Technical report v1.0 2 November 2026 28 min read

Gate Before You Publish: Deterministic Claim-Level Checks for LLM-Drafted Career Content and a Provenance Audit of a 12.8k-Record Generated Corpus

Yousof Selim
Sunway University (Lancaster University dual degree), Malaysia

75% of reworded violations still got past the checks

1. All 22 hand-written bad drafts were blocked by the seven gates (95% CI 85.1–100%), on template drafts with no language model.
2. 12 of 16 rephrased violations (75%), written after the gates, still passed.
3. 20 of 22 bad drafts were blocked by exactly one gate; before a security fix the gates caught 14 of 22, and with no gates none.
4. All 12,759 generated records were marked approved with no review date; in a 200-record random sample, one AI annotator found an editorial problem in 7.0% (4.2–11.4%).

Limits. All CareerOS results use the template provider (no LLM); cases and probes were written by the author with AI assistants; the corpus sample has one AI annotator.

Abstract

The problem AI can draft a résumé line in seconds, but a single invented number or borrowed achievement makes it unusable. CareerOS lets code, not the model, decide what may be published.

What I did I re-ran the project's 47-case evaluation on a clean checkout, switched each of its seven gates off in turn, compared it with no gates and with the gates before a security fix, and wrote a probe of

	rephrased violations. I also re-ran a provenance audit of Adelante's 12.8k-record generated quote library and had a seeded random sample of 200 labelled by one AI annotator.
What I found	**The gates blocked all 22 hand-written bad drafts (95% CI 85.1–100%), but 12 of 16 rephrased violations (75%) still got through. Twenty of the 22 drafts were blocked by exactly one gate. In Adelante, every record was marked approved with no review date, and 7.0% of the sample (4.2–11.4%) had an editorial problem.**
Why it matters	Rule-based checks are a floor for published AI text, not a ceiling, and an "approved" flag on generated data is not evidence of review unless something records who reviewed it and when. Every CareerOS draft came from the deterministic template provider; no language model was called.

Contributions

- A claim-first drafting design in which seven deterministic gates, not the model, decide whether career text may be published, with a threat model for career content.
- A clean-checkout reproduction of the 47-case evaluation with Wilson intervals, plus an ablation, ungated and pre-fix baselines and a paraphrase probe, all on the deterministic provider with no language model.
- A read-only re-run of a provenance audit of a 12.8k-record generated corpus, with a seeded 200-record random sample and interval estimates.
- A public artifact with the cases, harness, probes, annotations, raw outputs and commands.

Keywords. hallucination · grounding · publication gates · prompt injection · attribution · data provenance · evaluation · ablation · Wilson interval · career content

1 Introduction

Tools that turn a developer's work into résumé lines, portfolio posts and LinkedIn updates are an obvious use of large language models, and an obvious place for them to do harm. A drafted line that says "cut load time by 40%" or "single-handedly built" is not a stylistic slip: it is a factual claim about a person, published under their name, read by employers. Hallucination in generated text is well documented [1], and career text has three features that make it worse than the average case. The claims are about the author, so an error reads as dishonesty rather than a model mistake. The inputs (release notes, pull-request descriptions) are written by many people and can carry instructions aimed at the model. And some of the underlying work is confidential, so even a true sentence can be the wrong sentence for a public channel.

CareerOS, a system Yousof Selim designed to keep a person's career record, evidence and published outputs consistent, takes a conservative position: no model output is trusted to decide whether a draft may be published. Claims extracted from release notes are stored as inferred and cannot back public text until a person confirms them; every draft must cite claim identifiers; and seven deterministic gates (numbers, attribution, citations, privacy, links, length and style) run before a draft can be approved, and again inside the approval transaction. This report measures what that design buys and where it stops.

The second half of the report looks at the same problem from the other end. Adelante, a motivation app Yousof Selim also designed, ships a library of short texts, of which more than 99% were generated. An internal audit on 21 September 2026 found that all 12,828 records were marked reviewed and approved, none had a review date, and two records claiming to be original Adelante writing matched wording already published elsewhere. Where CareerOS gates text before it is published, Adelante is a case of a generated corpus whose approval flags were set by default. We re-run that audit and add a random-sample estimate of what the corpus contains.

Research questions

- **RQ1.** On the project's own evaluation suite, how many of the violations each gate targets do the gates block, and how much does each gate contribute? (Sections 5.2–5.4)

- **RQ2.** How well do the gates block rephrased violations written after the gates and the suite existed? (Section 5.5)
- **RQ3.** In a generated corpus whose approval flags were set by default, what does the provenance record show, and how often do records have an editorial problem? (Section 6)

In brief: the gates blocked 22 of 22 suite drafts, mostly with one gate per draft (RQ1); they blocked 4 of 16 rephrased violations (RQ2); and every one of Adelante's 12,759 records still claims review it never received, with an estimated 7.0% (4.2–11.4%) carrying an editorial problem (RQ3).

What this report is not

Every CareerOS number below comes from the gate code and from CareerOS's deterministic provider, which builds drafts from templates and never calls a model. No paid language-model API was used. These numbers therefore measure the gates on hand-written adversarial drafts and the template pipeline on synthetic release notes; they are not a measurement of how an LLM-backed pipeline behaves. A live evaluation is listed as future work and needs the owner's approval because it costs money (Section 9). The cases, probes, harness and annotations were all produced by the author working with AI coding assistants; none of the evidence is independent of the system's designers.

Contributions

- A description of a claim-first drafting pipeline in which seven deterministic gates, not the model, decide publishability, with a threat model specific to career content (Sections 3–4).
- A reproduction of the project's 47-case evaluation on a clean checkout, with Wilson 95% intervals, plus an ablation, an ungated baseline, a historical baseline (the gates before a security fix) and an exploratory paraphrase probe (Section 5).
- A read-only re-run of Adelante's provenance audit on the original and current corpus, and a seeded uniform sample of 200 records with rule-based and single-annotator classes and intervals (Section 6).
- An artifact with the cases, harness, probes, annotations, raw outputs and commands (Artifact section).

---

2 Related work

Hallucination and attribution

Ji et al. survey hallucination across summarisation, dialogue and other generation tasks and distinguish intrinsic errors (contradicting the source) from extrinsic ones (unverifiable from it) [1]. Both appear in career text: a wrong release year contradicts the evidence, while an invented user count is simply unsupported. Several lines of work score generated text against sources. FActScore splits long-form output into atomic facts and reports the share supported by a knowledge source [2]. ALCE benchmarks generation with citations and measures whether cited passages support each statement [3]. Rashkin et al. define Attributable to Identified Sources (AIS), a human-evaluation framework asking whether a statement is supported by a named source [4]. TRUE compares automatic factual-consistency metrics on a common footing [5]. SelfCheckGPT detects hallucination without external sources by checking consistency across sampled outputs [6], and RAGAS evaluates retrieval-augmented pipelines, including the faithfulness of answers to retrieved context, without reference answers [7].

CareerOS borrows the claim-level framing of FActScore and AIS: a draft is decomposed into statements that must each be backed. It differs in where the judgement happens. The metrics above score outputs, usually with a model or a human in the loop, so that systems can be compared. The gates here block outputs, using only string rules over the draft and the structured claims it cites, so that a decision can be explained to the person and reproduced exactly. The cost of that choice is coverage: a rule can check that every number in a draft is backed, but it cannot check that "added cloud sync" is entailed by the evidence. Section 5.5 measures that gap directly.

Prompt injection

Perez and Ribeiro showed that simple instructions in user input ("ignore previous instructions") redirect model behaviour [8]. Greshake et al. introduced indirect prompt injection, where the instructions sit in data the application retrieves rather than in the user's message [9]; release notes and pull-request bodies are exactly that kind of data. Liu et al. formalise injection attacks and benchmark five attacks and ten defences across ten models and seven tasks [10], and OWASP lists prompt injection first among risks for LLM applications [11]. CareerOS's response is architectural rather than detective: evidence is fenced as data, the model has no tools that act, outputs are schema-validated, and the gates check outputs regardless of what the input said. The links gate in particular targets the exfiltration and spam pattern of a model being talked into adding an attacker's URL.

Documentation and provenance of corpora

Datasheets for datasets [12], data statements [13] and model cards [14] argue that collections and models should ship with documentation of how they were made and checked. The Data Provenance Initiative audited more than 1,800 text datasets and found

widespread licence omission and misattribution [15]. The Adelante audit is a small, single-product instance of the same concern, with a twist: the corpus was not scraped but generated, so "provenance" means whether the record of review is truthful, and whether text labelled original is in fact original. Near-duplicate search in the audit uses MinHash candidate generation [16] followed by exact Jaccard verification.

Position of this report

The factuality work above asks how to score generated text against sources; the injection work asks how to stop inputs steering a model. Less has been reported on the deterministic policy layer that sits between a generator and publication, and on how such a layer behaves under ablation and under paraphrase. This report contributes a small, fully reproducible measurement of one such layer, not a new metric or defence, and pairs it with a corpus where that layer was missing. Its findings are about one system and author-written tests; they are meant to sharpen the question for larger studies, not to settle it.

Statistics

All proportions are reported with Wilson score intervals [17], which behave sensibly at 0 and at n, where the normal approximation fails; Brown, Cai and DasGupta recommend them for small n [18]. With 22 adversarial drafts, a perfect 22/22 has a lower bound of 85.1%, which is the honest summary of "all of them".

---

3 Problem and threat model

The system receives an event (a published release or a merged pull request) with its evidence, and must produce draft text for up to eight channels (résumé bullet, portfolio news, project summary, README line, LinkedIn post, LinkedIn profile, short and long bio). The person reviews and approves drafts; approved text is published or exported. The goal is that **no approved text states something the person has not confirmed, credits them with work they did not do, reveals something they marked private, or links somewhere they did not supply.**

Assets and adversaries

- **The person's reputation.** Threatened by the drafting step itself, with no adversary: a model that rounds up, invents a metric, or upgrades "contributed to" into "led".
- **Collaborators' credit.** Threatened by sole-ownership phrasing when the record shows co-authors.
- **Confidential work.** Records can be private, document-only (`selective_export`) or public; individual claims can be marked sensitive. Threatened by drafting that ignores those labels and by links or repository names that reveal a private project.
- **Readers of public channels.** Threatened by indirect prompt injection: anyone who can write release notes or a pull-request body in a repository the person is connected to can try to make the drafted text carry their link or claim.

Assumptions

The gates assume the structured inputs are correct: the claim states (verified, user-attested, inferred, contradicted, stale), the record's visibility, the person's recorded role and the collaborator list. A person who attests a false claim defeats the gates by design; the system is a guard against its own drafting step, not against its user. The gates also assume that a number, a link, or an ownership phrase is visible in the surface text. Paraphrase that avoids those surface forms is in scope as a threat and is measured in Section 5.5.

Failure categories

The evaluation suite (Section 5) is organised by nine categories that follow from this model: genuine achievements (should be suggested and pass), trivial changes (should not be suggested), collaborative work, unsupported metrics, conflicting evidence, ambiguous authorship, prompt injection, confidential information, and duplicate events.

4 Gate design

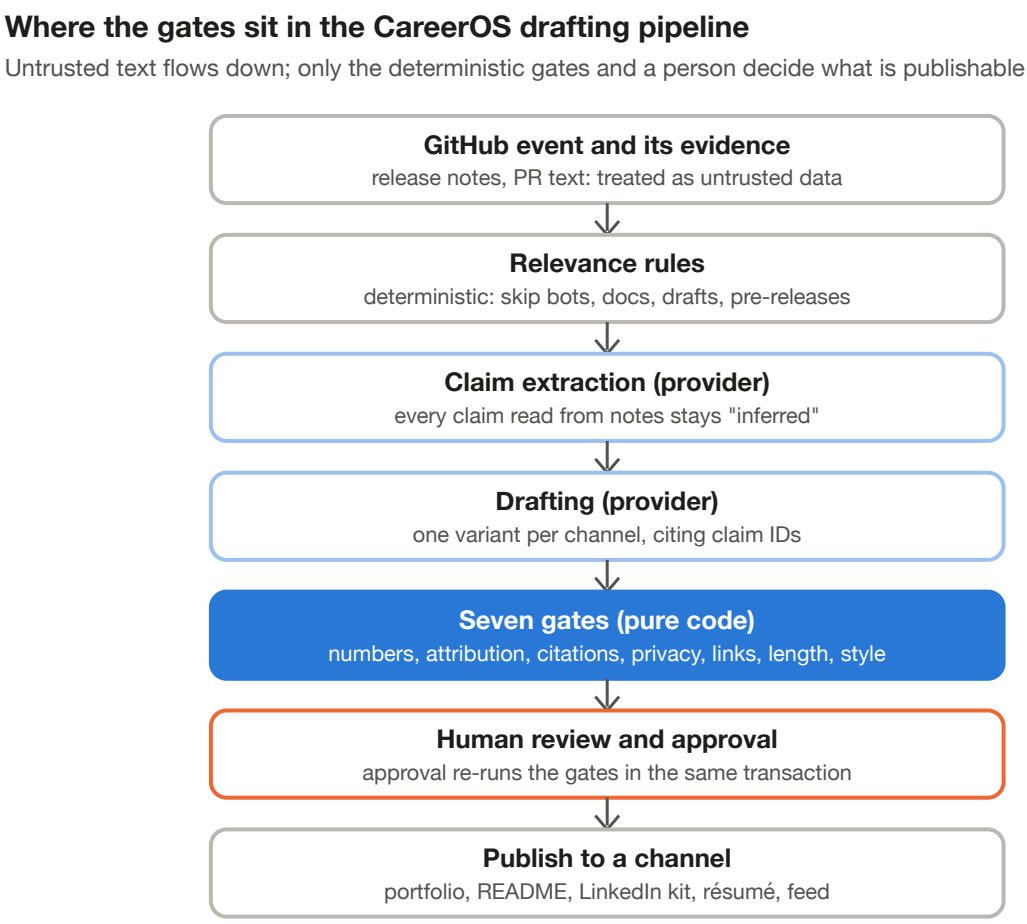


Figure 1. Where the gates sit. Claim extraction and drafting are the provider steps (templates in the deterministic provider used for every result here; a model in the optional live provider). The gates and the person decide publishability.

Figure 1 shows the pipeline. Two decisions upstream of the gates matter as much as the gates themselves. First, **claims are the unit of truth.** Source metadata ("Published release v2.3.0 of yeegz/recitation on 2026-10-12") is stored as a verified temporal claim. Anything read out of release-note text is stored as an inferred claim and stays inferred until the person confirms it; only `verified` and `user_attested` claims are publishable. Second, **drafts cite claims by identifier**, so the gates can see what a draft says it is based on.

`validateVariant` is a pure function of the draft (body and title), the cited claim identifiers, all claims on the record, the record's visibility, the attribution facts, the allowed

links and the record's locale. Each gate appends findings with a severity; a draft fails if any finding is a failure. The seven gates are:

Table 1. The seven gates in CareerOS at 63bcd5c (packages/core/src/gates.ts).

Gate	Fails when
citations	no claim is cited; a cited claim is not on the record; or a cited claim is contradicted, inferred, stale or unreviewed.
numbers	any number in the text is not backed by a publishable cited claim. Numbers are found by a tokenizer that understands currency, percentages, multipliers (3x, ×), k/m/bn, number words ("forty", "thousands", "doubled") and version identifiers. A verified temporal claim backs only its date (in every rendering of it in the record's locale and English) and its version identifier written as an identifier; a release tag cannot back the same digits used as a quantity, and the quoted title inside "Merged pull request #12 "…"" backs nothing. A quantity needs a structured numeric value on a cited claim with a matching unit.
attribution	sole-ownership phrasing (single-handedly, solely, by myself, sole developer, solo, built … alone) when the recorded role is not sole; first-person creation ("I built", "I designed") when there are collaborators or the role is unknown; leadership verbs (led, spearheaded, headed, founded, architected) unless the role is lead or sole. On LinkedIn, an uncredited collaborator is a warning.
privacy	the record is private (any channel) or document-only (public channel); a sensitive claim is cited on a public channel; or the text contains a link into, or the owner/name of, a repository whose evidence is private for this channel.
links	the text contains any link not on the allow-list. Links are found as full URLs, protocol-relative `//host`, `www.` hosts, `host.tld/path` written without a scheme (excluding file names such as README.md), and every Markdown link target, compared without scheme or trailing slash.
length	a title is multi-line or over 200 characters; the body is empty; a single-line channel has a line break; or the body exceeds the channel's hard word or character limit.
style	never fails. Warns on hype phrases, first person in résumé lines, and too many hashtags.

Several of these rules were written in response to specific failures. Commit `70b84e6` ("M4: close the privacy, numbers and links gate holes", 20 October 2026) followed an internal security review that found eight ways past the earlier gates: a number quoted

from a merged pull-request title, a release tag reused as a quantity, four link forms without a scheme or hidden in Markdown, and two ways to name a private repository. Its commit message records that the earlier gates caught 14 of the 22 adversarial drafts; Section 5.4 reproduces that number.

The deterministic provider deserves a description because every pipeline result depends on it. It extracts bullet lines from release notes as inferred claims, and it drafts each channel from a template filled only with verified or attested claims and source metadata. It never copies note text into a draft. That is a strong structural defence on its own, and Section 5.3 shows that it, not the gates, is why the pipeline-level injection and unsupported-number measures are zero.

---

5 Evaluation

5.1 Setup

The suite is `scripts/eval/cases.ts` at commit `63bcd5c`: 47 hand-labelled cases in nine categories and four kinds. Fourteen relevance cases run one GitHub event through the relevance rules against a human label (suggest or not). Twenty-three gate cases are hand-written drafts: 22 adversarial drafts that must fail a named gate and one good draft that must pass. Six pipeline cases run synthetic release notes through extraction, drafting and the gates, with a list of strings that must never appear in a draft that passes. Four duplicate cases check rollup keys. The cases were written by the author with AI assistants, as were the gates, and eight of the 22 adversarial drafts were added in the same commit as the fixes they test (Section 8).

The runner computes relevance precision and recall (thresholds 0.8), the unsupported-claim rate (numbers in publishable pipeline drafts not accounted for by the metadata; threshold 0), the gate catch rate (adversarial drafts failed by their named gate; threshold 1.0) and injection leaks (forbidden strings in publishable drafts; threshold 0). We ran it on 2 November 2026 from a fresh `git worktree` of `63bcd5c` with dependencies installed from the lockfile and `ANTHROPIC_API_KEY` unset; the runner's `--live` mode was not used. A separate harness (`gate-study.ts`) re-derives the same measures and adds the ablations. Because each gate only appends its own findings and no gate reads another's verdict, dropping one gate's findings and recomputing the verdict is equivalent to switching that gate off; no gate code was patched. Every command, with the commit and environment, is listed in the artifact README, so anyone with repository access can replicate the run.

5.2 RQ1: the suite on the deterministic provider

Table 2. pnpm eval at 63bcd5c, deterministic provider, 2 November 2026. Intervals are Wilson 95%.

Measure	Result	Threshold	95% CI
Cases passing	47/47	all	92.4–100%
Relevance precision (tp 5, fp 0)	1.00	≥ 0.8	56.6–100%
Relevance recall (tp 5, fn 0)	1.00	≥ 0.8	56.6–100%
Relevance specificity (tn 9, fp 0)	9/9	not set	70.1–100%
Gate catch rate (named gate fails)	22/22	1.0	85.1–100%
Good gate draft passes	1/1	all	not meaningful
Unsupported-claim rate, publishable drafts	0/20	0	0–16.1%
Injection leaks, publishable drafts in cases with forbidden strings	0/16	0	0–19.4%
of which the two prompt-injection pipeline cases	0/8	0	0–32.4%
Duplicate cases	4/4	all	51.0–100%

The run printed "PASS: every threshold met". Of the 24 pipeline drafts, 20 were publishable; the four that were not are the private-record case (`conf-4`), correctly blocked by the privacy gate. The intervals are the main message of Table 2: a perfect score on five positive relevance cases is compatible with a true precision as low as 57%, and 22/22 adversarial drafts with a true catch rate as low as 85% on drafts like these. The suite is a regression test that the rules do what they were written to do; it is too small to estimate how often they would fail in use.

5.3 RQ1: ungated baseline

With every gate off, none of the 22 adversarial drafts is blocked (0/22, 95% CI 0–14.9%). This is true by construction, since the drafts are hand-written to contain the violation, and it is the reference point for the ablation. The pipeline result is more informative. Ungated, all 24 pipeline drafts become publishable and the only leaks are the four drafts of the private record (every draft of `conf-4` counts as a leak). The unsupported-claim rate stays

at 0/24 and none of the injected strings ("300", "revenue", "evil.example", "SYSTEM", …) appears in any draft, gated or not.

The reason is the template provider: it drafts only from verified claims and metadata and never copies note text, so the injected instructions and metrics have no path into a draft. The pipeline-level "0 unsupported, 0 leaks" in Table 2 is therefore evidence about the templates, not about the gates. It would become evidence about the gates only when a model, which can copy anything, writes the drafts. We flag this because it is the easiest number in the project to over-read.

5.4 RQ1: ablation and historical baseline

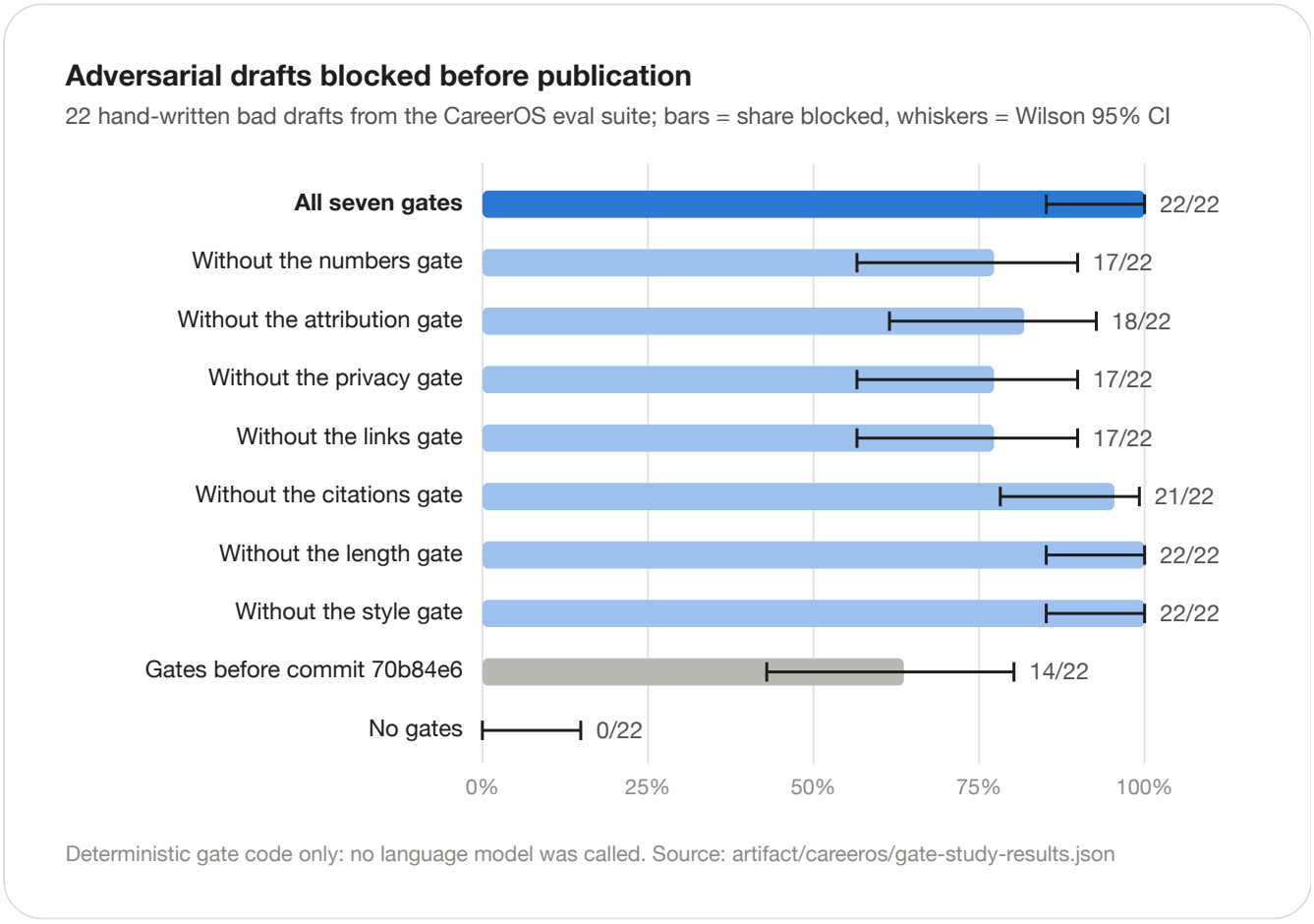


Figure 2. Share of the 22 adversarial drafts blocked (any gate fails) with one gate switched off, with the gates as they were before commit 70b84e6, and with none. Deterministic gate code only; no model was called.

Table 3. Leave-one-out and single-gate results on the 22 adversarial drafts, plus the effect on the six pipeline cases.

Gate	Blocked with this gate off	Drafts that get through	Blocked by this gate alone	Pipeline effect when off
numbers	17/22	met-1, met-2, num-pr-title, num-tag-as-metric, con-3	7	none
attribution	18/22	col-1, col-2, col-3, amb-3	4	none
privacy	17/22	conf-1, conf-2, conf-3, conf-private-repo-link, conf-private-repo-name	5	4 private drafts publishable (4 leaks)
links	17/22	inj-3, inj-www, inj-bare, inj-protocol-relative, inj-markdown	5	none
citations	21/22	con-1	3	none
length	22/22	none	0	none
style	22/22	none	0	none

Three findings follow. First, **there is almost no redundancy.** Twenty of the 22 adversarial drafts are blocked by exactly one gate, so each of the five substantive gates is the only line of defence for its own category; switching off numbers, privacy or links lets five drafts through (blocked share 77%, CI 56.6–89.9%). Second, the one overlap is instructive. A metric cited from an inferred claim (`met-3`) and a stale claim on a public channel (`con-2`) are blocked by both citations and numbers, because the numbers gate only accepts numbers from publishable claims; the citation state rule is enforced twice, once directly and once through the numbers it would otherwise back. Only `con-1`, which cites a contradicted claim and states no number, depends on the citations gate alone. Third, length and style contribute nothing on this suite. Style is advisory by design. Length is a real gate, but no adversarial case exercises it, which is a gap in the suite rather than evidence that the gate is unnecessary.

For the historical baseline we extracted `gates.ts` and `enums.ts` from the parent of `70b84e6` and ran the current 22 drafts through them. The earlier gates' named gate failed on 14 of 22 drafts (63.6%, CI 43.0–80.3%), reproducing the figure in the commit message. The eight misses are exactly the eight drafts added with the fix: the pull-request-title

number, the tag reused as a quantity, the four scheme-less or Markdown links and the two private-repository leaks. With the harness of that time (which passed no evidence-visibility information and a default allow-list) 15 of 22 were blocked by some gate, the extra one because a private repository link was also off the allow-list; with today's harness, 14. The comparison shows the fix doing what it claims. It also shows the limit of the method: the cases that measure the improvement were written alongside it.

5.5 RQ2: exploratory paraphrase probe

To get a first look at generalisation, we wrote 24 new drafts on 2 November 2026, after the gates and the suite existed: 16 that a careful reviewer would block and 8 benign controls a careful reviewer would let through. Labels were fixed in the probe file before the first run. The probe set was written by the author with an AI assistant, who also knew the gate code, so it is not a held-out test, and it is small. It is reported because its direction is clear.

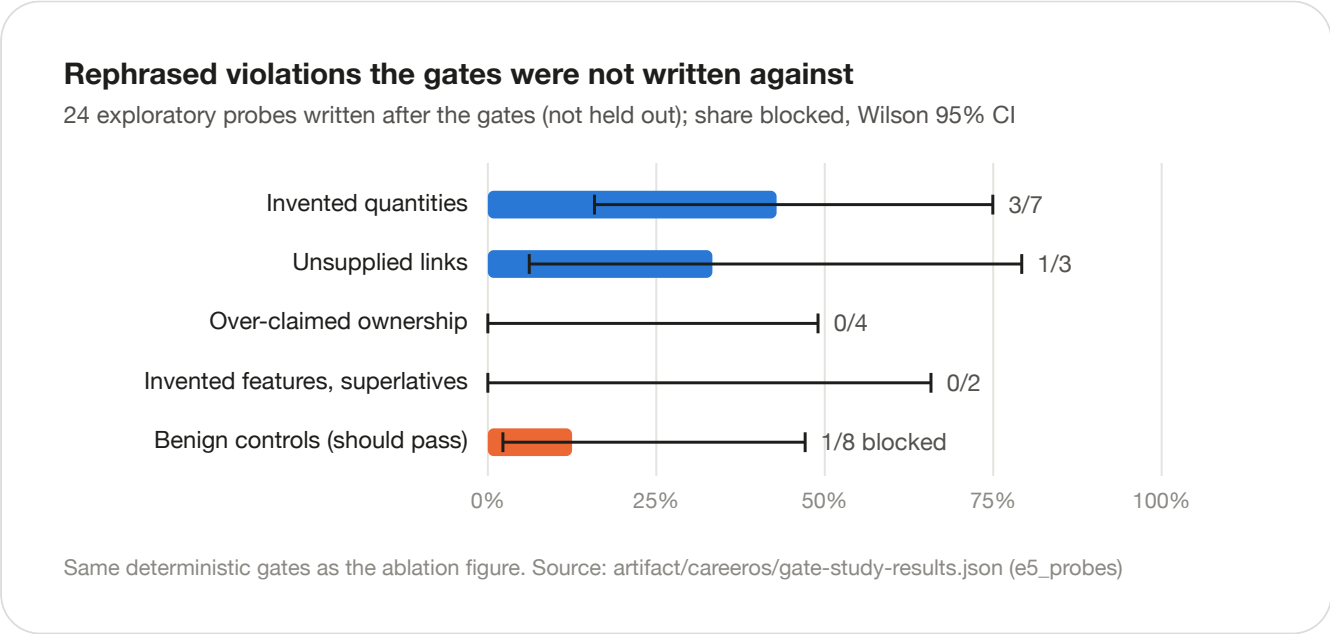


Figure 3. Exploratory probe: rephrased violations blocked by the current gates (4 of 16 in total), and benign controls wrongly blocked (1 of 8). Deterministic gates; no model was called.

The gates blocked 4 of the 16 violations (25%, CI 10.2–49.5%). They caught "forty percent", "thousands of learners", "3 crash bugs" and an upper-case `HTTPS://` link. They missed "twice as fast" and "a tenth of the memory" (comparatives not in the number vocabulary), Arabic-Indic and full-width digits (the tokenizer matches ASCII digits only), a bare domain without a path, and an e-mail address on an attacker's domain. All four rephrased ownership claims passed: "wrote the offline reader alone" (the "alone" rule requires the verb "built"), "owned … end to end", "I personally wrote" (an adverb between "I" and the verb) and "as its lead engineer" ("lead" as a noun). Both qualitative fabrications passed, as

expected: no gate checks whether "with cloud sync and a new speech model" is supported, because no gate checks entailment. Of the eight benign controls, one was wrongly blocked: "for iOS 18 and Android 15" fails the numbers gate because platform versions look like quantities.

The probe confirms what the design implies. The gates are precise about the surface forms they were written for and blind outside them. Their 22/22 on the suite and their 4/16 here are both true; the second is likely the better guide to how a model's paraphrases would fare, though with 16 probes the blocked share could plausibly lie anywhere between 10% and 50%. Most misses look cheap to patch, which is the strength of rules; each patch is likely to invite the next paraphrase, which is their weakness. Section 7 discusses how the gates could be combined with a model-based check rather than replaced by one.

---

6 Provenance audit of a generated corpus

6.1 Background

Adelante delivers short motivational and reflective texts by category and, optionally, faith. Its library was built by a generation tool and a quality gate whose numeric score rewards length, a short form and word count. On 21 September 2026 an internal audit (Adelante `docs/research/content-quality-audit.md`) examined the bundled library and found that, of 12,828 records, 12,732 were generated additions; that every record had `verified`, `reviewed`, `approved` and `active` set to true and `lastReviewedAt` null; that only 12 externally attributed records had a source URL; and that two records claiming to be original Adelante writing (author "Adelante", source "Original Adelante", original-work licence) matched wording already attributed to a published book. Those two were quarantined, together with corrections to edition references, a restored clause in a Thoreau quotation and six short forms that had changed the wording of attributed quotations. Later work (27 September) removed further weak records, bringing the quarantine to 69 and the active library to 12,759.

The audit's own manual sample was three ID-sorted records per category and faith (78 slots), which its author correctly described as a convenience sample "not a random sample from which an error rate can be estimated". This section re-runs the automated parts and adds that estimate.

6.2 Method (RQ3)

All inputs were read from git objects, so the Adelante working tree was not touched: the library at commit `3e0d06d` (before the audit; SHA-256 `728a70ab…`, identical to the "initial asset" hash the audit records) and at the current `HEAD`, `af1aea2` (asset last changed in `f7d479e`; SHA-256 `a1fcdb34…`). We ran the audit's Python script, changed only so that its input and output paths are arguments instead of the repository asset and `/tmp`, and the backend's TypeScript `auditCorpus`, which applies the server's publication-readiness rules (review evidence, provenance, rights labels, quote-meaning review) to every record.

We then drew a uniform random sample without replacement of 200 of the 12,759 current records (records sorted by ID, Python `random.Random(20261102)`). Each sampled record was joined with its rule-based findings and labelled for five editorial classes from a written codebook: **M1** a factual or prescriptive claim about health, money or employment that a specialist editor should check; **M1b** a doctrinal claim in a faith-tagged record (a review need, not an error); **M2** text unrelated to its categories (strict) or related only through its tag (liberal); **M3** wording the annotator recognises as a widely circulated saying, or a close paraphrase of one, in a record labelled original (strict and liberal); **M4** incoherent wording; and **M5** a specific quantity stated as fact with no support. The annotator was a single AI assistant working under Yousof Selim's direction; there was no second annotator, no adjudication and no source search for M3.

6.3 Reproduction

Table 4. The audit's automated measures, recomputed on the original and current corpus.

Measure	3e0d06d (before audit)	af1aea2 (current)
Records / generated	12,828 / 12,732	12,759 / 12,663
Flags true with no review date	12,828 (100%)	12,759 (100%)
Ready for publication under the server's evidence rules	0	0
Quality score ≥ 0.99 ("saturated")	12,260	12,193 (95.6%)
External records without a source URL (of 72)	60	0
External records without a translation field	39	15
Ambiguous rights labels	18	18
MinHash candidates / verified near-duplicate pairs (Python)	26,191 / 7	25,794 / 0
Near-duplicate pairs (TypeScript prefix index)	9	1
Six-word phrases in ≥ 4 records	50	35
Most repeated six-word phrase	"the first cold step onto the", 43 records	"you are not behind you are", 18 records
The two quarantined false-original records present	yes	no

The Python run on `3e0d06d` reproduces the audit's published figures exactly (12,828 and 12,732 records, 26,191 candidates, 7 verified pairs, the 43-record phrase, 12 external records with URLs). The two scripts disagree on near-duplicate counts (7 against 9 before; 0 against 1 now) because they tokenise differently and the TypeScript scan uses an exact prefix filter where the Python one uses banded MinHash; both state that they are lexical triage, not semantic duplicate detection. The corrections are visible: every external record now has a source URL and the most-repeated template phrase is gone. The central provenance problem is not: **every one of the 12,759 shipped records still says reviewed, approved and verified, with no review date,** and none would pass the backend's own

publication rules. The backend no longer treats those flags as evidence; the shipped asset still carries them.

6.4 RQ3: random-sample estimates

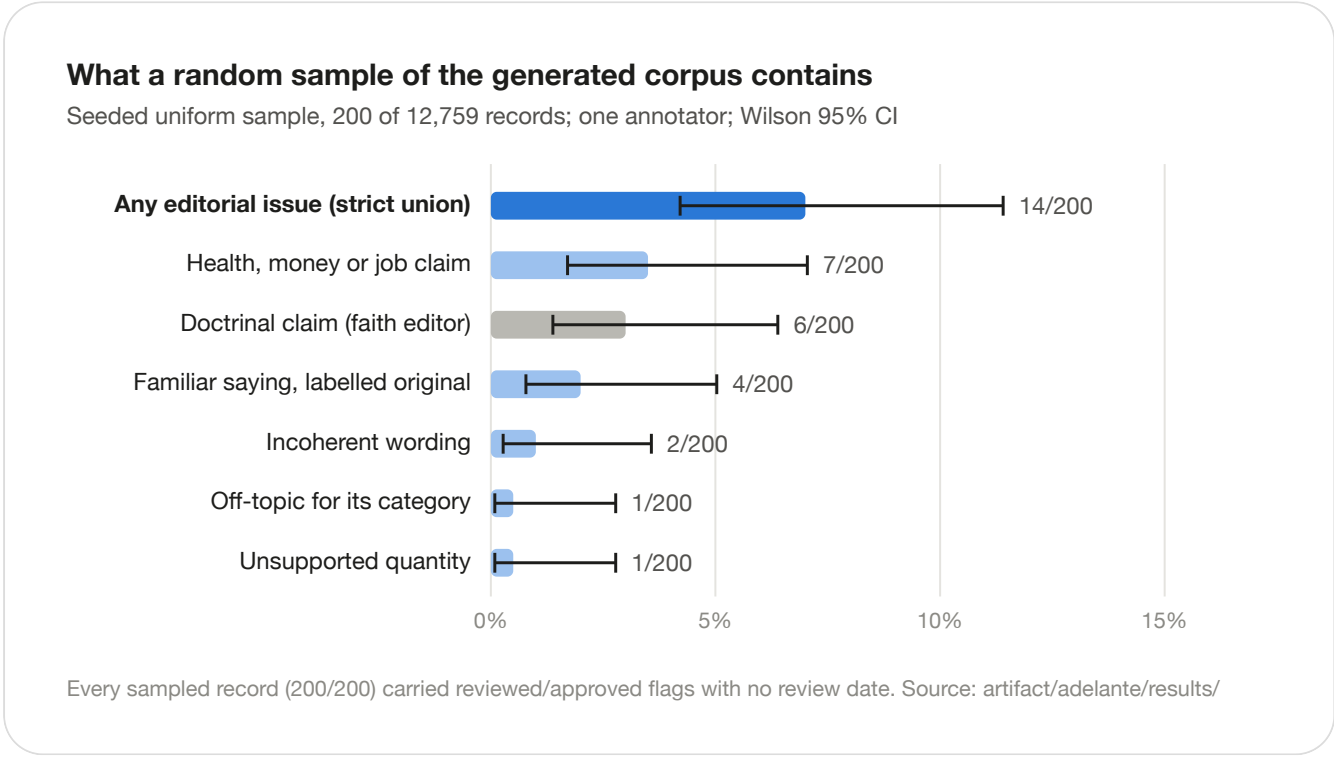


Figure 4. Issue classes in a seeded uniform sample of 200 of the 12,759 current Adelante records, with Wilson 95% intervals. One AI annotator; no second rater.

All 200 sampled records were generated originals; none of the 72 externally attributed records was drawn (expected count 1.1), so the sample says nothing about quotation accuracy. The rule-based classes agree with the census: 189/200 saturated quality scores (94.5%, CI 90.4–96.9%, against 95.6% in the census), 5/200 containing a six-word phrase used in at least four records (2.5%, against 1.3%), and 200/200 with approval flags but no review date (CI 98.1–100%).

Table 5. Annotated classes in the 200-record sample. Projections multiply the rate by 12,759 and inherit its uncertainty.

Class	k/200	Rate (95% CI)	Example (paraphrased)
M1 health, money or job claim	7	3.5% (1.7–7.0%)	a longer exhale makes the body tell the mind it is safe; invest whatever the outlook
M1b doctrinal claim (review need)	6	3.0% (1.4–6.4%)	statements about grace or honouring the Divine in faith-tagged records
M3 familiar saying, strict	4	2.0% (0.8–5.0%)	a widely shared line about starting again "from experience"
M3 familiar saying, liberal	8	4.0% (2.0–7.7%)	structural echoes of well-known sayings
M4 incoherent	2	1.0% (0.3–3.6%)	"trading down for something that lasts"
M2 off-topic, strict / liberal	1 / 5	0.5% (0.1–2.8%) / 2.5% (1.1–5.7%)	a music-practice line tagged as an Islamic reflection
M5 unsupported quantity	1	0.5% (0.1–2.8%)	"three slow breaths" as the difference between forgetting and remembering
Any of M1, M2 strict, M3 strict, M4, M5	**14**	**7.0% (4.2–11.4%)**	about 890 records (538–1,455 from the interval)

The strict union excludes doctrinal claims, which need a faith editor but are not errors, and the liberal variants. Read plainly, and subject to the single-annotator caveat: most of the generated corpus is short, on-topic and harmless, and a minority large enough to matter (on the order of several hundred records) contains advice-like claims, familiar sayings labelled as original, or wording that does not hold together. The familiar-saying class is the most consequential for provenance, since it repeats in miniature the two quarantined records: a generator can reproduce well-known lines and label them original, and neither a model's declaration nor a lexical similarity score establishes otherwise. The 2.0% strict estimate rests on one annotator's recognition with no source search, so it is likely an undercount of near-verbatim matches and an uncertain count of paraphrases.

7 Discussion

Answers to the research questions

- **RQ1.** On the suite, the gates block all 22 targeted drafts (85.1–100%), and 20 of the 22 are blocked by exactly one gate, so each substantive gate is the sole defence for its category. Without the numbers, privacy or links gate the blocked share falls to 17 of 22 (77%).
- **RQ2.** On rephrased violations they block 4 of 16 (25%, 10.2–49.5%) and wrongly block 1 of 8 benign controls.
- **RQ3.** All 12,759 Adelante records carry approval flags with no review date, and none passes the backend's own publication rules; 7.0% (4.2–11.4%) of sampled records have an editorial problem by one AI annotator's judgement.

All CareerOS answers are for the deterministic provider; none describes an LLM-backed pipeline.

Two projects, one lesson

The two systems sit at opposite ends of the same pipeline. CareerOS assumes that generated text is wrong until a rule it can explain says otherwise. Adelante's generator assumed the opposite: records were born approved, and the only score attached to them measured format. The Adelante result is what happens when the gate is a default value. The CareerOS result is what a gate can and cannot do when it exists.

What deterministic gates are good for

Their decisions are exact, explainable and stable. A person can be told which number is unbacked and which claim to confirm, the same draft always gets the same answer, and the approval transaction can re-run them cheaply. They also encode policy that no factuality metric covers: a true sentence about a private repository is still a leak, and a true sentence that says "led" when the record says "contributor" is still unfair to a collaborator. Those are not hallucination problems in the sense of the metrics in Section 2; they are publication-policy problems, and rules suit them well.

What they are not good for

They cannot judge entailment. Both qualitative probes passed, and every unsupported feature claim a model might write would pass the same way. They are also brittle to paraphrase, as the probe shows for ownership language in particular, and the suite

cannot reveal this because its drafts were written by the people who wrote the rules. The near-total lack of overlap between gates (Section 5.4) means a single gap is a complete gap for its category.

A combined design

We suggest a layered design rather than a replacement; it has not been built or tested. Keep the deterministic gates as the authority on numbers, links, privacy and recorded roles, where they are exact. Add a model-based support check (claim decomposition and per-claim entailment against the cited claims, in the spirit of FActScore and AIS) that can only add findings, never clear one, so that a manipulated checker cannot unlock publication. Widen the rule vocabulary where the probe found cheap misses (comparatives, non-ASCII digits, bare domains, e-mail addresses, nominal "lead", adverbs inside first-person creation verbs) and treat platform version numbers as identifiers. And grow the suite with drafts the gate authors did not write.

For generated corpora

The Adelante backend already encodes the right rule: a record is publishable only with a reviewer, a date and, for quotations, a meaning review bound by hash to the exact wording. The remaining gap is that the shipped asset still carries flags that say the opposite. The cheapest honest fix would be to ship the generated records with their flags set to false and to show users that a text is an original Adelante reflection, not a reviewed or expert-endorsed one; the costly fix is a review queue ordered by the classes above, starting with health, money and employment claims and familiar sayings.

---

8 Threats to validity

We group threats in the usual way and state what each one means for the results.

- **Construct validity: deterministic provider.** Every CareerOS pipeline number measures templates. The template provider never copies note text, so pipeline-level injection and unsupported-number results say nothing about a model's behaviour. Gate-level results do not depend on the provider and are the meaningful part.
- **Construct validity: what counts as a violation.** The probe labels and the Adelante codebook encode one person's view of what should be blocked or flagged; another reviewer could draw the lines differently, especially for ownership language and familiar sayings.

- **Internal validity: author-written, AI-assisted evidence.** The 47 cases, the 24 probes, the study harness and the Adelante labels were produced by the author working with AI assistants (Claude, Codex), who also wrote much of the gate code. Eight adversarial drafts were added in the same commit as the fixes they test. None of this evidence is independent of the designers, and perfect scores on it are expected rather than informative about generalisation.
- **Internal validity: ablation by finding removal.** Correct for this code because gates do not read one another's results; it would not be for a design where one gate's verdict changed another's input.
- **Internal validity: historical baseline.** The pre-fix gates were run with today's cases and two harness variants. The 14/22 figure matches the commit message, but other differences in the code of that time (outside `gates.ts`) are not reconstructed.
- **Internal validity: single AI annotator.** Adelante classes M1–M5 were labelled once, by an AI assistant, without adjudication. Familiar-saying recognition depends on what that annotator recognises, and no source search was done. Inter-annotator agreement is unknown.
- **Conclusion validity: small n.** Five positive relevance cases, 22 adversarial drafts, 20 publishable pipeline drafts and 16 probe violations give wide intervals, reported throughout. Differences between ablation conditions of one or two drafts are within noise.
- **External validity.** Each half studies one system built by one person, with English-language cases on a small set of synthetic repositories. The results describe these systems and should not be read as estimates for career-content tools or generated corpora in general.
- **Sample frame.** The 200-record sample is uniform over records, so external quotations (72 records, 0.6% of the corpus) are effectively unsampled; quotation accuracy is covered only by the earlier audit's targeted work.
- **Corpus drift.** The asset changed between the audit (12,828 records) and now (12,759). We report both where the difference matters and pin every input by hash.

---

9 Limitations and future work

Limitations

- No language model was run, so nothing here measures how often an LLM-backed CareerOS would leak, invent numbers or follow injected instructions.
- No test case or label was written by anyone outside the project; there is no held-out set and no second annotator.

- The gates do not check entailment, so unsupported qualitative claims are out of their reach by design.
- The Adelante estimates cover generated originals only, not the 72 external quotations.
- The CareerOS gate code is private, so outside readers can inspect the cases and outputs but not re-run the gates.

Future work

Live-model evaluation. The runner already supports `pnpm eval --live`, which sends the six pipeline cases to Claude through CareerOS's Anthropic provider and reports token usage and cost. It was not run, because it is a paid API and needs the owner's explicit approval. It is the single most important next experiment: with a model drafting, the pipeline-level injection and unsupported-claim measures would for the first time test the gates rather than the templates. A useful protocol is several samples per case at fixed settings, reporting leaks and unsupported numbers per draft with intervals, and recording the model identifier and prompt version, which the system already stores per generation.

Held-out cases by other people. Adversarial drafts and benign controls written by people who have not read the gate code, ideally including recruiters and developers who write their own career posts, would give a generalisation estimate the current suite cannot. The probe in Section 5.5 is a template for that set, not a substitute for it.

A second annotator. Re-labelling the 200 Adelante records independently, computing agreement (for example Cohen's kappa) and adjudicating disagreements would turn the estimates in Table 5 into defensible ones. A source search for the familiar-saying class and a stratified sample of the 72 external quotations would complete the picture.

Gate improvements. The cheap misses in Section 5.5, a length-gate adversarial case, and an add-only model-based support check are concrete follow-ups, each testable with the existing harness.

---

10 Conclusion

On CareerOS's 47-case suite, run on the free deterministic provider with no language model, the seven gates blocked all 22 hand-written bad drafts (95% CI 85.1–100%), up from 14 before a security fix; with no gates, none were blocked. The ablation shows that each substantive gate is the only defence for its category, and a small probe written afterwards shows how

narrow that defence is: 12 of 16 rephrased violations passed. The Adelante audit shows the opposite case: a generated corpus in which every record is still marked approved without any record of review, and in which one AI annotator found an editorial problem in about 7% of sampled records (4.2–11.4%).

Within those limits, the evidence supports a modest conclusion. Rule-based gates are worth having for publishing generated text about people, because they are exact and explainable, but they are a floor rather than a complete check, and their value has to be measured on text their authors did not write and on drafts a real model produced.

Artifact

The artifact at /research/gate-before-you-publish/artifact/ contains the 47 CareerOS cases, the study harness and probes, the raw output of every CareerOS experiment and of `pnpm eval`, the Adelante audit script (with a diff showing the only change), the sampling and summary scripts, the codebook and labels, the 200 sampled records, input hashes, and the script that draws Figures 1–4. Its README lists every command. CareerOS is a private repository: its gate code and providers are not included, and re-running those experiments requires access to commit `63bcd5c`. The Adelante experiments need the library at `3e0d06d` and `af1aea2`, identified by hash.

References

1. Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto and P. Fung. Survey of hallucination in natural language generation. ACM Computing Surveys 55(12), article 248, 2023. doi.org/10.1145/3571730
2. S. Min, K. Krishna, X. Lyu, M. Lewis, W. Yih, P. W. Koh, M. Iyyer, L. Zettlemoyer and H. Hajishirzi. FActScore: fine-grained atomic evaluation of factual precision in long form text generation. In Proceedings of EMNLP 2023, pp. 12076–12100. doi.org/10.18653/v1/2023.emnlp-main.741

3. T. Gao, H. Yen, J. Yu and D. Chen. Enabling large language models to generate text with citations. In Proceedings of EMNLP 2023, pp. 6465–6488. doi.org/10.18653/v1/2023.emnlp-main.398

4. H. Rashkin, V. Nikolaev, M. Lamm, L. Aroyo, M. Collins, D. Das, S. Petrov, G. S. Tomar, I. Turc and D. Reitter. Measuring attribution in natural language generation models. Computational Linguistics 49(4), pp. 777–840, 2023. doi.org/10.1162/coli_a_00486

5. O. Honovich, R. Aharoni, J. Herzig, H. Taitelbaum, D. Kukliansy, V. Cohen, T. Scialom, I. Szpektor, A. Hassidim and Y. Matias. TRUE: re-evaluating factual consistency evaluation. In Proceedings of NAACL 2022, pp. 3905–3920. doi.org/10.18653/v1/2022.naacl-main.287

6. P. Manakul, A. Liusie and M. J. F. Gales. SelfCheckGPT: zero-resource black-box hallucination detection for generative large language models. In Proceedings of EMNLP 2023, pp. 9004–9017. doi.org/10.18653/v1/2023.emnlp-main.557

7. S. Es, J. James, L. Espinosa Anke and S. Schockaert. RAGAs: automated evaluation of retrieval augmented generation. In Proceedings of EACL 2024: System Demonstrations, pp. 150–158. doi.org/10.18653/v1/2024.eacl-demo.16

8. F. Perez and I. Ribeiro. Ignore previous prompt: attack techniques for language models. NeurIPS 2022 ML Safety Workshop; arXiv:2211.09527, 2022. arxiv.org/abs/2211.09527

9. K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz and M. Fritz. Not what you've signed up for: compromising real-world LLM-integrated applications with indirect prompt injection. In Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec '23), pp. 79–90, 2023. doi.org/10.1145/3605764.3623985

10. Y. Liu, Y. Jia, R. Geng, J. Jia and N. Z. Gong. Formalizing and benchmarking prompt injection attacks and defenses. In Proceedings of the 33rd USENIX Security Symposium, pp. 1831–1847, 2024. usenix.org/conference/usenixsecurity24/presentation/liu-yupei

11. OWASP Foundation. OWASP Top 10 for Large Language Model Applications (2025 edition), LLM01: Prompt Injection. Accessed 2 November 2026. genai.owasp.org/llm-top-10

12. T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. Daumé III and K. Crawford. Datasheets for datasets. Communications of the ACM 64(12), pp. 86–92, 2021. doi.org/10.1145/3458723

13. E. M. Bender and B. Friedman. Data statements for natural language processing: toward mitigating system bias and enabling better science. Transactions of the ACL 6, pp. 587–604, 2018. doi.org/10.1162/tacl_a_00041

14. M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji and T. Gebru. Model cards for model reporting. In Proceedings of FAT* 2019, pp. 220–229.

doi.org/10.1145/3287560.3287596

15. S. Longpre, R. Mahari, A. Chen, et al. A large-scale audit of dataset licensing and attribution in AI. Nature Machine Intelligence 6, pp. 975–987, 2024. Preprint: The Data Provenance Initiative, arXiv:2310.16787, 2023. doi.org/10.1038/s42256-024-00878-8
16. A. Z. Broder. On the resemblance and containment of documents. In Proceedings of Compression and Complexity of SEQUENCES 1997, pp. 21–29. doi.org/10.1109/SEQUEN.1997.666900
17. E. B. Wilson. Probable inference, the law of succession, and statistical inference. Journal of the American Statistical Association 22(158), pp. 209–212, 1927. doi.org/10.1080/01621459.1927.10502953
18. L. D. Brown, T. T. Cai and A. DasGupta. Interval estimation for a binomial proportion. Statistical Science 16(2), pp. 101–133, 2001. doi.org/10.1214/ss/1009213286

Cite and reuse

Authorship and AI assistance. Yousof Selim designed both systems (CareerOS and Adelante) and directed this work. Much of the implementation of both systems, the experiment harness, the annotation of the Adelante sample and parts of this report were produced with AI coding assistants (Claude, Codex) under his direction and review. No paid language-model API was called for any experiment. All experiments are reproducible from the artifact; the CareerOS gate code is private and needs repository access at the stated commit. This is a technical report and has not been peer reviewed.

```
@techreport{selim2026gate,
  author      = {Yousof Selim},
  title       = {Gate Before You Publish: Deterministic Claim-Level Chec
  institution = {Sunway University (Lancaster University dual degree), M
  type        = {Technical report},
  number      = {v1.0},
  year        = {2026},
  url         = {https://yousofselim.com/research/gate-before-you-publis
}
```