations. Tomczak et al. [6] showed that the biological interpretation of a fixed experiment changes materially as the Gene Ontology and its annotations evolve. Groß et al. [7], studying the evolution of the Gene Ontology and its annotations, found enrichment results stable overall but identified specific datasets in which a substantial fraction of significant categories changed. All three are studies of one ontology family and one application. To the authors' knowledge, the general question has not been addressed: for an arbitrary ontology-backed inference, what is the probability that it still holds t years later, and can that probability be estimated when the inference is drawn?

This article poses and answers that question empirically for the taxonomic fragment. The unit of longitudinal analysis is the derived taxonomic entailment, an ordered pair of classes (subclass, superclass) in the transitive closure of the asserted subsumption hierarchy. Treating such an entailment as a subject that is born when first derivable, dies when it ceases to hold, and is right-censored if it survives to the end of observation, allows survival functions to be estimated, deaths to be attributed to causes, and time-to-death

to be tested for predictability. The scope restriction is deliberate and is reflected in the title: the analysis covers the named-class taxonomic fragment as published in each release, not full description-logic entailment; Sections 3.2 and 5.5 state precisely what this excludes and in which direction it biases the results. For biomedical semantics the question is one of resource management: it asks how the routine maintenance of an ontology propagates to the annotations, query results and explanations that were derived from it.

The corpus is assembled from public, version-controlled ontology releases. Eight ontologies of the OBO family, each covering 5.4 to 16.3 years of curation history, contribute 311 release snapshots, from which 929,486 entailment lifetimes are reconstructed, 623,167 of them incident, together with 377,716 events in which the recorded justification for a still-valid entailment was superseded. To the authors' knowledge this is the first cross-ontology longitudinal corpus of entailment survival.

The study addresses five research questions. RQ1: How long does a derived taxonomic entailment remain valid? RQ2: Which curation operations cause invalidation, and which of them emit a signal that a consumer could watch? RQ3: Can the volatility of an entailment be predicted at derivation time? RQ4: Does prioritising re-verification by predicted volatility save work? RQ5: How do entailment-level results translate into changes in the answer sets of subsumption queries?

The contributions are as follows.

- A formalisation of entailment validity over an ontology version series, including survival semantics under which the removal or obsoletion of either class counts as an invalidation, and a distinction between hard invalidation, where the conclusion no longer holds, and soft invalidation, where it holds but its justification has been rewritten so that any cached explanation is stale.
- A multi-ontology measurement of entailment survival, to the authors' knowledge the first, including the loss of the justifying derivation, together with an assessment of uncertainty at the level of whole ontologies and single releases, which is much wider than the within-history bootstrap and is reported as the operative interval (Section 4.2).

- An attribution of invalidations to curation operations, showing that most arise from path breakage in which both endpoint classes, and almost always every intermediate class on the justification path, remain present and non-obsolete, so that deprecation-watching misses the large majority of invalidations, watching the parents of the subclass as well recovers a larger but still incomplete share, and only re-checking the edges of a stored justification path detects every one (Section 4.3).
- A restatement of the measurement at the level of a workload: of 3,987 cohort-style subsumption queries live at the start of 2022, 21.9% lose at least one member of their answer set within three years, and 57.9% of the lost members are invisible to deprecation-watching.
- An assessment of predictability, together with the diagnostics required to interpret a negative result. Volatility is weakly predictable in one of six evaluable ontologies once a calendar-time proxy is removed from the feature set and the comparisons are corrected for multiplicity; it does not transfer between ontologies, and it yields only modest gains for budget-constrained re-verification.
- A uniform release-artefact audit of all 311 snapshots, rather than a hand-inspection of the outliers, which bounds the contribution of packaging changes to the main estimates at 1.2% of invalidations; and a direct measurement, over the resolved import closure, showing that restricting the closure to native classes accounts for 0.01% of the invalidations counted here.
- A released corpus of entailment survival records, the extraction and analysis code, and per-release provenance sufficient to reproduce every number from commit identifiers alone.

The remainder of the article is organised as follows. Section 2 positions the work against five adjacent strands of literature, including prior work on truth and view maintenance against which the present recommendation is situated, and introduces the OBO family as the study system. Section 3 describes corpus construction and defines entailment survival and the volatility, re-verification and query-level methodology used throughout. Section 4 reports results for the five research questions, a release-artefact audit and further robustness controls. Section 5 explains why prediction fails, draws out implications for system builders and for ontology developers, and discusses threats to validity. Section 6 concludes.

2. Related Work

2.1. Change Detection and Change Taxonomies

A mature body of work formalises what kinds of change an ontology can undergo and how to detect them between versions. Klein et al. [3] introduced ontology versioning and change detection on the Web, distinguishing conceptual from specification change. Noy et al. [4] developed methods for tracking changes during ontology evolution, capturing edit operations as first-class objects. Flouris et al. [8] provide a classification and survey of ontology change that remains the standard taxonomy, and Zablith et al. [9] survey the evolution lifecycle from a process-centric perspective. Khattak et al. [10] propose a change-history framework for evolving ontologies that covers versioning, provenance, consistency, recovery and visualisation of changes, and Mahfoudh et al. [11] formalise ontology changes as typed graph grammar transformations, so that an inconsistent version cannot be produced by a change in the first place. In the life-science setting specifically, Hartung et al. [12] generate complex evolution mappings between ontology versions with COnto-Diff, and Jackson et al. [13] package diffing and release workflows in ROBOT, now standard tooling across the OBO family. Closest in spirit to the present measurement, Gonçalves et al. [14] separate logically effectual from ineffectual changes across 88 versions of the National Cancer Institute (NCI) Thesaurus, 86 of which could be parsed. The logical-difference line of work makes the comparison of two versions precise: Konev et al. [15] compute, for description-logic terminologies, the concept inclusions over a given signature that one version entails and the other does not, and Gonçalves et al. [16] categorise such differences between OWL ontology versions.

This strand characterises change to the ontology as an artefact. It is diagnostic rather than predictive, and, critically for the present purpose, its unit is the edit rather than the conclusion. Even change operators that guarantee each new version is consistent [11] say nothing about whether the conclusions a downstream system derived from the previous version still follow from the new one. Gonçalves et al. [14] and the logical-difference work [15, 16] come closest in method, computing which entailments differ between two

given versions, but the question asked here is longitudinal and downstream of it: given a specific conclusion a system is holding, does it still follow? Knowing that a subtree was restructured does not tell a system which of its stored conclusions are affected, because whether an entailment survives depends on the position of that entailment relative to the edit, not on the edit alone. The contribution here is to shift the unit of analysis from the edit to the entailment and to quantify the relationship between them.

2.2. Predicting Where an Ontology Will Change

A smaller literature predicts change rather than detecting it. Pesquita and Couto [17] predict which regions of a biomedical ontology will be extended in a future version, using supervised learning over features of previous versions, and report encouraging results on Gene Ontology biological-process terms. Cardoso et al. [18] identify outdated concepts together with the type of revision they require.

This is the closest prior work on prediction, and it differs from the present study in its target. In both cases the prediction target is the ontology class: which concept will be extended, or which is outdated. The target here is the derived entailment: which conclusion will cease to hold. These are not the same, and Section 4.3 shows why the distinction matters empirically. The majority of entailment invalidations involve no change at all to either class named in the entailment; they are caused by edits elsewhere on the justification path. A model that flags classes needing revision would not, by construction, flag most of the conclusions that are about to fail.

2.3. Validity of Instances Under Evolution

Qin and Atluri [19] evaluate the validity of data instances against ontology evolution, proposing notions of structural and semantic validity and the use of semantic views to localise validation. Köpke and Eder [20] address semantic invalidation of annotations due to ontology evolution, using annotation paths and change dependencies to identify annotations requiring maintenance. Cardoso et al. [21] construct a historical knowledge graph that retains information from previous ontology versions within a single graph.

This strand is the nearest in spirit to the present work: it recognises that artefacts downstream of an ontology can be silently invalidated. It differs in unit and in method. The unit is the asserted instance or the annotation, not the derived entailment; the treatment is a validity check at a transition rather than a longitudinal survival characterisation across a decade of releases; and no predictive component is offered or evaluated. The authors are not aware of prior work that estimates a survival function for entailments, attributes invalidations to curation operations at scale, or tests whether residual lifetime can be predicted.

2.4. Truth Maintenance and View Maintenance

The design recommendation this article arrives at — store the justification with the conclusion and re-check its edges, or re-derive outright — is not new as a mechanism: it is the central idea of truth maintenance. Doyle [22] records for each belief the justification that supports it and retracts beliefs whose justifications fail; de Kleer [23] generalises this to assumption sets. The database and rule-engine literature solves the same problem under the name of view maintenance: Gupta et al. [24] maintain derived relations incrementally under updates, Volz et al. [25] apply the technique to ontologies stored in logic databases, and Motik et al. [26] give the current state of the art for datalog materialisation maintenance. On the justification side, Kalyanpur et al. [27] compute all justifications of an OWL DL entailment and Horridge et al. [28] refine them to laconic and precise forms; the "justifying path" used here is a weak special case of that notion, restricted to the taxonomic fragment so that it is a node sequence rather than an axiom set.

What that literature does not supply is the empirical case for choosing between the options. It establishes that incremental maintenance is possible and characterises its cost; it does not report how often, in practice, a cached conclusion over a real curated ontology goes stale, nor which of the cheap approximations to full maintenance actually catch the failures. That is the gap this article addresses. It matters because the cheapest approximation is also the one that existing conventions make most accessible: deprecation-watching, in which consumers track obsoletion markers and replaced-by pointers, is supported directly by the OBO obsoletion conventions and tooling [13, 29], and is therefore a natural default for ontology-backed systems. Section 4.3 quantifies what that substitution costs.

2.5. Downstream Distortion and Versioned Data

A fifth strand documents distortion in one application: the GO enrichment studies summarised in Section 1 show that results depend on the currency of the ontology and its annotations, strongly in two studies [5, 6] and for specific datasets in the third [7]. These studies are retrospective and domain-specific: a single ontology family, a single analysis type, no formalism, no cross-ontology comparison, and no mechanism that a running system could use. They establish that the problem is real and consequential; they do not characterise it generally. A related infrastructure literature asks how versioned semantic data should be stored and queried at all: Fernández et al. [30] benchmark archiving and retrieval strategies for RDF archives, Pellissier Tanon and Suchanek [31] query the edit history of Wikidata directly, and Mountantonakis et al. [32] track connectivity and integration-quality metrics across successive versions of an aggregated semantic warehouse to detect degradation from source changes. That work makes histories accessible or monitors integration quality; none of it measures the durability of what is derived from the data.

2.6. The OBO Family as a Study System

The corpus is drawn from the OBO Foundry and Library [2], a coordinated family of biomedical ontologies developed under shared principles, whose governance and quality criteria were operationalised further by Jackson et al. [29] and whose members increasingly share tooling through the Ontology Development Kit [33]. The family is a favourable study system for the present purpose because it is well governed: its ontologies are versioned in public repositories, carry explicit release identifiers, and follow documented obsoletion conventions. Because these ontologies are curated under shared conventions, drift measured here is unlikely to be inflated by undisciplined release practice; whether less governed knowledge sources drift more or less remains an empirical question (Section 5.5).

3. Methods

The aim of the study is to measure how long taxonomic entailments derived from biomedical reference ontologies remain valid, what invalidates them, and how their invalidation can be detected or predicted. The design is a retrospective longitudinal cohort study: entailments are followed across the public release histories of eight OBO-family ontologies, from first derivation to invalidation or to the end of observation, and no human participants or interventions are involved. The materials are the versioned release artefacts of these ontologies (Section 3.1); the statistical analysis is described in Sections 3.3 to 3.7.

3.1. Corpus Construction

Eight OBO-family ontologies were selected on the basis of history length and the presence of a release artefact, the OBO-format file of the ontology, committed to a public repository: the Sequence Ontology (SO) [34], the Phenotype And Trait Ontology (PATO) [35], the Human Phenotype Ontology (HP) [36], the Human Disease Ontology (DOID) [37], the Cell Ontology (CL) [38], the Environment Ontology (ENVO) [39], the Ontology of Biological Attributes (OBA) [40] and the Uberon multi-species anatomy ontology (UBERON) [41]. For each, all commits touching the release artefact were listed oldest-first and evenly subsampled to at most forty (the resulting commit lists are released with the corpus). A commit at which the release artefact is absent from the repository yields no snapshot and is skipped; this happens once in CL (2022-12-15) and once in UBERON (2022-04-07), which is why those two ontologies contribute 39 snapshots. This yields 311 snapshots in total. Because the data-version header is not always updated between commits, several snapshots share a header value, and snapshots are therefore distinguished by commit hash rather than by header value. Because the series is subsampled, calendar time rather than release index is the primary time axis; release index is reported as a sensitivity analysis in Section 4.7. Table 1 summarises the corpus.

Table 1. Corpus composition.

Ontology	Snap.	First release	Last release	Span (yr)	Classes (last)	Entail. (last)	Incident	Hard inval. (%)
SO	40	2020-04-14	2025-09-10	5.4	2,404	15,052	1,664	6.07
PATO	40	2015-03-15	2025-05-14	10.2	1,887	10,546	2,536	8.56
HP	40	2017-03-09	2026-06-23	9.3	19,836	202,740	90,121	6.37
DOID	40	2015-11-25	2026-06-30	10.6	12,221	111,432	81,791	17.81
CL	39	2015-06-17	2026-06-08	11.0	3,335	34,591	24,513	18.89
ENVO	33	2015-08-31	2026-06-29	10.8	3,872	17,235	19,242	26.60
OBA	40	2015-06-15	2026-07-14	11.1	25,344	190,324	210,960	12.23
UBERON	39	2010-03-15	2026-06-23	16.3	14,975	135,596	192,340	37.84

For each ontology: number of release snapshots analysed, first and last release date, span in years, active native classes and taxonomic entailments at the last release, number of incident entailment lifetimes, and the percentage of incident lifetimes ending in hard invalidation within the observation window.

Subsampling has two consequences worth stating. Any entailment created and destroyed entirely between two sampled releases is never observed, so measured invalidation is biased downward. And because commits are subsampled evenly over the commit sequence rather than over calendar time, the interval between sampled releases is very uneven: the median gap ranges from 13 days in SO to 98 days in DOID, and the largest single gaps are 1,153 days in OBA, 834 in UBERON and 640 in ENVO, giving ratios of largest to median interval between selected commits of 32.1 in SO and 25.6 in OBA. Deaths are dated at the release where absence is detected, so they are interval-censored rather than exactly observed; Section 5.5 reports the sensitivity of the survival estimates to that interval, which is small overall but material for OBA, together with a rebuild of the corpus under a calendar-uniform sampling rule (code/resample_calendar.py). Every snapshot is identified by commit hash and release date in the released provenance files, so the corpus is reconstructible exactly.

3.2. Entailments Over a Version Series

Let an ontology be observed as a series of releases $O_1, O_2, \ldots, O_n$ ordered by release date $\tau_1 < \tau_2 < \ldots < \tau_n$. For a release O_i let $N(O_i)$ denote the classes that are native to the ontology — those whose identifier prefix matches the ontology — and C_i those of them that are not flagged obsolete (Eq. (1)). Restricting to native classes prevents churn imported from other projects being attributed to this one; removing obsolete classes reflects what downstream consumers actually see. Section 5.5 reports the measured cost of that restriction, which is 0.01% of the invalidations counted here.

$$C_i = \{\, c \in N(O_i) : c \text{ is not flagged obsolete in } O_i \,\} \tag{1}$$

The entailment set E_i is the transitive closure of the is_a links published in the release artefact, restricted to C_i and excluding reflexive pairs (Eq. (2)), where $a \sqsubseteq^{+}_{i} b$ means that a reaches b through one or more is_a links of O_i whose endpoints all belong to C_i. An entailment is an ordered pair $(a, b) \in E_i$, read as "a is a subclass of b, and the closure justifies it"; a is called the subclass and b the superclass. For each entailment the analysis records the hop distance $d_i(a, b)$, the length of the shortest justifying path (Eq. (3)), and, at the release in which the entailment is first derived, the node sequence of one such shortest path. Recording the sequence rather than only its length is what allows Section 3.3 to detect that a justification has ceased to hold, and Section 3.4 to ask what became of the classes it passed through.

$$E_i = \{\, (a, b) \in C_i \times C_i : a \neq b \text{ and } a \sqsubseteq_i^{+} b \,\} \tag{2}$$

$$d_i(a, b) = \min\{\, |\pi| : \pi \text{ is an is_a path from } a \text{ to } b \text{ in } O_i \,\} \tag{3}$$

The measured object is therefore the named-class taxonomic fragment, not full description-logic entailment. Because the closure is taken over the is_a links as published, what it contains depends on the release pipeline. Where a release is produced by a reasoner, subsumptions inferred from logical definitions are materialised as is_a links and are therefore part of the measured entailment set; this holds for DOID and ENVO, whose entailment counts in the corpus match their classified releases rather than their asserted-only base products (Section 5.5). Where a release publishes only asserted links, inferred subsumptions are not

captured. Entailments that involve anonymous class expressions or property chains are not captured in either case. The choice keeps the corpus deterministic and reproducible without running a reasoner over 311 snapshots, and it measures exactly the hierarchy that consumers of each release artefact receive. Section 5.5 shows that, in DOID, inferred subsumptions are markedly more fragile than asserted ones, so for releases that do not materialise them the figures reported here are likely to understate the attrition that a consumer who classifies the ontology would see.

3.3. Survival Semantics

An entailment lifetime begins at the first release in which the pair appears in E_i; this is its birth, and its hard invalidation, defined next, is referred to as its death. Thereafter, at each subsequent release, exactly one of three things happens. If the pair is absent from E_i, the lifetime ends in a hard invalidation: the conclusion no longer holds. If the pair is present but at least one edge of the most recently recorded justifying path no longer holds, this is recorded as a soft invalidation: the conclusion still holds, but the derivation supporting it has been rewritten, so any cached explanation, provenance record or proof is stale while the conclusion itself is not. The recorded path is one shortest path, chosen deterministically by the order in which parents are listed. Where several shortest paths exist another may survive, so the soft event measures the staleness of a single stored explanation, which is what a system that caches one explanation holds, and overstates staleness for a system that stores every justification. If the pair is present with its recorded justification intact, the lifetime continues. Entailments still valid at the final observed release are right-censored. An entailment that ceases to hold and later holds again is recorded as a hard invalidation followed by a new lifetime; Section 4.7 reports how often this happens and how survival changes when such interruptions are not counted. A simpler proxy for the soft event, a change in the length of the shortest justifying path, is not used because it errs in both directions: adding a shortcut edge shortens the path while leaving the original derivation intact, whereas a rewrite that preserves the hop count is invisible. The length-change series is nevertheless retained in the released corpus for comparison: 40.8% of length changes are shortenings, and

the two definitions happen to give similar pooled survival (0.669 against 0.674 at three years) because the two errors roughly cancel.

Formally, let T denote the time from first derivation to hard invalidation, measured in years on the calendar axis. For a set of entailments the survival function S(t) of Eq. (4) is the probability that an entailment first derived at time zero still holds t years later. It is estimated with the Kaplan–Meier product-limit estimator [42] of Eq. (5), which accommodates the right-censoring induced by a finite observation window.

$$S(t) = \Pr(T > t) \tag{4}$$

$$\hat{S}(t) = \prod_{j:\, t_j \le t} \left(1 - \frac{e_j}{n_j}\right) \tag{5}$$

In Eq. (5), t_j are the distinct event times, e_j is the number of hard invalidations at t_j and n_j the number of entailments at risk just before t_j. Where the median of T falls inside the window it is reported; where it does not, which is the common case here, S(t) is reported at fixed horizons rather than extrapolated beyond the data. For covariate effects on the instantaneous hazard the analysis uses Cox proportional-hazards regression [43] stratified by ontology (Eq. (6)), so that each ontology contributes its own baseline hazard and only within-ontology covariate structure is pooled.

$$h_k(t \mid \mathbf{x}) = h_{0,k}(t) \exp(\boldsymbol{\beta}^\top \mathbf{x}) \tag{6}$$

In Eq. (6), h_k is the hazard in ontology k, $h_{0,k}$ its baseline hazard, x the vector of covariates computed at derivation and β the vector of coefficients shared across ontologies. Soft invalidations are not modelled as a separate competing risk: an entailment that undergoes a soft event remains under observation, and the combined hard-or-soft analysis uses the earlier of the two event times.

Because entailments present at the first observed release have unknown true age, they are left-truncated [44], and including them would bias the survival estimates in a direction that depends on the ontology's history (Section 4.2). All primary analyses therefore use incident entailments only, meaning those first observed after the first release of the series; the prevalent cohort is reported separately as a sensitivity

analysis. The two cohorts differ sharply for some ontologies, confirming that this restriction is substantive rather than cosmetic.

3.4. Attributing Invalidation to Curation

For each hard invalidation the two adjacent releases are compared and a cause assigned by the first matching condition in the following order: the subclass was removed outright; the subclass was obsoleted, with or without a replaced-by pointer; the superclass was removed; the superclass was obsoleted, with or without a replaced-by pointer; the direct subsumption edge between the two was removed (direct edge removed); otherwise, path breakage — both endpoint classes remain present and non-obsolete, but every path that previously connected them has been severed, either by a change to the parents of the subclass itself or by an edit further up the path. This ordering assigns each invalidation to the most proximate observable cause and makes path breakage a residual category, which is conservative for the claim drawn from it, since a case can only reach this category once every prior, more specific explanation has been ruled out. Because it is a residual category it is decomposed further. Let P be the set of intermediate classes that lay on some path from a to b in the release preceding the invalidation, that is, the classes x other than a and b for which $a \sqsubseteq^{+} x$ and $x \sqsubseteq^{+} b$ in that release. For brevity, the members of P are referred to below as the classes on the justification path; because P collects the intermediate classes of every path from a to b, and not only of the recorded one, the detection rates reported for deprecation-watching on the justification path are upper bounds for a consumer that stores a single path. A path breakage is classified as intermediate-removed if any member of P is absent from the later release, otherwise intermediate-obsoleted if any member is flagged obsolete, and otherwise intermediate-reparented, meaning that every class involved in the conclusion and in its derivation survives unchanged in identity and status while the edges between them no longer compose. Only the first two categories emit a deprecation signal a consumer could watch for; Section 4.3 measures how much of the third is visible as a change to the parents of the subclass itself. Throughout, a deprecation signal means that a class has been flagged obsolete (with or without a replaced-by pointer) or has been removed from the release; a consumer can observe either without re-deriving any entailment. A class counts

as removed when its identifier no longer heads a term in the release; this includes classes merged into another class, whose identifier is retained only as an alternative identifier of the surviving class.

3.5. Volatility Modelling

To test whether residual lifetime is predictable a classification task is defined; the fitted classifier is referred to throughout as the volatility model, and its output as the predicted volatility of an entailment. For a horizon H, an incident entailment is labelled as in Eq. (7): positive if it is hard-invalidated within H years of its first derivation, and negative if it is observed to survive at least H years. Entailments whose status H years after derivation is not observable are dropped, so no label is ever inferred from unobserved time. A training entailment derived shortly before the cut date has its label resolved after the cut, so the split is temporal in derivation date but not in label availability; the deployment simulation of Section 3.6 uses the stricter criterion. H = 3 throughout.

$$y = \begin{cases} 1 & \text{if } T \leq H \\ 0 & \text{if } T > H \text{ is observed} \end{cases} \tag{7}$$

Splitting is temporal in derivation date: models train on entailments derived before a cut date and are tested on entailments derived on or after it, with the cut at 2021.0. Fourteen features are computed at the release in which the entailment was first derived and use no information from the future: hop distance, depth of the subclass (the largest shortest-path distance from the subclass to any of its ancestors), its number of parents, its number of ancestors, the number of descendants of the superclass, whether the subclass is a leaf, the age of the subclass in releases, the number of prior revisions to its parent set, the size of the ontology, whether the subclass is newly added, and four features describing the intermediate classes that lie on some justifying path between the two endpoints: how many there are, the largest number of prior parent-set revisions among them, the age of the youngest, and the largest parent count. The last four are motivated by the attribution result of Section 4.3, which finds that the great majority of invalidations originate at exactly those intermediate classes. They are used in the pooled models and the stratified diagnostics of Section 4.4; per-ontology models use the ten base features. Because ontology size is, within a single ontology, a monotone

proxy for release index and hence for birth date, within-ontology models are reported both with it and without it, the latter on the remaining nine features. L2-regularised logistic regression and histogram-based gradient boosting (the scikit-learn implementation, which is inspired by LightGBM [45]) are fitted, and additionally a Cox proportional-hazards model stratified by ontology (Eq. (6)), fitted with an L2 penalty of 0.1, to characterise covariate effects on the hazard. All models are fitted on the full training fold (Section 3.8).

Because a naive reading of a single AUROC value is unsafe on data of this kind, models are evaluated against a permutation null [46]: for each ontology the model is retrained on 500 independent random permutations of the training labels (350 for UBERON and 200 for the CL configuration that retains ontology size, whose fits are the slowest; the number is recorded per row in the released table), holding the training and test folds, the feature set and the model hyperparameters identical to those of the observed model, and the resulting distribution of test AUROC is recorded. Observed performance is judged against the 95th percentile of that null rather than against 0.5, and the empirical permutation p-value is also reported. Section 4.4 adds three diagnostics that are needed to interpret a negative result of this kind: an evaluation stratified by ontology, which separates within-ontology discrimination from the effect of the changing ontology mix; a sign-flip test whose direction is chosen on a validation window preceding the test fold, so that an inverted ranking cannot be claimed as a finding unless it could actually have been exploited; and a decomposition of the permutation null against two simpler nulls, which establishes what makes it as wide as it is.

3.6. Re-Verification Simulation

The practical question is not whether volatility is predictable in principle but whether prioritising re-verification by predicted volatility saves work. The simulation models a system that at reference date $\tau_0 = 2022.0$ holds every entailment then valid and may re-verify only a fixed fraction B of them before the next audit. The ranking model is trained only on entailments whose outcome was already observable before τ_0. The measure is recall at budget B (Eq. (8)): of the entailments that are actually invalidated over the

following three years, what fraction fall within the re-verified budget? The learned model is compared against random ordering and three heuristics, and the ratio of a ranking's recall to that of random ordering is reported as its lift.

$$\mathrm{Recall}(B) = \frac{|R_B \cap I|}{|I|}, \qquad \mathrm{Lift}(B) = \frac{\mathrm{Recall}_{\mathrm{model}}(B)}{\mathrm{Recall}_{\mathrm{random}}(B)} \tag{8}$$

In Eq. (8), R_B is the set of entailments re-verified under budget B and I the set of entailments invalidated over the following three years. Ties are broken uniformly at random. This choice is consequential: the survival table is written in event order, so row order correlates with outcome, and any deterministic tie-break leaks that ordering into rankings with many tied scores; about 81.2% of the incident entailments live at τ_0 have a subclass that has never been revised.

3.7. Query-Level Restatement

Entailment-level survival is the right unit for measurement but the wrong unit for a system owner, who holds queries rather than pairs. The results are therefore restated at the level of a cohort-style subsumption query, the request that an ontology-backed system issues whenever it expands a term to its descendants or propagates annotations along the hierarchy, as phenotype-driven diagnostic tools do [47, 48]: return everything subsumed by X. The answer set of such a query at a given release is exactly the set of subclasses of X entailed in that release, which the survival corpus records directly, so the restatement requires no additional data and introduces no assumption about a specific application. For every superclass with at least ten members alive at $\tau_0 = 2022.0$, the analysis computes the members lost by $\tau_0 + 3$, and splits those losses by whether a consumer watching deprecation signals on the two classes named in the entailment could have detected them. This is a query-level restatement of the same corpus, not a deployed workload with measured costs; Section 5.5 says what it does and does not support.

3.8. Implementation and Reproducibility

Snapshots are extracted from git by commit hash and parsed with a purpose-written parser. Survival analysis uses lifelines [49, 50] and predictive models use scikit-learn [51, 52]; the code runs under Python 3.10–3.13. Confidence intervals are obtained by a bootstrap with 1,000 replicates that resamples whole subclasses rather than individual entailments, each interval drawn from its own seeded generator so that it does not depend on the order in which the tables are computed; the standard errors of the Cox model reported in Section 4.4 use 200 replicates of the same subclass bootstrap; the ontology-level interval of Section 4.2 uses 2,000 replicates that resample whole ontologies. All analyses are run on a deterministically ordered corpus. This matters: emitting survival records in the iteration order of a set of class identifiers and then taking a positional subsample of the training fold makes several figures depend on the interpreter's hash seed, and the pooled logistic AUROC, for instance, varies between 0.477 and 0.521 across four row orders under that procedure (ordering_sensitivity.csv). The extraction code, survival-construction code, analysis scripts, per-release provenance tables and the full survival corpus are released with the article. Five further analyses, whose methods are described where their results are reported, require the ontology repositories rather than the released corpus: the prediction of bulk-restructuring events from release-process features (Section 5.2), the timing of re-derivation and of the justification check (Section 5.3), and, in Section 5.5, the comparison of asserted and reasoner-classified release products, the recomputation of the closure over the resolved import closure, and the rebuild of the corpus under calendar-uniform sampling. They are shipped as runnable scripts together with their outputs (see Availability of data and materials). Parts of the analysis code were developed with the assistance of a large language model (Claude, Anthropic); all code was reviewed by the authors and is released in full, and the reported quantities are checked by the released verification scripts.

4. Results

4.1. Corpus

Figure 1 shows growth in classes and in entailments over the observed release histories summarised in Table 1. In total the corpus contains 1,691,732 survival records: 929,486 terminal lifetimes (of which 623,167 are incident), 377,716 events in which the justification recorded for a still-valid entailment was superseded, and 384,530 events in which the shortest-path length changed — the proxy discussed in Section 3.3, retained in the released corpus for comparison. Because births accumulate over the window while the entailment set is measured at a single release, an ontology can contribute more incident lifetimes than it holds entailments at the end: OBA contributes 210,960 incident lifetimes against 190,324 entailments at its last release.

The proportion of incident entailments that are hard-invalidated within the observation window varies sixfold across ontologies, from 6.1% in SO to 37.8% in UBERON. This spread is large enough that no claim about "ontologies" in general is safe without per-ontology reporting, and per-ontology results are reported throughout.

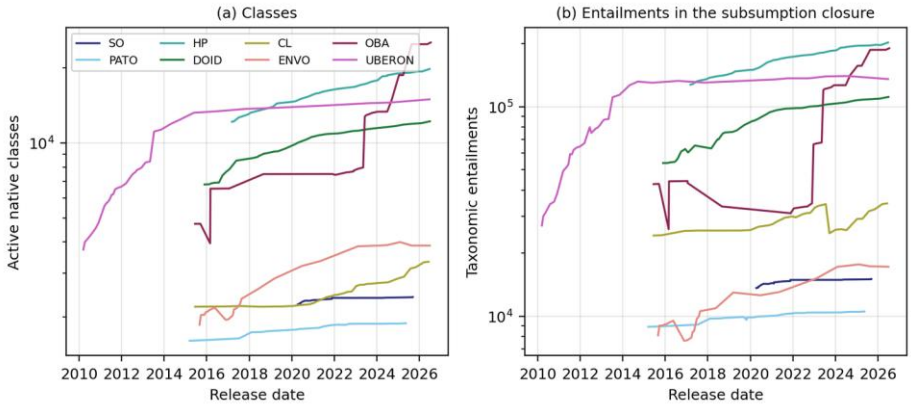


Figure 1. Growth of the eight-ontology corpus over its observed release history. (a) Active native classes per release; (b) taxonomic entailments in the subsumption closure per release. Both vertical axes are logarithmic.

4.2. RQ1: How Long Does an Entailment Last?

Figure 2 shows Kaplan–Meier survival curves for incident entailments, and Table 2 gives survival at fixed horizons with 95% intervals from a bootstrap that resamples whole subclasses, since entailments sharing a subclass tend to be invalidated together. Pooled over 623,167 incident lifetimes, the probability that an entailment still holds is 0.927 at one year, 0.823 (95% CI 0.822–0.825) at three years and 0.786 (0.784–0.788) at five years. In other words, roughly one taxonomic entailment in five is no longer valid five years after it was first derived. Estimates are reported only where at least 100 entailments remain at risk at that horizon, which is why the eight-year column is empty for SO, whose observation window is 5.4 years, and for PATO, where 67 entailments remain. SO's three-year and five-year estimates are identical because no incident SO entailment is invalidated between those two horizons, not because a value has been carried over.

The intervals in Table 2 describe sampling variation among entailments within the observed release histories, not variation between histories, and because attrition is episodic (Section 4.7) the latter dominates. Resampling whole ontologies with replacement gives a 95% ontology-level interval of 0.757–0.902 for pooled three-year survival and 0.708–0.886 for five-year survival. The pooled estimate implicitly weights ontologies by their number of entailments at risk, so that OBA and UBERON supply 65% of the incident lifetimes; the unweighted mean over the eight ontologies is 0.850 at three years and 0.818 at five. Dropping the invalidations of any single release moves pooled five-year survival only within 0.785–0.811, so no single release drives the pooled figure, although single releases do drive several per-ontology figures (Section 4.7). The intervals in Table 2 should therefore be read as conditional on the observed histories, and the ontology-level interval as the uncertainty of a claim about curated ontologies of this kind. These analyses are released as robustness_uncertainty.csv.

Table 2. Kaplan–Meier survival of incident entailments at fixed horizons.

Ontology	n	S(1 yr)	S(3 yr) [95% CI]	S(5 yr) [95% CI]	S(8 yr)	At risk 5 yr	S(3 yr) h+j	S(5 yr) h+j
SO	1,664	0.940	0.936 [0.903–0.964]	0.936 [0.903–0.964]	—	622	0.875	0.875
PATO	2,536	0.984	0.944 [0.920–0.967]	0.903 [0.871–0.933]	—	1,385	0.915	0.863
HP	90,121	0.979	0.955 [0.952–0.957]	0.942 [0.939–0.945]	0.894	46,265	0.816	0.743
DOID	81,791	0.881	0.831 [0.824–0.837]	0.815 [0.808–0.821]	0.808	49,691	0.723	0.675
CL	24,513	0.917	0.772 [0.762–0.781]	0.713 [0.699–0.724]	0.617	3,837	0.597	0.473
ENVO	19,242	0.896	0.776 [0.766–0.786]	0.713 [0.703–0.724]	0.698	9,493	0.676	0.587
OBA	210,960	0.979	0.850 [0.848–0.852]	0.838 [0.836–0.840]	0.754	26,949	0.787	0.739
UBERON	192,340	0.871	0.738 [0.734–0.741]	0.683 [0.679–0.687]	0.664	119,936	0.491	0.428
Pooled	623,167	0.927	0.823 [0.822–0.825]	0.786 [0.784–0.788]	0.758	258,178	0.674	0.608

S(t) is the probability that an entailment first derived at time 0 still holds at time t; bracketed intervals are 95% bootstrap intervals (1,000 replicates) resampling whole subclasses within the observed release histories; Section 4.2 reports the wider interval obtained by resampling whole ontologies. A dash means fewer than 100 entailments remain at risk at that horizon, so no estimate is reported. The final two columns require in addition that the specific path that justified the entailment at derivation still holds (h+j).

The restriction to incident entailments is not a technicality. Repeating the analysis on the prevalent cohort — entailments already present at the first observed release, whose true ages are unknown — gives higher five-year survival than the incident cohort in six of the eight ontologies (for example, 0.971 against 0.713 for CL), but much lower survival in OBA (0.136 against 0.838) and ENVO (0.403 against 0.713). Per-ontology values are released as prevalent_cohort.csv. In these two ontologies a large population of long-standing entailments was swept away by a restructuring that postdates their unobserved birth. Analyses that ignore left-truncation therefore report biased survival, and the direction of the bias differs between ontologies.

Counting the loss of the justifying derivation alongside hard invalidation changes the picture materially. The probability that an entailment both still holds and retains the derivation that originally justified it is 0.674 at three years and 0.608 at five. Two conclusions in five have therefore either lapsed outright or had

their supporting derivation silently rewritten within five years. For a system that caches explanations, provenance chains or proofs alongside conclusions — as auditable decision-support systems are expected to do — the second figure is the operative one.

Median survival is not reached within the observation window for hard invalidation in seven of the eight ontologies; only UBERON yields a finite estimate, at 16.11 years, which sits at the very edge of a 16.3-year window and should be treated as unstable rather than as a point estimate. For hard invalidation or loss of justification, medians exist for UBERON at 2.91 years, CL at 4.06 years and pooled at 11.76 years. Fixed-horizon survival is therefore the primary summary; a half-life is not reported because the data mostly do not identify one.

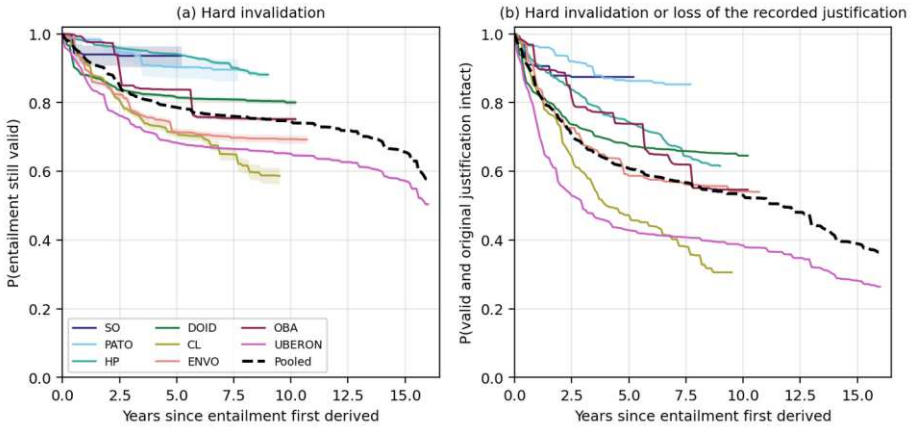


Figure 2. Kaplan–Meier survival of incident taxonomic entailments. (a) Hard invalidation: the entailment no longer holds. (b) Hard invalidation or loss of the recorded justification. Curves are truncated where fewer than 100 entailments remain at risk; shaded bands in (a) are 95% bootstrap intervals resampling whole subclasses (1,000 replicates). Dashed black line is the pooled estimate (n = 623,167).

Figure 3 stratifies survival by four covariates available at derivation time. Differences are visible but modest, and the strata are unevenly populated: 495,863 of the 623,167 incident entailments have a superclass with more than 100 descendants, so that is the ordinary case rather than the exception. Those entailments survive less well than the ones whose superclass has at most one descendant, at 0.768 against 0.899 at five years, with the intermediate bins at 0.904 and 0.824. Entailments whose subclass was newly

added when the entailment was derived survive best (0.829 at five years), whereas those whose subclass had existed for one or two releases survive worst (0.633), consistent with the protective effect of the newly-added indicator in the Cox model of Section 4.4. Entailments whose subclass had already been revised are more fragile than those whose subclass had never been revised, at 0.713 against 0.806 at five years, but the gradient is not monotone: five-year survival is 0.704 after a single prior revision, 0.710 after two or three, and 0.748 after four or more, so a high revision count is not by itself a danger signal. These gradients are too shallow to support the triage of individual conclusions, which anticipates the predictive results in Section 4.4.

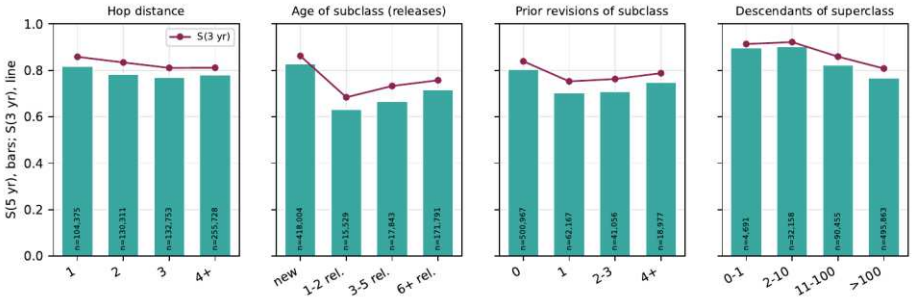


Figure 3. Survival of incident entailments stratified by four covariates computable at derivation time. The covariates are hop distance, age of the subclass in releases, prior revisions to the subclass's parent set, and number of descendants of the superclass. Bars give five-year survival and the line three-year survival; n is the number of entailments in each stratum.

4.3. RQ2: What Causes Invalidation?

Table 3 and Figure 4 give the composition of hard invalidations by attributed cause. Pooled across ontologies, 71.8% are path breakage and a further 13.7% are removal of the direct subsumption edge. Obsoletion or removal of the subclass accounts for 5.3% and of the superclass for 9.2%, together 14.5%. Because path breakage is a residual category, it is decomposed further by asking what became of the intermediate classes that lay on some path between the two endpoints in the preceding release. Of the 92,638 path-breakage events, 97.2% occur with every such intermediate class still present and non-obsolete; only 2.7% involve an obsoleted intermediate and 0.1% a removed one. Figure 5 gives the per-ontology composition.

Table 3. Composition of hard invalidations by attributed cause.

Ontology	Path broken	Direct edge removed	Subclass obs./rem.	Superclass obs./rem.
SO	28.71	4.95	58.42	7.92
PATO	71.43	22.12	4.15	2.30
HP	55.87	5.26	26.44	12.43
DOID	80.87	10.68	5.20	3.25
CL	39.05	5.14	5.42	50.39
ENVO	57.09	7.15	15.38	20.38
OBA	81.21	16.81	1.17	0.81
UBERON	71.16	14.78	4.36	9.70
Pooled	71.84	13.65	5.32	9.19

Values are percentages of each ontology's incident hard invalidations. Subclass and superclass columns aggregate obsoletion and removal, with and without a replaced-by pointer. Path breakage means both endpoint classes remain present and non-obsolete while every connecting path was severed; Figure 5 decomposes that column by what became of the intermediate classes, which in 97.2% of cases are themselves still present and non-obsolete.

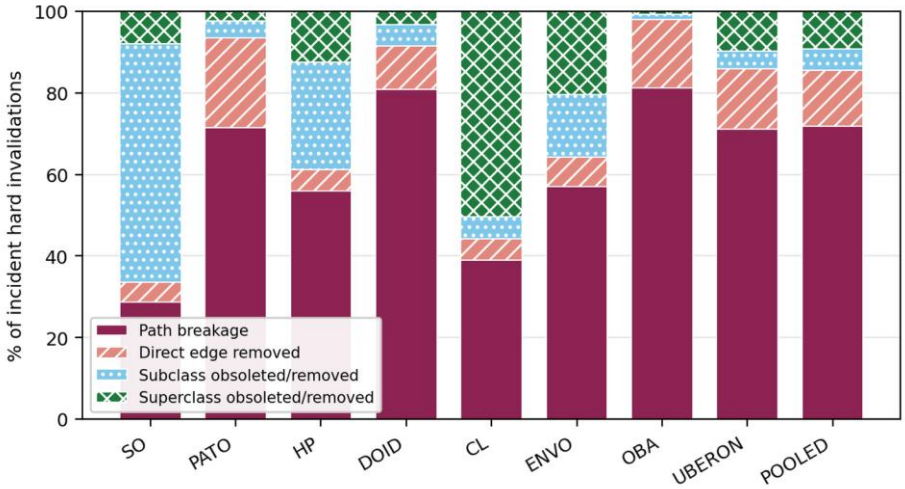


Figure 4. Composition of hard invalidations by attributed cause, per ontology and pooled. Values are percentages of incident hard invalidations. Hatching duplicates colour so that the figure remains readable in greyscale.

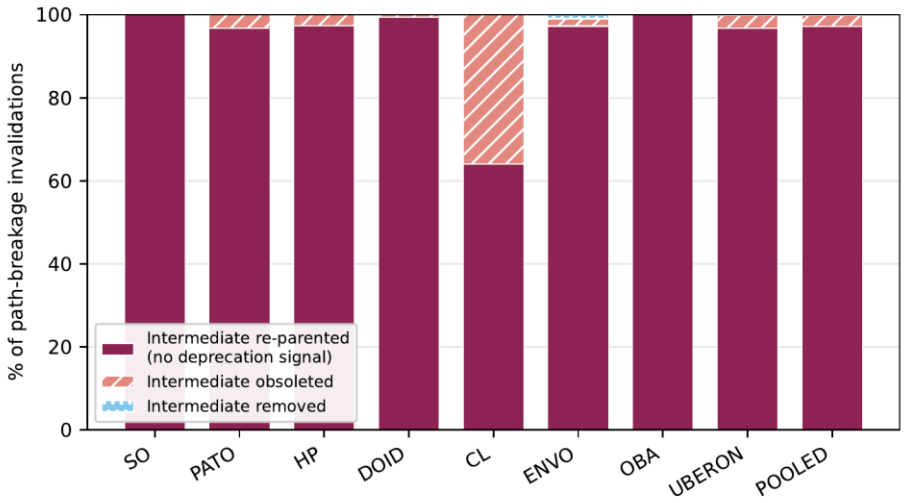


Figure 5. Decomposition of path-breakage invalidations by what became of the intermediate classes. The intermediate classes are those lying on any path between the two endpoints in the preceding release (Section 3.4). In 97.2% of cases every intermediate class is still present and non-obsolete, so no deprecation signal anywhere on the path could have detected the invalidation.

The dominance of path breakage has the most direct practical implications. The defence against ontology drift that OBO conventions and tooling support directly [13, 29] is deprecation-watching: check whether the classes referenced by a stored conclusion have been obsoleted, and follow replaced-by pointers when they have. These results show that, pooled over the corpus, this strategy is blind to roughly 85% of invalidations, although the share varies widely by ontology (Table 3), and that extending it to every class on the justification path recovers only two percentage points more, reaching 16.5%. In most cases every class involved in the conclusion and in its derivation is still present, still non-obsolete, and still carries the same identifier; what changed is which edges connect them. No deprecation signal can detect this. Detecting it requires either re-deriving the entailment or storing the justifying path and re-checking its edges, which is the truth-maintenance discipline of Section 2.4 applied to the taxonomic case.

Path breakage is not, however, always caused far from the conclusion. In at least 21.8% of path breakages (20,230 of 92,638) the subclass lost one of its own direct parents at the same release, so the edit that severed the path was made at the subclass rather than at an intermediate class; the share is 27.2% in UBERON and

79.4% in PATO. The figure is a lower bound, because the corpus records a lost parent only when that loss itself invalidated an entailment. A consumer that watches, in addition to deprecation markers, whether the parent set of the subclass has changed therefore detects at least 43.8% of invalidations, from 29.7% in DOID to 85.3% in PATO. That is a much better cheap monitor than deprecation-watching, but it still misses up to 56% of invalidations. Re-checking the edges of a stored justification path detects every hard invalidation by construction, since a path whose edges all still hold still entails the conclusion; across the incident cohort it raises 335,044 alarms, of which 38.5% are hard invalidations and the remainder soft invalidations, in which the conclusion holds but its recorded derivation does not. The per-ontology comparison of these detectors is released as robustness_detectors.csv.

Composition varies by ontology. Path breakage plus direct-edge removal is the majority cause in six of the eight ontologies, ranging from 61.1% in HP to 98.0% in OBA. The two exceptions are CL, at 44.2%, where superclass obsoletion with a replaced-by pointer is unusually frequent, and SO, at 33.7%, where outright removal of the subclass — removal specifically, as distinct from the 58.42% of SO invalidations that Table 3 aggregates over removal and obsoletion together — accounts for 50.5%; SO contributes only 101 incident hard invalidations, so this reflects a small number of bulk deletions rather than a distinct curation regime.

4.4. RQ3: Is Volatility Predictable?

The Cox proportional-hazards model, stratified by ontology and fitted on the full incident cohort, achieves a concordance of 0.642, rising to 0.669 when the path features are added. Table 4 reports its covariate effects. Among the covariates that increase the hazard, the number of ancestors of the subclass carries the largest hazard ratio, at 1.322 per unit of log-scaled count, followed by the number of descendants of the superclass at 1.065 and the number of parents at 1.048; ontology size (0.660) and a newly added subclass (0.664) have the largest protective effects. Read descriptively, entailments whose subclass has many ancestors and whose superclass has many descendants are the fragile ones. Because entailments sharing a subclass tend to be invalidated together, the standard errors in Table 4 come from a bootstrap that resamples whole subclasses and refits the model (200 replicates); these clustered standard errors are 0.77 to 1.57 times

the model-based ones, and wider for seven of the ten covariates. With them, the evidence for depth of the subclass ($p = 0.030$) and the leaf indicator ($p = 0.032$) is weaker than the model-based standard errors suggest, although the p-value of every coefficient remains below 0.05. Because the fit is penalised, the p-values are approximate and are reported descriptively. Three further cautions apply, and together they mean this model should be read as a description of association rather than as a mechanism. A Schoenfeld residual test [53] rejects proportional hazards for six of the ten covariates, so these ratios should be read as time-averaged rather than constant; the covariates are strongly collinear, ancestor count, depth and hop distance being three views of position in the hierarchy, so individual coefficients should not be given a causal reading; and dropping each ontology's single largest invalidation release moves them materially: the hazard ratio for ontology size goes from 0.660 to 0.807 and that for ancestor count from 1.322 to 1.479.

Table 4. Cox proportional-hazards model of time to hard invalidation.

Covariate	Coefficient	Hazard ratio	SE	p
Ancestors of subclass (log)	0.2793	1.322	0.0079	<1e-4
Descendants of superclass (log)	0.0633	1.065	0.0009	<1e-4
Number of parents	0.0470	1.048	0.0044	<1e-4
Hop distance	−0.0230	0.977	0.0013	<1e-4
Depth of subclass	−0.0039	0.996	0.0018	0.0300
Subclass is a leaf	−0.0198	0.980	0.0092	0.0317
Age of subclass (releases)	−0.0024	0.998	0.0003	<1e-4
Prior revisions of subclass	−0.0084	0.992	0.0027	0.0022
Ontology size (log)	−0.4152	0.660	0.0055	<1e-4
Subclass newly added	−0.4088	0.664	0.0061	<1e-4

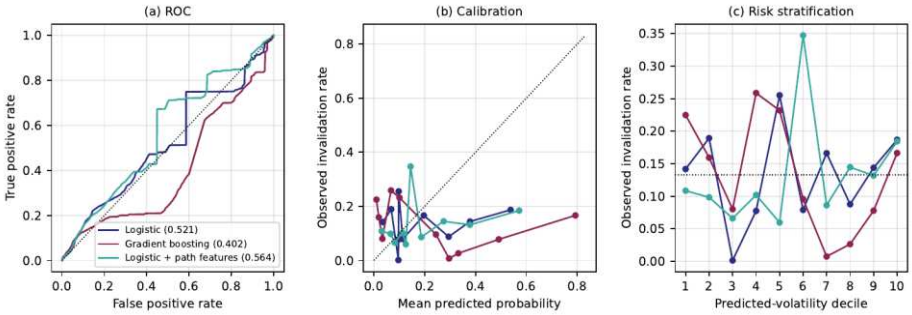
The model is stratified by ontology and fitted on the full incident cohort of 623,167 lifetimes with an L2 penalty of 0.1 (concordance 0.642). Coefficients and hazard ratios are from that fit; standard errors and approximate p-values come from a bootstrap that resamples whole subclasses (200 replicates over the 74,951 subclasses) and are 0.77 to 1.57 times the model-based ones, which table4.csv also carries. Counts marked (log) are transformed as log(1 + x). Hazard ratios are time-averaged, since proportional hazards are rejected for six of the ten covariates.

Modest covariate effects translate into poor prediction. Table 5 reports the pooled cross-ontology model under the temporal holdout. Logistic regression reaches an AUROC of 0.521 (95% CI 0.513–0.529) — close to chance rather than usefully above it — and gradient boosting 0.402 (0.395–0.409), materially below chance. Removing ontology size, which doubles as a proxy for ontology identity and for calendar time, lowers the gradient-boosting AUROC further, to 0.356. Adding the path features lifts logistic regression to 0.564 (0.556–0.571), the strongest pooled result, and leaves gradient boosting below chance at 0.443; a single path feature — the largest number of prior parent-set revisions among the intermediate classes — reaches 0.590 on its own. Figure 6 shows discrimination, calibration and risk stratification for these models.

Table 5. Pooled cross-ontology volatility model under a temporal holdout on derivation date.

Model / baseline	**AUROC**	**AUPRC**	**Brier**
Logistic regression	0.521 [0.513–0.529]	0.155	0.140
Gradient boosting	0.402 [0.395–0.409]	0.130	0.198
Gradient boosting, ontology size removed	0.356	0.100	0.265
Baseline: hop distance	0.563	0.152	—
Baseline: superclass descendants	0.531	0.154	—
Baseline: subclass age	0.519	0.239	—
Baseline: prior revisions of subclass	0.478	0.133	—
Baseline: path revision count	0.590	0.167	—
Logistic regression + path features	0.564 [0.556–0.571]	0.166	0.139
Gradient boosting + path features	0.443 [0.436–0.450]	0.130	0.192

Models are trained on entailments derived before 2021 and tested on those derived from 2021 onward (n = 150,682, base rate 0.133), with a three-year horizon. AUROC below 0.5 indicates an inverted ranking.



615

616 *Figure 6. Pooled cross-ontology volatility models under the temporal holdout. (a) ROC, with AUROC in the legend; (b)*
617 *reliability diagram, with the diagonal marking perfect calibration; (c) observed invalidation rate by predicted-volatility*
618 *decile, with the dotted line marking the test base rate of 0.133. Colours are shared across panels. Neither the logistic nor*
619 *the gradient-boosted model transfers across the temporal split, and adding the path features does not change this.*

620 A sub-chance AUROC on 150,682 test cases is a systematic inversion rather than noise, and it should not
621 be reported without saying what produces it and whether it could be exploited. Table 6 gives the diagnostics.
622 The ontology mix does shift sharply across the cut: UBERON supplies 51.8% of the training fold and OBA
623 63.1% of the test fold, and the base rate of UBERON in training (0.269) is about three times that of OBA
624 in testing (0.093). But stratifying the evaluation by ontology, so that discrimination is measured within each
625 ontology and averaged with weights proportional to test size, makes every model worse, not better: logistic
626 regression falls from 0.521 to 0.415 and gradient boosting from 0.402 to 0.352. The appropriate
627 interpretation is therefore the opposite of the intuitive one. The pooled figures are flattered by between-
628 ontology variation, which lets a model separate high-risk from low-risk ontologies without saying anything
629 useful about individual entailments; conditional on ontology, the ranking these models produce on later
630 data is inverted. The negative result is stronger than the pooled numbers alone suggest.

631 Whether the inversion can be exploited is a separate question, and the answer is no. Choosing the sign of
632 the score on a validation window that precedes the test fold and is excluded from training — the only way
633 to select a flip without using the test data — never selects a flip, because on that window every model
634 discriminates positively, with AUROC between 0.658 and 0.740. The direction of the feature–outcome

relationship itself reverses between the validation period and the test period. This is non-stationarity, not an inverted but stable signal, and it is why no simple recalibration corrects these models.

Table 6. Diagnostics for the pooled models of Table 5.

Model	AUROC pooled	AUROC stratified	AUROC validation window	Flip selected
Logistic regression	0.521	0.415	0.714	no
Gradient boosting	0.402	0.352	0.658	no
Gradient boosting, ontology size removed	0.356	0.344	0.696	no
Logistic regression + path features	0.564	0.471	0.740	no
Gradient boosting + path features	0.443	0.403	0.680	no

Stratified AUROC is computed within each ontology on the test fold and averaged with weights proportional to test size. The validation window is entailments derived in [2019.5, 2021.0), used neither for training nor for testing; the sign of the score is selected on it and applied unchanged to the test fold, so a flip is never chosen with knowledge of the test outcome. No configuration selects a flip.

Restricting each model to a single ontology, so that it trains and tests on that ontology's own release history, is the operationally realistic setting and requires care about what the model is allowed to see. With ontology size in the feature set, two ontologies exceed the 95th percentile of their permutation null and UBERON reaches an AUROC of 0.919 (95% CI 0.907–0.929) — but within a single ontology that feature is a monotone proxy for the release index, so such a model is largely learning how close an entailment's birth was to the next bulk revision, which is not knowable at derivation time. Table 7 therefore reports both configurations. Without the time proxy, UBERON falls to 0.554 (0.539–0.569), OBA to 0.587, ENVO to 0.597, HP to 0.518 and CL to 0.371, which is inverted; only DOID clears its null, at 0.708 (0.674–0.740) against a null 95th percentile of 0.647, with an empirical permutation p of 0.006 and a Benjamini–Hochberg [54] q of 0.036 across the six comparisons. Two ontologies are not evaluable and are omitted: SO, whose pre-cut training fold holds only 814 entailments, and PATO, only two of whose 402 test entailments are positive. The permutation null is wide — its 95th percentile ranges from 0.567 to 0.687 across the six

evaluable ontologies — which is precisely why judging these models against a nominal 0.5 would have been misleading.

Table 7. Within-ontology volatility models under the same temporal holdout.

Ontology	n test	Base rate	AUROC without time proxy [95% CI]	Null 95th pct.	Exceeds null (BH q)
DOID	9,569	0.126	0.708 [0.674–0.740]	0.647	yes (q = 0.036)
ENVO	4,517	0.390	0.597 [0.573–0.618]	0.624	no (q = 0.324)
OBA	95,080	0.093	0.587 [0.581–0.592]	0.683	no (q = 0.412)
UBERON	8,422	0.308	0.554 [0.539–0.569]	0.687	no (q = 0.441)
HP	23,220	0.087	0.518 [0.498–0.536]	0.593	no (q = 0.470)
CL	8,768	0.409	0.371 [0.357–0.384]	0.567	no, inverted (q = 0.682)

Ontology size is excluded because within a single ontology it is a monotone proxy for release index. Models are judged against the 95th percentile of a permutation null obtained by retraining on permuted training labels; q values are Benjamini–Hochberg-corrected across the six comparisons. SO and PATO are not evaluable (see text). With ontology size retained, UBERON reaches 0.919 and DOID 0.768; that configuration is regarded as measuring the timing of past shocks rather than fragility.

The width of that null deserves an explanation, because a label permutation preserving the base rate would centre a random score on 0.5 and a 95th percentile near 0.70 looks anomalous. Two candidate explanations were tested against the same test folds. Clustering of entailments under shared subclasses accounts for very little of it: scores drawn at random but held constant within a subclass give a null whose 95th percentile is 0.505 for OBA, 0.514 for CL and 0.535 for DOID, far narrower than the permutation null. The explanation is instead that a model retrained on permuted labels still lands on some direction in feature space, and under temporal shift an arbitrary direction discriminates surprisingly well: the AUROC achieved by single features and their negations reaches a 95th percentile of 0.685 in OBA, 0.650 in ENVO and 0.815 in UBERON, which matches or exceeds the permutation null in every evaluable ontology, while the cluster

null lies below it; together the two simpler nulls bracket the permutation null. The permutation null is therefore measuring the right thing, and judging against it rather than against 0.5 is essential to the conclusions of this section.

Cross-ontology transfer fails. Training on seven ontologies and testing on the eighth gives an AUROC below chance in five of the seven evaluable cases (PATO, with only two positive test entailments, is not evaluable), from 0.224 for SO to 0.626 for ENVO. Volatility structure is ontology-specific: a model borrowed from another ontology is not merely uninformative but, on this evidence, systematically misleading, although with seven values and no confidence intervals that reading should be treated as descriptive. A feature-family ablation in the within-ontology setting shows both feature families contribute and neither suffices alone, with mean AUROC of 0.594 using all features, 0.587 using structural features only and 0.566 using edit-history features only.

4.5. RQ4: Does Prioritisation Save Work?

Table 8 and Figure 7 report the re-verification simulation. Averaged over the six ontologies with sufficient events (HP, DOID, CL, ENVO, OBA and UBERON), the volatility model recovers 13.6% of the invalidations that occur over the following three years when re-verifying 10% of the live cache, against 10.5% for random ordering — a lift of 1.30. At a 5% budget the lift is 1.39, and it decays towards 1.08 as the budget grows. Two simple orderings do at least as well: ranking by how often the subclass's own parent set has previously been revised reaches 14.1% at a 10% budget, and ranking by the largest such count among the intermediate classes on the justification path reaches 16.9%, a lift of 1.61 and better than the learned model at every budget up to 30%. The average conceals wide variation: in CL and ENVO the model recovers fewer invalidations than random ordering (3.1% and 2.8% at a 10% budget, against 10.4% and 10.5%), and even the path-revision heuristic falls below random ordering in OBA (11.0% against 11.9%).

Table 8. Re-verification simulation at $\tau_0 = 2022.0$ with a three-year horizon.

Budget	Volatility model	Random	Subclass revisions	Path revisions	Oldest first	Lift vs random
5%	0.072	0.051	0.085	0.101	0.029	1.39
10%	0.136	0.105	0.141	0.169	0.059	1.30
20%	0.257	0.204	0.249	0.298	0.108	1.26
30%	0.363	0.302	0.346	0.389	0.175	1.20
50%	0.552	0.511	0.526	0.547	0.328	1.08

Values are the recall, as a proportion, of entailments actually invalidated over the following three years, averaged over the six ontologies with sufficient events (HP, DOID, CL, ENVO, OBA, UBERON). Ties are broken uniformly at random. The final column is the lift of the volatility model over random ordering.

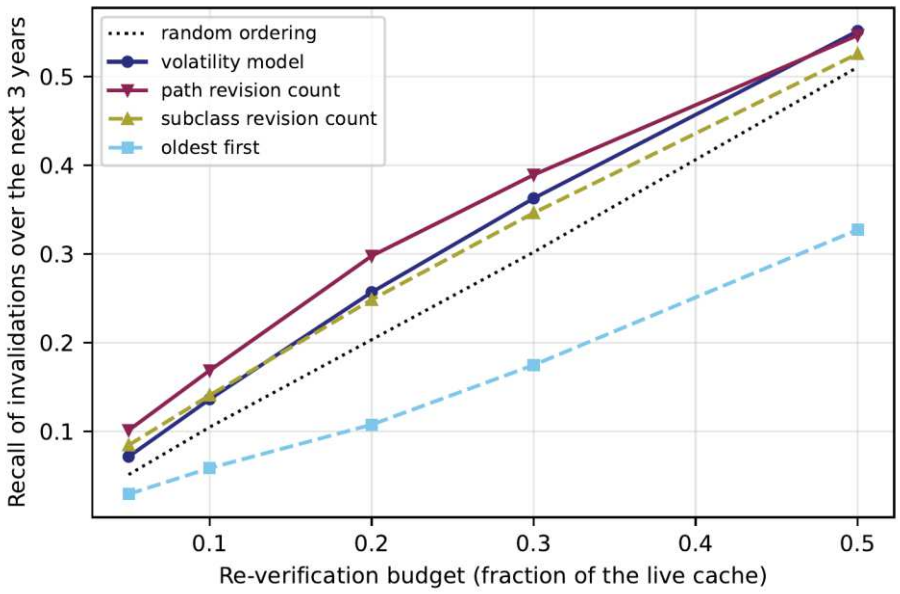


Figure 7. Recall of future invalidations as a function of the re-verification budget. Recall is averaged over the six ontologies with sufficient events. Solid lines show the learned volatility model and the path-revision heuristic, dashed lines the subclass-revision and oldest-first heuristics, and the dotted line the random-ordering expectation.

This is reported as a qualified negative result. At realistic budgets, drift-aware prioritisation offers a real but small advantage over checking conclusions in arbitrary order, and a single-feature heuristic captures more than the learned model does. The heuristic that works best is the one the attribution analysis predicts:

count how often the classes on the justification path have been re-parented, not how often the endpoints have. Section 5 draws the practical consequence.

4.6. RQ5: What This Means for a Workload

The results so far are stated per entailment, which is the right unit for measurement and the wrong one for a system owner. Table 9 restates them per query, using the cohort-style subsumption queries defined in Section 3.7. Of the 3,987 queries with at least ten members alive at the start of 2022, 21.9% lose at least one member of their answer set within three years. Averaged over all queries, 3.1% of members are lost; among the queries that change at all, the median loses 4.0% of its members, and the ninetieth percentile of loss across all queries is 4.8%. The dispersion matters more than the average: a fifth of queries lose members, and CL's queries lose a median of 10.6% of their members when they change at all.

Table 9. Query-level drift for cohort-style subsumption queries.

Ontology	Queries	Median size	Changed (%)	Mean members lost (%)	Median loss, changed (%)	Losses visible to deprecation (%)
SO	177	23	3.39	0.05	0.39	100.00
PATO	91	22	14.29	3.21	5.56	41.09
HP	1,376	25	26.24	3.30	3.57	39.67
DOID	671	27	24.14	2.58	2.23	22.67
CL	276	25	35.51	8.50	10.58	71.97
ENVO	200	24	21.50	4.59	7.14	25.86
OBA	295	19	18.31	3.05	4.01	47.26
UBERON	901	26	15.21	1.93	4.35	4.02
Pooled	3,987	24	21.92	3.13	4.04	42.08

Queries have at least ten members alive at $\tau_0 = 2022.0$ and are observed over the next three years. The final column is the share of lost members whose loss a consumer could have detected by watching obsoletion and removal of the two classes named in the entailment.

The final column tempers the principal result of Section 4.3 and is reported for that reason. At the level of individual incident entailments, deprecation signals detect 14.5% of invalidations. At the level of answer-set members lost from a live query cache, they detect 42.1%, because this population includes long-standing entailments whose losses are more often endpoint obsoletions than mid-path re-parentings, and because a single obsoleted superclass removes many members at once. The two numbers are consistent and measure different things. The conclusion is unchanged in direction and weaker in magnitude: a system that watches only deprecation signals still misses 57.9% of the answer-set members it loses, and the miss rate ranges from nothing in SO to 96.0% in UBERON, so calibrating on one ontology's experience is unsafe.

A worked example. To make the consequence concrete, consider a use of the Human Phenotype Ontology [36] that is common in deployed tooling: retrieving all descendants of a category to assemble a phenotype set, as phenotype-driven diagnostic and variant-prioritisation systems do [47, 48]. In the HP release of February 2021, the entailment that "transient ischemic attack" (HP:0002326) is a subclass of "abnormal vascular morphology" (HP:0025015) ceased to hold, although both classes remained present and non-obsolete and the subclass kept its direct parent "stroke" (HP:0001297). All four intermediate classes on the justification path also remained present and non-obsolete; what changed were the edges between them. A system that had cached, or that re-issued, the query "all descendants of abnormal vascular morphology" would from that release silently cease to retrieve transient ischemic attack, and no deprecation signal would have warned it. The entailment was restored in the December 2022 release, so the cached answer set was wrong for almost two years, an instance of the reversed invalidations quantified in Section 4.7. Each such change is small, but each is invisible to the monitoring strategy that a consumer would naturally adopt, and across the corpus such losses affect about one subclass query in five within three years. This is the practical content of the re-derivation recommendation: a consumer who re-derives sees the change, whereas one who watches for deprecations does not.

4.7. Robustness Controls and Release-Artefact Audit

The following controls test whether the main estimates depend on a single ontology, on the time axis, on a small number of releases, on invalidations that are later reversed, or on changes in how releases are packaged.

Leave-one-ontology-out. Pooled three-year survival moves within 0.802 to 0.866 and five-year survival within 0.760 to 0.845 when any single ontology is dropped, against baselines of 0.823 and 0.786. The pooled estimate is not an artefact of one contributor.

Time-axis sensitivity. Recomputing survival on the release-index axis rather than in calendar years preserves the ordering of ontologies, with UBERON and ENVO lowest and HP and SO highest at ten releases (release_index_survival.csv). Conclusions do not depend on the choice of axis.

Size and growth confound. Across the eight ontologies, the Spearman correlation between the hard-invalidation rate and the growth ratio (active native classes at the last release divided by those at the first) is 0.619, and that with final size only 0.190 (size_growth.csv). Faster-growing ontologies invalidate more. With n = 8 this is suggestive rather than significant and nothing is built on it.

Concentration in mass-revision releases. This control qualifies the interpretation of the survival estimates. Invalidations are heavily concentrated in a small number of releases: the single largest release accounts for 77.9% of PATO's hard invalidations, 58.7% of CL's, 50.5% of SO's and 47.9% of OBA's, with only HP at 11.1% and UBERON at 19.5% approaching an even spread. Excluding each ontology's single largest invalidation release raises five-year survival substantially, as Table 10 shows: CL from 0.713 to 0.850, PATO from 0.903 to 0.978, OBA from 0.838 to 0.919, ENVO from 0.713 to 0.817 and UBERON from 0.683 to 0.738. Generalising the control to leave out each release in turn gives the same picture: five-year survival for CL ranges from 0.713 to 0.850 and for OBA from 0.838 to 0.919 depending on which single release is dropped. Entailment attrition is therefore not a smooth, memoryless decay process; it is episodic, dominated by occasional bulk restructuring events.

Release-artefact audit. Some of these bulk events are not curation at all but changes in what the release file contains, and manual inspection identifies two such incidents. Inspection cannot establish that the remaining snapshots are clean, so all 311 are audited under one uniform rule set, reported in the right-hand columns of Table 10 and released as artefact_releases.csv. A release is flagged as a packaging artefact if the release product named in its data-version header differs from the previous snapshot's, if a fall of more than 2% in the term count is recovered within three snapshots, or if the total term count falls by more than 5% while the native active class count moves by less than 1%. It is flagged separately as gap-affected if the interval since the previous snapshot exceeds four times that ontology's median interval and is at least a year, because deaths dated at such a release are interval-censored over a window too long to attribute to it.

The audit flags 8 packaging artefacts and 15 gap-affected releases out of 311. Four of the eight flagged releases belong to the two incidents found by manual inspection: PATO's switch to the -base product at 2019-10-29 and back again, and OBA's transient collapses of the term count in 2016 and 2021. The remaining four had not been identified by manual inspection: a switch of DOID to a differently named release file in 2019, a CL snapshot in 2019 whose data-version header names cl-simple.owl and whose term count falls by 78.0%, and two further CL releases losing terms without a corresponding change in native classes. The important result is how little they matter in aggregate. Deaths dated at packaging-artefact releases are 1.2% of all hard invalidations, and excluding them moves pooled three-year survival from 0.823 to 0.825 and five-year survival from 0.786 to 0.788. The concentration reported above is therefore overwhelmingly real curation and interval censoring rather than packaging: deaths at gap-affected releases are 27.6% of the total, and excluding those instead moves pooled five-year survival from 0.786 to 0.824. Per ontology the packaging correction is negligible everywhere except PATO, where 77.9% of invalidations fall at an artefact release and five-year survival moves from 0.903 to 0.978; PATO's figures should be read with that in mind and are flagged in Table 10. The reason the correction is so small elsewhere is that the measurement does not see most of what a packaging change alters: the four product substitutions move the

number of terms in the release file by as much as 78.0% while moving the native active class count by at most 1.53% and the taxonomic entailment count by at most 3.03%.

Table 10. Concentration of invalidation in mass-revision releases, and the release-artefact audit.

Ontology	Share in largest release (%)	S(5 yr) all	S(5 yr) excl. largest	Artefact releases	Gap releases	Hard inval. at artefact (%)	S(5 yr) excl. artefact
SO	50.5	0.936	0.967	0	1	0.00	0.936
PATO	77.9	0.903	0.978	2	2	77.88	0.978
HP	11.1	0.942	0.948	0	0	0.00	0.942
DOID	23.0	0.815	0.851	1	0	6.40	0.825
CL	58.7	0.713	0.850	3	1	3.24	0.716
ENVO	38.7	0.713	0.817	0	2	0.00	0.713
OBA	47.9	0.838	0.919	2	2	1.10	0.839
UBERON	19.5	0.683	0.738	0	7	0.00	0.683
Pooled	—	0.786	—	8	15	1.19	0.788

Columns 2 to 4 give the share (%) of an ontology's hard invalidations occurring at its single largest invalidation release and five-year survival with and without that release. Columns 5 to 8 give the audit: the number of snapshots flagged as packaging artefacts and as gap-affected, the share of hard invalidations dated at packaging-artefact releases, and five-year survival with those excluded.

Two of the gap-affected releases are the largest invalidation releases of their ontologies: ENVO's follows a 465-day sampling gap and a fall of 2,762 in the term count, and OBA's follows a 590-day gap. Figure 8 plots invalidations per release against the preceding sampling gap and against the change in term count; across releases, the Spearman correlation between the gap preceding a release and the number of invalidations at it is 0.65 in ENVO, 0.46 in HP and 0.46 in OBA. CL's largest release, by contrast, follows a 63-day gap with a stable term count and appears to be genuine curation. The survival curves in Figure 2 should therefore be read as summarising a process punctuated by shocks, most of which are real reorganisations observed through a coarse sampling window, and this distinction matters for practice: the

risk facing a deployed system is not gradual erosion but an unpredictable re-validation event when its ontology undergoes a major reorganisation.

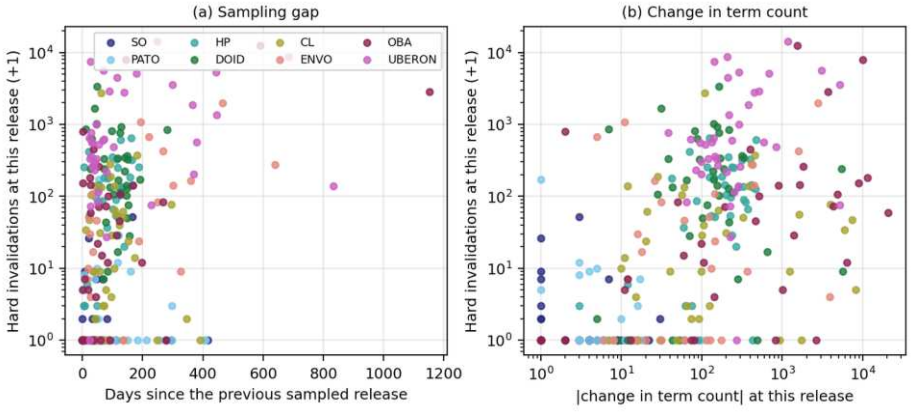


Figure 8. Hard invalidations per release against sampling gap and change in term count. The panels show (a) the number of days since the previous sampled release and (b) the absolute change in term count at that release. Counts are shown as n + 1 on logarithmic axes. Several of the releases that dominate the survival curves follow unusually long sampling gaps or large changes in the composition of the release artefact.

Reversed invalidations. An entailment that ceases to hold can hold again at a later release, when a restructuring is partly undone or a class is moved back. Of the 128,950 incident hard invalidations, 16,873 (13.1%) are followed by the same pair being entailed again, and 32,988 incident lifetimes (5.3%) begin with such a re-derivation. Reversal is rare in HP (2.2%) and SO (2.0%) but common in DOID (34.5%), ENVO (27.8%) and CL (20.8%), and in PATO (65.0%) it coincides with the switch to and from the -base product identified above. Counting only permanent invalidations, by joining each interrupted lifetime to its continuation, raises pooled five-year survival from 0.786 to 0.812, with the largest movements in DOID (0.815 to 0.868), ENVO (0.713 to 0.787) and CL (0.713 to 0.768). For a system that caches conclusions an interrupted lifetime is still an error while it lasts, so the primary analysis counts it; the permanent-only figures, released as robustness_reversals.csv, bound how much of the measured attrition is transient.

5. Discussion

5.1. What the Results Mean for System Design

Three findings bear directly on how ontology-backed systems should be built.

First, conclusions expire, and at a rate that is not negligible. One entailment in five lapses within five years, and two in five lapse or lose the derivation that justified them; one query in five loses at least one member of its answer set within three years. Any system that caches derived conclusions across a horizon of years without re-derivation is accumulating silent error at a measurable rate. This is not an argument against caching; it is an argument for attaching an explicit derivation date and ontology version to every stored conclusion, and for treating conclusions as assertions with a validity interval rather than as facts.

Second, and more sharply, deprecation-watching is insufficient in the biomedical ontologies studied here, and extending it to the whole justification path does not remedy this. It is the natural defence and it addresses 14.5% of invalidations at the entailment level and 42.1% of lost answer-set members at the query level. Adding a check on the parent set of the subclass raises entailment-level detection to at least 43.8%, which is worth doing where nothing better is possible, but still leaves up to 56% of invalidations undetected. A system that follows replaced-by pointers and flags obsoleted terms will still hold most of its stale conclusions undetected, because those conclusions name classes that remain active and are justified by paths whose classes also remain active. The failure is structural: validity depends on which edges connect the classes, and edges carry no deprecation signal. The design implication is that either the justifying path must be stored alongside the conclusion and its edges re-checked, or the conclusion must be re-derived. Given that re-derivation over a taxonomic fragment is computationally inexpensive — a transitive closure over a graph of tens of thousands of nodes — the recommendation for this class of inference is to re-derive the closure on every ontology update and to compare it with the previous closure, so that every stored conclusion that no longer follows is identified, rather than to rely on change signals.

Third, volatility scoring is not a substitute for re-derivation. It was the natural candidate for a cheaper alternative, and it does not deliver one. Within an ontology it beats a permutation null in one of the six ontologies where the comparison is evaluable, once a calendar-time proxy is removed and the six comparisons are corrected for multiplicity; across ontologies it fails; conditional on ontology its ranking on later data is inverted; in a budget-constrained deployment it recovers 1.30 times as many invalidations as arbitrary ordering at a 10% budget, and a single-feature heuristic based on the revision history of the intermediate classes on the justification path does better still, at 1.61. Section 5.3 sets out where such scoring still has a role.

5.2. Why Prediction Fails

The reason why the predictive results are weak is itself informative. The concentration control in Section 4.7 shows that invalidation is dominated by episodic bulk restructuring. Such events are curatorial decisions — a reorganisation of a branch, an alignment with an upstream ontology, a change in modelling convention — and their timing is not encoded in the structural features of individual entailments. A model trained on structural position is attempting to predict the consequences of decisions that have not yet been made and whose causes lie outside the artefact. That the resulting signal is weak, ontology-specific and unstable across periods is to be expected. Structural features may indicate which entailments are exposed if a restructuring occurs, but they cannot indicate whether or when it will.

The diagnostics of Section 4.4 sharpen this. The models are not uninformative on data adjacent in time to their training fold, where they reach AUROC between 0.658 and 0.740; they invert on data from a later period. What changes between the two periods is which reorganisations happened, and to which regions of the hierarchy. This is also why a within-ontology model that is allowed to see a proxy for calendar time appears to do so much better: it is not predicting fragility but locating the shocks that already happened. And it explains the failure of cross-ontology transfer. Each ontology has its own curation culture, release discipline and history of reorganisations. What made UBERON entailments fragile in one period reflects

UBERON's particular restructuring events, and the absence of cross-ontology transfer (Section 4.4) is consistent with there being no reason for that to generalise to HP.

To test whether the bulk-restructuring events identified above are themselves predictable from curation activity — as distinct from the structural features of individual entailments — release-level process features were mined from the version-control history of each repository: the interval since the previous release, the volume of commits and of commits touching the release artefact, the number of distinct editors, and a burstiness measure, each computed over the interval preceding a release so that no future information is used, over 284 releases in total. Labelling as a bulk event each release in the top decile of its ontology's hard-invalidation share (33 bulk events), a leave-one-ontology-out model reaches an AUROC of 0.507, and 0.534 with the sampling-gap feature removed; the most discriminating single feature is the gap since the previous sampled release, at 0.619, which is the interval-censoring confound of Section 4.7 rather than a curatorial signal. Within a single ontology the mean AUROC is 0.552, ranging from 0.125 to 0.815 across the eight ontologies on only four or five bulk releases each, which is too wide to indicate real structure. The features and results are released as process_features.csv and process_prediction.json. Process signals therefore do no better than the structural features of Section 4.4 at anticipating the events that drive attrition, which supports the reading that their timing is set by exogenous curatorial decisions not encoded in the observable release process.

5.3. A Drift-Aware Conclusion Record

The results support a concrete, low-cost design, which is the taxonomic specialisation of a justification-based truth maintenance record [22]. Every stored conclusion should carry three additional fields: the identifier and release version of the ontology that produced it, the date of derivation, and the justification — for a taxonomic entailment, the sequence of classes on the path that supported it. The first two make staleness detectable in principle at negligible cost. The third is what makes it detectable in practice, because it converts the path-breakage problem from invisible to checkable: a system holding the justification can test, at each new release, whether each consecutive link on the recorded path still holds, which requires one

edge lookup per link rather than a full re-derivation. Over the corpus this check detects every hard invalidation, and 38.5% of the alarms it raises are hard invalidations; the others mark conclusions that still hold but whose recorded derivation has been rewritten, which a system that caches explanations must refresh in any case. It must test the links, not the classes: Section 4.3 shows that in 97.2% of path breakages every class on the path is still present and non-obsolete.

As Section 5.1 argues, full re-derivation dominates every alternative where it is affordable, which it is for the taxonomic fragment studied here; the incremental-maintenance literature surveyed in Section 2.4 supplies the machinery for the cases where it is not. Where neither is available — for conclusions that have already been externalised into publications or downstream databases, or for systems that consume an ontology they do not control — the justification-monitoring check gives most of the benefit at a small fraction of the cost. Volatility scoring should be regarded as a last resort, used only when neither re-derivation nor justification monitoring is available, and calibrated on the ontology's own history rather than borrowed, as Section 5.1 also concludes. The record is also complementary to change-history management [10]: a change log records what was edited, whereas this record stores what was concluded and on what grounds, which is the information needed to decide whether an edit matters to a given consumer.

Scalability. The recommendation to re-derive holds at the scale of the largest ontologies. Re-deriving the entire taxonomic closure is a transitive closure over a directed acyclic graph, whose cost in these ontologies grows approximately linearly with the number of derived entailments: measured on the last corpus release of each ontology with a single-threaded, unoptimised Python implementation, a complete re-derivation takes 5–17 ms for SO, PATO, CL and ENVO, 53 ms for DOID, and 68, 104 and 108 ms for UBERON, HP and OBA (136k, 203k and 190k entailments), at a stable 0.43–0.57 µs per entailment. An ontology an order of magnitude larger would re-derive in about a second, negligible against a release cadence of weeks. The justification-monitoring check of the previous paragraph scales better still and in the dimension that matters here: it tests each edge of a stored path for continued existence — measured at 37–118 ns per lookup — so its cost is linear in the length of the justification and independent of the size of the ontology, growing only

with the number of cached conclusions. The practical ceiling on both strategies is therefore I/O and orchestration, not the graph computation, at every scale the OBO family currently reaches.

5.4. Implications for Ontology Developers

The attribution results also speak to curation practice. Because most invalidations are path breakages in which every class on the path is still present and non-obsolete, and because an edit above the subclass can sever the path without changing the subclass itself, the downstream impact of a re-parenting operation can be far larger than its local extent suggests, and it is invisible to consumers. Moving a single mid-level class can invalidate every entailment that passed through it. Release notes that enumerate obsoletions but not re-parentings would therefore understate the impact of a release on downstream consumers; this article does not examine release notes directly. Publishing, alongside each release, the set of entailments added and removed — which is cheap to compute for a taxonomic fragment, and which existing tooling such as ROBOT [13] could emit — would let consumers detect exactly the invalidations that deprecation signals currently hide. A second, cheaper recommendation follows from the artefact audit: because a change of release product is indistinguishable, to a consumer, from a mass deletion, release files should record the product they were generated from and pipelines should not silently substitute one product for another.

5.5. Threats to Validity

Fragment. As introduced in Section 1 and detailed in Section 3.2, the analysis covers the named-class taxonomic fragment as published in each release, not full description-logic entailment. Its relation to the reasoned taxonomy was measured for the three ontologies that publish both an asserted-only base product and a classified product, by comparing the two products at the latest available release (2026-08-31 for DOID, 2026-06-29 for ENVO, 2025-05-14 for PATO). Subsumptions that only the reasoner derives are 33.6% of DOID's named-class subsumptions and 28.1% of ENVO's, and none of PATO's, whose logical definitions reference external ontologies. For DOID and ENVO the corpus already contains them: the entailment counts at their last corpus release, 111,432 and 17,235, are those of the classified product rather

than of the base product, because their release artefacts materialise inferred links. Across three transitions between DOID's classified releases from 2021 to 2026, conditioned on both endpoint classes surviving so that obsoletion is excluded, inferred-only subsumptions are invalidated at 3.7 to 7.0 times the rate of asserted ones. Two consequences follow. The survival figures of ontologies whose releases materialise inferences, such as DOID and ENVO, already include this more fragile component; and for releases that publish only asserted links, the figures describe the more durable asserted hierarchy and understate what a consumer who classifies the ontology would experience, which is the sense in which they remain an optimistic bound. The comparison is released as dl_entailment_survival.csv; extending it to every snapshot of every ontology requires classifying each snapshot with a reasoner. Extending the measurement to part-of relations is likewise left to future work.

Native-class closure. The closure is taken over classes native to the ontology only, so an edit that re-routes a justifying path through an imported class would register here as an invalidation even though a consumer of the merged release would still hold the entailment. This bears directly on the article's principal result, so it is measured rather than argued. Recomputing every snapshot closure with the import closure resolved, and comparing the two entailment sets at every consecutive snapshot transition, gives 211,970 entailment disappearances under the native-only closure, of which 30 — 0.01% — are still entailed once imported classes are admitted. All 30 occur in a single ontology, DOID, where they are 0.12% of its disappearances; the other seven ontologies contribute none. Restricting attention to the disappearances in which both endpoint classes remain present and non-obsolete, which is the population from which path breakage is drawn, 30 of 154,617 are recoverable, or 0.02%. The native-class restriction therefore has a negligible effect, and the attribution result of Section 4.3 does not depend on it. This count of disappearances is a superset of the 128,950 incident hard invalidations analysed elsewhere, because it includes prevalent entailments and pairs that are re-derived at a later release, which makes the bound conservative. The measurement is in imports_recheck.csv and is reproduced by code/imports_recheck.py from local clones.

A related observation supports the same conclusion from the opposite direction: the release-product substitutions found by the audit of Section 4.7 barely move the native active class count and the entailment count. The restriction to native classes is thus what insulates the measurement from changes in how a release is packaged.

Sampling. As Section 3.1 notes, at most forty snapshots per ontology means entailments created and destroyed between sampled releases are invisible, biasing measured invalidation downward, so the survival estimates are optimistic. Because commits rather than calendar time are subsampled, the gaps between snapshots are very uneven, reaching 1,153 days in OBA. Deaths are dated at the release where absence is detected; placing them instead at the midpoint or the start of the interval moves pooled three-year survival from 0.823 to 0.819 and 0.817 respectively, so the effect is negligible overall, but for OBA, where the median interval within which an invalidation is known to have occurred is 1.6 years, five-year survival moves between 0.784 and 0.858. The artefact audit in Section 4.7 quantifies the consequence from the other direction: 27.6% of hard invalidations are dated at releases following an unusually long gap.

Re-selecting snapshots uniformly on the calendar axis over the full commit history of each repository, at the same budget of forty, removes most of that unevenness: the ratio of the largest to the median interval falls from 32.1 to 5.5 in SO, from 11.9 to 1.4 in UBERON, from 7.6 to 2.9 in PATO and from 4.7 to 2.6 in CL. It does not remove it for OBA, where the ratio falls only from 25.6 to 10.4, nor for ENVO, where it barely moves (13.5 to 13.3), because those two repositories contain only 50 and 33 commits touching the release artefact over the whole observation period. The long intervals in OBA and ENVO are therefore genuine holes in the release history rather than artefacts of subsampling, and no sampling rule can close them. This is the strongest reason to read the concentration results of Section 4.7 as describing an episodic process observed coarsely rather than a smooth process observed badly.

To establish whether the commit-sequence sampling rule drives the estimates, the entire corpus was rebuilt from these calendar-uniform snapshots; because the budget is unchanged, the comparison isolates the rule rather than the sample size. Where commits touching the release artefact are few or clustered in time, several

calendar targets fall on the same commit, so the calendar rule selects fewer snapshots (23 for SO, 24 for OBA, 30 for PATO, 32 for ENVO and 36 for CL), and the observation window is extended only for SO, HP and DOID, whose latest releases postdate the published corpus; five-year survival moves by at most 0.005 in these three ontologies. Five-year survival changes by at most 0.017 in six of the eight ontologies and is identical in DOID and ENVO. Two ontologies move, both through mechanisms documented above: PATO rises from 0.903 to 0.965 because denser sampling dilutes the packaging-artefact release that dominates its figures (Section 4.7), and OBA falls from 0.838 to 0.657 because of a gap-affected release; excluding gap-affected releases, as in the audit of Section 4.7, restores OBA to 0.788. The same exclusion raises ENVO from 0.713 to 0.831 under both rules, revealing a latent shock that the unadjusted comparison does not show. Pooled five-year survival is 0.739 under the calendar rule, or 0.772 once gap-affected releases are excluded, against 0.786 under the published rule. Table 11 reports the full comparison. Every qualitative conclusion is preserved. The pooled estimate moves toward more drift, consistent with the reading that the reported pooled figures are an optimistic bound, although the estimates for PATO, UBERON and SO move in the opposite direction.

Table 11. Survival under commit-sequence (CS) versus calendar-uniform (Cal) sampling.

Ontology	**Interval ratio (CS→Cal)**	**S(5 yr) (CS)**	**S(5 yr) (Cal)**	**Δ**	**S(5 yr) (Cal, gap-excl.)**
SO	32.1→5.5	0.936	0.941	+0.005	0.941
PATO	7.6→2.9	0.903	0.965	+0.062	0.965
HP	2.2→1.7	0.942	0.940	−0.002	0.940
DOID	2.9→1.4	0.815	0.815	0.000	0.815
CL	4.7→2.6	0.713	0.711	−0.001	0.711
ENVO	13.5→13.3	0.713	0.713	0.000	0.831
OBA	25.6→10.4	0.838	0.657	−0.181	0.788
UBERON	11.9→1.4	0.683	0.701	+0.017	0.701

Ontology	Interval ratio (CS→Cal)	S(5 yr) (CS)	S(5 yr) (Cal)	Δ	S(5 yr) (Cal, gap-excl.)
Pooled	—	0.786	0.739	−0.047	0.772

Interval ratio is the ratio of the largest to the median interval between selected commits; Δ is the five-year survival difference, computed from unrounded estimates; the final column excludes gap-affected releases as defined in Section 4.7.

Under the calendar-uniform rebuild, the pooled share of invalidations detectable by deprecation signals on the justification path rises from 16.5% to 27.9%. Most of this shift comes from OBA's gap-affected release, which records a bulk subclass-obsoletion event that the sparser commit-sequence sampling straddled (OBA: 2.0% to 54.4%). The detectable share also changes in PATO (8.8% to 25.0%), where denser sampling dilutes the packaging-artefact release, and in UBERON (16.4% to 7.6%); in the remaining five ontologies it moves by at most two percentage points.

Episodic dynamics and power. Because attrition is concentrated in bulk events, survival estimates depend on whether an ontology underwent a major reorganisation during the observation window (Table 10), and all correlations across ontologies rest on n = 8 and are reported as suggestive only.

Generalisation. The corpus comprises eight ontologies, all of the OBO family and predominantly biomedical, which share tooling and governance conventions. To probe generalisation, two plant-science ontologies curated by a different community were added, the Plant Ontology (PO) and the Plant Experimental Conditions Ontology (PECO), both from Planteome [55]. They were chosen because, unlike the Mondo Disease Ontology (MONDO) [56] or GO [57], they commit their release artefact to version control and can therefore be processed by the same pipeline. Both are themselves OBO-family ontologies, so the comparison probes generalisation across domains and curation communities, not beyond OBO conventions. The phenomenon itself is not confined to biomedicine: PECO's three-year survival is 0.673 (95% CI 0.588–0.763), lower than that of any of the biomedical ontologies, whereas PO is stable at 0.986, so conclusions expire outside biomedicine as well. The invalidation mechanism, however, appears specific to the biomedical structural ontologies studied here. In both plant ontologies invalidation was dominated by obsoletion rather than mid-path re-parenting, so a deprecation-watching consumer would have detected

83–88% of it, against 16.5% pooled over the eight biomedical ontologies. The central recommendation — which is motivated by the invisibility of path breakage — is therefore stated as evidenced for the large biomedical anatomy and disease ontologies, where it is strongest (UBERON and DOID), rather than as a universal property; it is already weaker in SO and CL, where obsoletion or removal of an endpoint class accounts for the majority of invalidations (Table 3). This plant evidence is preliminary and not conclusive: it rests on only six hard invalidations in PO and a single restructuring episode over PECO's short history, and both ontologies come from one curation community, so they sample a single alternative culture rather than several. Whether curation that proceeds mainly by obsoletion systematically restores the value of deprecation-watching is thus an open question these two ontologies raise but cannot settle. A broader, genuinely domain-diverse survey remains future work; the practical obstacle is that several candidate ontologies, including MONDO and GO, distribute their releases outside the version-control history of the repository, so that including them requires either a separate acquisition step for archived releases or rebuilding each release from the edit file with a reasoner.

Simulation, not deployment. The re-verification experiment operates on the entailment set itself, and the query-level analysis of Section 4.6 restates the same corpus in terms of subsumption queries rather than measuring a real workload with real costs. Neither supports a claim about clinical or scientific consequences; both support claims about how much of what a system holds becomes wrong, and how much of that a given monitoring strategy would catch. Connecting drift to decision outcomes requires a deployed system and is left to future work.

6. Conclusions

This article has posed a question that ontology-backed knowledge-based systems implicitly answer every day without measuring it: how long does a derived conclusion remain true? Taking the derived taxonomic entailment as the unit of longitudinal analysis, and drawing on 311 public release snapshots from eight

OBO-family ontologies spanning up to 16.3 years, 929,486 entailment lifetimes were tracked, 623,167 of them incident, and their survival estimated.

The answer is that conclusions expire measurably. Pooled five-year survival is 0.786, with a 95% interval of 0.708–0.886 when whole ontologies are resampled; counting the loss of the justifying derivation, it is 0.608. Of all hard invalidations, 71.8% arise from path breakage in which both classes named in the conclusion remain present and non-obsolete, and in 97.2% of those cases every class on the justifying path does as well; deprecation signals therefore detect only 16.5% of invalidations, watching the parents of the subclass as well detects at least 43.8%, and only re-checking the edges of a stored justification detects them all. Restated as a workload, 21.9% of cohort-style subsumption queries lose at least one member within three years, and 57.9% of the lost members are invisible to deprecation-watching. This is the article's principal practical result.

Volatility is largely unpredictable at derivation time. Pooled cross-ontology prediction is weak and deteriorates further when the evaluation is stratified by ontology; within-ontology prediction exceeds a permutation null in one of the six evaluable ontologies once a calendar-time proxy is removed and multiplicity is controlled; transfer between ontologies falls below chance in five of seven; and budget-constrained triage yields a lift of 1.30 over random ordering at a 10% budget, less than a simple count of prior revisions to the classes on the justification path. The reason is visible in the data: invalidation is dominated by episodic curatorial restructuring whose timing is not encoded in the structure of individual entailments.

The practical recommendation follows directly. For taxonomic inference, re-derive rather than monitor: re-derivation is cheap, and in the large biomedical ontologies studied here every cheaper signal misses a large share of invalidations, although Section 5.5 shows that this balance shifts in ontologies whose invalidations are dominated by obsoletion rather than re-parenting. Where re-derivation is infeasible, store the justifying path and re-check its edges, which is justification-based truth maintenance specialised to the taxonomic case; failing that, rank conclusions by the revision history of the classes on that path, calibrated on the

ontology's own history, and expect modest gains. In all cases, store conclusions with the ontology version and date that produced them, so that staleness is at least detectable in principle. For ontology developers, the corresponding recommendation is to publish, with each release, the set of taxonomic entailments added and removed (Section 5.4).

Four directions remain open. The first is to extend the description-logic survival measurement of Section 5.5 from DOID to the ontologies that publish only an asserted release, which requires classifying each snapshot with a reasoner. The second is to broaden the generalisation test beyond the two plant ontologies examined here, ideally across several curation communities, to establish whether curation by obsoletion systematically restores the value of deprecation signals. The third is to identify release-process or editorial signals that anticipate bulk restructuring, since neither structural nor release-process features examined here do so. The fourth is to move from the query-level restatement and worked example of Section 4.6 to a deployed decision-support workload, where the cost of a stale conclusion is measured directly rather than proxied by recall.

List of abbreviations

AUPRC: area under the precision–recall curve; AUROC: area under the receiver operating characteristic curve; BH: Benjamini–Hochberg; CI: confidence interval; CL: Cell Ontology; DOID: Human Disease Ontology; ENVO: Environment Ontology; GO: Gene Ontology; HP: Human Phenotype Ontology; MONDO: Mondo Disease Ontology; NCI: National Cancer Institute; OBA: Ontology of Biological Attributes; OBO: Open Biological and Biomedical Ontology; OWL: Web Ontology Language; PATO: Phenotype And Trait Ontology; PECO: Plant Experimental Conditions Ontology; PO: Plant Ontology; RDF: Resource Description Framework; ROC: receiver operating characteristic; RQ: research question; SE: standard error; SO: Sequence Ontology; UBERON: Uberon multi-species anatomy ontology

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Availability of data and materials

The datasets supporting the conclusions of this article are available in the Zenodo repository, https://doi.org/10.5281/zenodo.23064570 [58]. The record contains the entailment survival corpus, the per-release provenance tables, the extraction and analysis code, and all result tables and figures. All primary data are public, version-controlled ontology releases, and every snapshot is identified by commit hash and release date, so the corpus can be reconstructed exactly. The package README maps every table and figure to the file and command that produces it.

Reproduction proceeds in two stages. code/extract2.py and code/build3.py rebuild the snapshots and the survival corpus from local clones of the ontology repositories; code/imports_recheck.py, code/resample_calendar.py, code/process_features.py, code/dl_entailment_survival.py and code/scalability_timing.py run the analyses of Sections 5.2 to 5.5 that need the repositories, and their outputs are included. code/paper_tables.py, code/paper_figures.py, code/model_diagnostics.py, code/query_drift.py, code/artefact_audit.py, code/robustness.py and code/process_prediction.py regenerate every table, figure and in-text quantity from the released data alone. code/verify.py recomputes the survival, attribution, robustness, Cox, predictive-model and re-verification results from the corpus and checks each against the printed value over 145 assertions, and code/check_manuscript.py checks every table cell and in-text quantity printed in this article against the released tables over a further 649 assertions. The analysis is deterministic apart from wall-clock timings, and dependency versions are pinned in requirements.txt.

Competing interests

The authors declare that they have no competing interests.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Authors' contributions

EU and AU conceived the study, designed the methodology and developed the software. EU curated the data and performed the formal analysis. EU and AU wrote the original draft of the manuscript. MB contributed to the investigation, administered the project, and reviewed and edited the manuscript. AMS contributed to the investigation, validated the results, and reviewed and edited the manuscript. All authors read and approved the final manuscript.

Acknowledgements

Not applicable.

References

1. Studer R, Benjamins VR, Fensel D. Knowledge engineering: principles and methods. Data Knowl Eng. 1998;25:161–97. https://doi.org/10.1016/S0169-023X(97)00056-6.

2. Smith B, Ashburner M, Rosse C, Bard J, Bug W, Ceusters W, et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. Nat Biotechnol. 2007;25:1251–5. https://doi.org/10.1038/nbt1346.

3. Klein M, Fensel D, Kiryakov A, Ognyanov D. Ontology versioning and change detection on the Web. In: Knowledge Engineering and Knowledge Management (EKAW 2002). LNCS, vol 2473. Berlin: Springer; 2002. p. 197–212. https://doi.org/10.1007/3-540-45810-7_20.

4. Noy NF, Kunnatur S, Klein M, Musen MA. Tracking changes during ontology evolution. In: The Semantic Web – ISWC 2004. LNCS, vol 3298. Berlin: Springer; 2004. p. 259–73. https://doi.org/10.1007/978-3-540-30475-3_19.

5. Wadi L, Meyer M, Weiser J, Stein LD, Reimand J. Impact of outdated gene annotations on pathway enrichment analysis. Nat Methods. 2016;13:705–6. https://doi.org/10.1038/nmeth.3963.

6. Tomczak A, Mortensen JM, Winnenburg R, Liu C, Alessi DT, Swamy V, et al. Interpretation of biological experiments changes with evolution of the Gene Ontology and its annotations. Sci Rep. 2018;8:5115. https://doi.org/10.1038/s41598-018-23395-2.

7. Groß A, Hartung M, Prüfer K, Kelso J, Rahm E. Impact of ontology evolution on functional analyses. Bioinformatics. 2012;28:2671–7. https://doi.org/10.1093/bioinformatics/bts498.

8. Flouris G, Manakanatas D, Kondylakis H, Plexousakis D, Antoniou G. Ontology change: classification and survey. Knowl Eng Rev. 2008;23:117–52. https://doi.org/10.1017/S0269888908001367.

9. Zablith F, Antoniou G, d'Aquin M, Flouris G, Kondylakis H, Motta E, et al. Ontology evolution: a process-centric survey. Knowl Eng Rev. 2015;30:45–75. https://doi.org/10.1017/S0269888913000349.

10. Khattak AM, Latif K, Lee S. Change management in evolving web ontologies. Knowl Based Syst. 2013;37:1–18. https://doi.org/10.1016/j.knosys.2012.05.005.

11. Mahfoudh M, Forestier G, Thiry L, Hassenforder M. Algebraic graph transformations for formalizing ontology changes and evolving ontologies. Knowl Based Syst. 2015;73:212–26. https://doi.org/10.1016/j.knosys.2014.10.007.

12. Hartung M, Groß A, Rahm E. COnto-Diff: generation of complex evolution mappings for life science ontologies. J Biomed Inform. 2013;46:15–32. https://doi.org/10.1016/j.jbi.2012.04.009.

13. Jackson RC, Balhoff JP, Douglass E, Harris NL, Mungall CJ, Overton JA. ROBOT: a tool for automating ontology workflows. BMC Bioinformatics. 2019;20:407. https://doi.org/10.1186/s12859-019-3002-3.

14. Gonçalves RS, Parsia B, Sattler U. Analysing multiple versions of an ontology: a study of the NCI Thesaurus. In: Proceedings of the 24th International Workshop on Description Logics (DL 2011). CEUR Workshop Proceedings, vol 745; 2011. p. 147–57. https://ceur-ws.org/Vol-745/paper_40.pdf.

15. Konev B, Walther D, Wolter F. The logical difference problem for description logic terminologies. In: Automated Reasoning (IJCAR 2008). LNCS, vol 5195. Berlin: Springer; 2008. p. 259–74. https://doi.org/10.1007/978-3-540-71070-7_21.

16. Gonçalves RS, Parsia B, Sattler U. Categorising logical differences between OWL ontologies. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM 2011). New York: ACM; 2011. p. 1541–6. https://doi.org/10.1145/2063576.2063797.

17. Pesquita C, Couto FM. Predicting the extension of biomedical ontologies. PLoS Comput Biol. 2012;8:e1002630. https://doi.org/10.1371/journal.pcbi.1002630.

18. Cardoso SD, Pruski C, Da Silveira M. Supporting biomedical ontology evolution by identifying outdated concepts and the required type of change. J Biomed Inform. 2018;87:1–11. https://doi.org/10.1016/j.jbi.2018.08.013.

19. Qin L, Atluri V. Evaluating the validity of data instances against ontology evolution over the Semantic Web. Inf Softw Technol. 2009;51:83–97. https://doi.org/10.1016/j.infsof.2008.01.004.

20. Köpke J, Eder J. Semantic invalidation of annotations due to ontology evolution. In: On the Move to Meaningful Internet Systems: OTM 2011. LNCS, vol 7045. Berlin: Springer; 2011. p. 763–80. https://doi.org/10.1007/978-3-642-25106-1_25.

21. Cardoso SD, Da Silveira M, Pruski C. Construction and exploitation of an historical knowledge graph to deal with the evolution of ontologies. Knowl Based Syst. 2020;194:105508. https://doi.org/10.1016/j.knosys.2020.105508.

22. Doyle J. A truth maintenance system. Artif Intell. 1979;12:231–72. https://doi.org/10.1016/0004-3702(79)90008-0.

23. de Kleer J. An assumption-based TMS. Artif Intell. 1986;28:127–62. https://doi.org/10.1016/0004-3702(86)90080-9.

24. Gupta A, Mumick IS, Subrahmanian VS. Maintaining views incrementally. ACM SIGMOD Rec. 1993;22:157–66. https://doi.org/10.1145/170036.170066.

25. Volz R, Staab S, Motik B. Incrementally maintaining materializations of ontologies stored in logic databases. In: Journal on Data Semantics II. LNCS, vol 3360. Berlin: Springer; 2005. p. 1–34. https://doi.org/10.1007/978-3-540-30567-5_1.

26. Motik B, Nenov Y, Piro R, Horrocks I. Maintenance of datalog materialisations revisited. Artif Intell. 2019;269:76–136. https://doi.org/10.1016/j.artint.2018.12.004.

27. Kalyanpur A, Parsia B, Horridge M, Sirin E. Finding all justifications of OWL DL entailments. In: The Semantic Web – ISWC/ASWC 2007. LNCS, vol 4825. Berlin: Springer; 2007. p. 267–80. https://doi.org/10.1007/978-3-540-76298-0_20.

28. Horridge M, Parsia B, Sattler U. Laconic and precise justifications in OWL. In: The Semantic Web – ISWC 2008. LNCS, vol 5318. Berlin: Springer; 2008. p. 323–38. https://doi.org/10.1007/978-3-540-88564-1_21.

29. Jackson R, Matentzoglu N, Overton JA, Vita R, Balhoff JP, Buttigieg PL, et al. OBO Foundry in 2021: operationalizing open data principles to evaluate ontologies. Database (Oxford). 2021;2021:baab069. https://doi.org/10.1093/database/baab069.

30. Fernández JD, Umbrich J, Polleres A, Knuth M. Evaluating query and storage strategies for RDF archives. Semant Web. 2019;10:247–91. https://doi.org/10.3233/SW-180309.

31. Pellissier Tanon T, Suchanek F. Querying the edit history of Wikidata. In: The Semantic Web: ESWC 2019 Satellite Events. LNCS, vol 11762. Cham: Springer; 2019. p. 161–6. https://doi.org/10.1007/978-3-030-32327-1_32.

32. Mountantonakis M, Minadakis N, Marketakis Y, Fafalios P, Tzitzikas Y. Quantifying the connectivity of a semantic warehouse and understanding its evolution over time. Int J Semant Web Inf Syst. 2016;12:27–78. https://doi.org/10.4018/IJSWIS.2016070102.

33. Matentzoglu N, Goutte-Gattat D, Tan SZK, Balhoff JP, Carbon S, Caron AR, et al. Ontology Development Kit: a toolkit for building, maintaining and standardizing biomedical ontologies. Database (Oxford). 2022;2022:baac087. https://doi.org/10.1093/database/baac087.

34. Eilbeck K, Lewis SE, Mungall CJ, Yandell M, Stein L, Durbin R, et al. The Sequence Ontology: a tool for the unification of genome annotations. Genome Biol. 2005;6:R44. https://doi.org/10.1186/gb-2005-6-5-r44.

35. Gkoutos GV, Green ECJ, Mallon A-M, Hancock JM, Davidson D. Using ontologies to describe mouse phenotypes. Genome Biol. 2005;6:R8. https://doi.org/10.1186/gb-2004-6-1-r8.

36. Gargano MA, Matentzoglu N, Coleman B, Addo-Lartey EB, Anagnostopoulos AV, Anderton J, et al. The Human Phenotype Ontology in 2024: phenotypes around the world. Nucleic Acids Res. 2024;52:D1333–46. https://doi.org/10.1093/nar/gkad1005.

37. Schriml LM, Munro JB, Schor M, Olley D, McCracken C, Felix V, et al. The Human Disease Ontology 2022 update. Nucleic Acids Res. 2022;50:D1255–61. https://doi.org/10.1093/nar/gkab1063.

38. Diehl AD, Meehan TF, Bradford YM, Brush MH, Dahdul WM, Dougall DS, et al. The Cell Ontology 2016: enhanced content, modularization, and ontology interoperability. J Biomed Semantics. 2016;7:44. https://doi.org/10.1186/s13326-016-0088-7.

39. Buttigieg PL, Pafilis E, Lewis SE, Schildhauer MP, Walls RL, Mungall CJ. The environment ontology in 2016: bridging domains with increased scope, semantic density, and interoperation. J Biomed Semantics. 2016;7:57. https://doi.org/10.1186/s13326-016-0097-6.

40. Stefancsik R, Balhoff JP, Balk MA, Ball RL, Bello SM, Caron AR, et al. The Ontology of Biological Attributes (OBA)—computational traits for the life sciences. Mamm Genome. 2023;34:364–78. https://doi.org/10.1007/s00335-023-09992-1.

41. Mungall CJ, Torniai C, Gkoutos GV, Lewis SE, Haendel MA. Uberon, an integrative multi-species anatomy ontology. Genome Biol. 2012;13:R5. https://doi.org/10.1186/gb-2012-13-1-r5.

42. Kaplan EL, Meier P. Nonparametric estimation from incomplete observations. J Am Stat Assoc. 1958;53:457–81. https://doi.org/10.1080/01621459.1958.10501452.

43. Cox DR. Regression models and life-tables. J R Stat Soc Series B Methodol. 1972;34:187–202. https://doi.org/10.1111/j.2517-6161.1972.tb00899.x.

44. Klein JP, Moeschberger ML. Survival analysis: techniques for censored and truncated data. 2nd ed. New York: Springer; 2003. https://doi.org/10.1007/b97377.

45. Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, et al. LightGBM: a highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems 30 (NIPS 2017). Red Hook: Curran Associates;

2017. p. 3146–54. https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

46. Ojala M, Garriga GC. Permutation tests for studying classifier performance. J Mach Learn Res. 2010;11:1833–63. https://www.jmlr.org/papers/v11/ojala10a.html.

47. Köhler S, Schulz MH, Krawitz P, Bauer S, Dölken S, Ott CE, et al. Clinical diagnostics in human genetics with semantic similarity searches in ontologies. Am J Hum Genet. 2009;85:457–64. https://doi.org/10.1016/j.ajhg.2009.09.003.

48. Smedley D, Jacobsen JOB, Jäger M, Köhler S, Holtgrewe M, Schubach M, et al. Next-generation diagnostics and disease-gene discovery with the Exomiser. Nat Protoc. 2015;10:2004–15. https://doi.org/10.1038/nprot.2015.124.

49. Davidson-Pilon C. lifelines: survival analysis in Python. J Open Source Softw. 2019;4:1317. https://doi.org/10.21105/joss.01317.

50. Davidson-Pilon C. lifelines: survival analysis in Python, version 0.30.3. 2026. https://pypi.org/project/lifelines/0.30.3/. Accessed 2 Oct 2026.

51. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: machine learning in Python. J Mach Learn Res. 2011;12:2825–30. https://www.jmlr.org/papers/v12/pedregosa11a.html.

52. The scikit-learn developers. scikit-learn, version 1.7.2. Zenodo. 2025. https://doi.org/10.5281/zenodo.17084288.

53. Grambsch PM, Therneau TM. Proportional hazards tests and diagnostics based on weighted residuals. Biometrika. 1994;81:515–26. https://doi.org/10.1093/biomet/81.3.515.

54. Benjamini Y, Hochberg Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. J R Stat Soc Series B Methodol. 1995;57:289–300. https://doi.org/10.1111/j.2517-6161.1995.tb02031.x.

55. Cooper L, Meier A, Laporte M-A, Elser JL, Mungall C, Sinn BT, et al. The Planteome database: an integrated resource for reference ontologies, plant genomics and phenomics. Nucleic Acids Res. 2018;46:D1168–80. https://doi.org/10.1093/nar/gkx1152.

56. Vasilevsky NA, Toro S, Matentzoglu N, Flack JE, Mullen KR, Hegde H, et al. Mondo: integrating disease terminology across communities. Genetics. 2026;232:iyaf215. https://doi.org/10.1093/genetics/iyaf215.

57. The Gene Ontology Consortium. The Gene Ontology knowledgebase in 2023. Genetics. 2023;224:iyad031. https://doi.org/10.1093/genetics/iyad031.

58. Udrescu E, Bîzoi M, Udrescu A, Suduc A-M. Replication package for The Temporal Validity of Taxonomic Entailments in Evolving Biomedical Ontologies, version 1. Zenodo. 2026. https://doi.org/10.5281/zenodo.23064570.