Accurate but Uninformative: Checking an LLM Assessment Pipeline Without Ground Truth

Martin Macak
Jožef Stefan Institute
Artificial Intelligence Laboratory
Ljubljana, Slovenia
martinmacak@protonmail.com

Marko Grobelnik
Jožef Stefan Institute
Artificial Intelligence Laboratory
Ljubljana, Slovenia
marko.grobelnik@ijs.si

Abstract

Language models are increasingly used to assess systems against a fixed taxonomy: a list of properties, mechanisms or impacts written in advance, from which the model picks those that apply. Such taxonomies are checked for accuracy. That check needs a trusted answer to compare against, and for the downstream effects a system may have there is none. We propose a check that needs none: does the taxonomy describe different systems differently? An element asserted for almost every system carries almost no information about which system it is, and its base rate alone measures how little, as binary entropy, at most one bit per element; summed over a stage's vocabulary this is the stage's capacity. Low capacity proves that a stage cannot distinguish systems; high capacity proves nothing, so the measure is a screen rather than a verdict. Applied to a deployed five-stage AI-risk analyser over 298 system descriptions, 84 of them the AI systems that reach the later stages, it finds the mechanism stage carrying 3.0 of a possible 7 bits. One mechanism is asserted for 81 of the 84 and two are almost never asserted, and no internal consistency check can see either. The same analyser had no scope decision at all. Asking the model directly whether a submission is an AI system scores 0.94–0.97 AUC (area under the receiver operating characteristic curve) across three models; the rule it replaces, which let any submission with one asserted property through, scores 0.51–0.75, or 0.73–0.80 in its strongest form.

Keywords

large language models, structured output, taxonomy design, measurement validity, item analysis, AI system assessment

1 Introduction

Large language models (LLMs) are increasingly used to assess systems against a fixed taxonomy: a list of categories written in advance, from which the model picks the ones that apply to the case in front of it. A *risk taxonomy* names the capability properties a system may have, the mechanisms by which a capability can turn into harm, and the kinds of impact that harm can take. An assessment is the subset of those names the model asserts for one system, with a justification under each. We call each name an *element* of the taxonomy.

The system studied here is a five-stage AI-risk analyser that we built at the Jožef Stefan Institute for policy work (Section 3.1). Its input is a text description of a software system, such as the opening of a repository README. Stage 1 asserts which of 12 capability properties the text supports, Stage 2 which of 7 risk mechanisms follow, Stage 3 which of 18 impact axes are affected, and Stages 4 and 5 write causal chains, analogies and a summary. Each stage is one model call that may refer only to elements the stage above it asserted, and six deterministic checks, the validators, enforce that rule and check that every quoted span is really in the input.

`afero` is a filesystem library for the Go language, a drop-in replacement for the standard `os` package, with no machine learning in it. We submitted the first hundred words of its README. The analyser returned a full assessment: two risk mechanisms, two causal chains, a recommendation that mitigation be coordinated at the organisational level, and two historical analogies, standardised shipping containers and the electrical grid. All four validators the analyser had at the time passed. Nothing in the output is malformed. It is an AI-risk assessment of a filesystem library.

Two things went wrong, and only one of them is visible. The visible one is scope: the analyser had no step that asks whether the submission is an AI system at all, and Section 4.4 adds one. The invisible one is in the vocabulary. The mechanism the analyser asserted for `afero`, `capability_overhang`, is defined in the taxonomy as *"dual-use nature: once deployed, the capability cannot be restricted to its intended use; niche-filling proceeds faster than governance response."* That is true of nearly every deployed capability, so the element is asserted for nearly every system it is applied to. An element that is true of everything is accurate and says nothing, and repairing scope does not repair it.

Taxonomies are checked for accuracy: is each element true of the systems it is applied to? That check needs ground truth, a trusted answer to compare against. For the first stage we have one, because a project's dependency manifest says whether it uses machine learning. For the mechanisms and impacts a system may have, we know of no verified list and no obvious way to build one. In hindsight the question we ask next is obvious; in practice it was not asked, and this analyser was in use without it.

So we ask what can be measured without ground truth: does the pipeline say something *different* for different systems? A stage whose elements are asserted for almost every system, or almost none, cannot distinguish systems. A stage whose elements vary might distinguish systems or might vary at random. Low capacity proves a failure; high capacity proves nothing. We measure capacity over 298 descriptions of software systems (Section 3.3).

Contributions. (1) A *capacity* measure for one stage of a fixed-vocabulary pipeline, computed from element base rates alone, with no labels (Section 3.2). (2) A check of the measure against the one labelled result available, on Stage 1, which shows what it can and cannot do: base rate alone finds the elements to inspect, and only labels say which of them are empty (Section 4.1). (3) The finding: the mechanism stage of this analyser carries 3.0 of a possible 7 bits, with one mechanism asserted for 81 of 84 AI

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
Information Society 2026, Ljubljana, Slovenia
© 2026 Copyright held by the owner/author(s).
https://doi.org/10.70314/is.2026.sikdd.102

systems and two almost never, while the property and impact stages are not dominated (Sections 4.2 and 4.3). (4) A repair of the scope decision, the one decision that does have ground truth: asking the model directly scores 0.94–0.97 AUC against 0.51–0.75 for the property rule it replaced, and 0.73–0.80 for the strongest version of that rule (Section 4.4).

Section 2 places the work, Section 3 describes the analyser, the measure and the corpus, Section 4 reports the measurements, and Sections 5 to 7 say what they show and what follows.

2 Related Work

Work on constrained decoding asks whether a model can be made to emit valid structured output and what that costs in quality [3]: whether the schema is *satisfied*. We ask whether a satisfied schema *says* anything. Hallucination and faithfulness are the usual lens on reliability [4]; here every quote is verbatim and the pipeline still fails.

The closest relative is classical test theory [1], where an item every candidate passes has a discrimination index of zero and is cut. What is new is to treat a policy taxonomy filled in by a language model as a test instrument.

Work on AI risk taxonomies asks about *coverage*: the MIT AI Risk Repository [7] extracts 777 risks from 43 taxonomies, asking what the field misses. Coverage and discrimination are different axes: a taxonomy can cover the space completely and still describe nearly every system the same way.

3 Data and Method

3.1 The analyser

The analyser is a five-stage pipeline. Stage 1 returns the capability properties it finds evidence for, from a fixed list of 12, each with a level from 1 to 5. Levels 2 to 5 assert that the property is present and must carry a verbatim quote from the input; level 1 asserts that it is absent or minimal and carries no quote. Stage 2 asserts a subset of 7 risk mechanisms, the named routes from capability to harm. Stage 3 asserts a subset of 18 impact axes, each with a severity. Stages 4 and 5 write causal chains, historical analogies and two summary assessments. Each stage is one structured model call.

Six validators run over the output. They check that every quoted span is a verbatim substring of the input and that a stage refers only to identifiers the stage above it committed to. Two consequences matter below: a Stage 1 that asserts nothing leaves every later stage nothing to cite, and only the identifiers propagate, since no later stage reads the prose under them.

3.2 The measure

An element *fires* for a system when the stage asserts it. For Stage 1 that means the property appears in the returned list at any level, whether the level says the property is present or absent, because that is what the later stages can cite. Let p be the fraction of systems for which an element fires. The information the element carries about which system this is equals the binary entropy [6] $H(p) = -p \log_2 p - (1-p)\log_2(1-p)$: one bit at $p = 0.5$, 0.29 bits at $p = 0.95$, and zero at $p = 1$. Each stage draws on a fixed list of elements, its **vocabulary**: 12 properties, 7 mechanisms, 18 impact axes. A stage's **capacity** is $\sum H(p)$ over that vocabulary, at most one bit per element. Summing treats the elements as independent, so capacity is a ceiling, and a ceiling in one direction only: elements firing independently at random would make it maximal. Low capacity therefore shows that a stage cannot distinguish systems. High capacity does not show that it can (Section 3.4).

We call an element **uninformative** when $H(p) < 0.30$ bits, the information in a one-in-twenty event, which happens when it fires for fewer than 5.3% or more than 94.7% of systems. The rule we fixed before running screened on base rate instead, at $p \geq 0.67$ or $p \leq 0.05$. That cut charged Stage 1 for two elements that carry 0.81 and 0.86 of a possible bit, so we report the entropy screen and give both counts in Table 2.

A stage is *dominated* if either

(i) uninformative elements make up at least half its vocabulary, or
(ii) its capacity is under half its maximum.

Clause (ii) has one fixed cut, half of the maximum, and nothing else to tune. We wrote the rule, the base-rate screen and a prediction into the analysis script before the 84-system run. A pilot on 22 systems, run through every stage, had already shown `capability_overhang` firing for three quarters of them. The prediction was that Stages 1 and 2 would be dominated and Stage 3 not. Moving the screen to entropy afterwards changes no verdict. Stage 1 was not dominated, and Section 4.1 says what that shows.

3.3 Corpus

The corpus has 298 items in three strata: 106 genuine AI systems, 102 non-AI systems and 90 borderline cases. They come from four sources. Repository READMEs (170) were retrieved programmatically from GitHub topic searches with a star floor, such as `topic:filesystem stars:>500`. Entries from the United States federal AI use-case inventories (95) and court opinions naming a system (13) were retrieved the same way from their published lists. The other 20, such as model cards and reports by non-governmental organisations, were found by hand to fill thin strata, and a hand-found item carries the stratum it was sought for. A README, model card or report item is a verbatim span of about 100 words, cut at whole sentences from the opening of its source; an inventory item is one whole field of the agency's record, and a court item is the passage that describes the system.

For the items found by search, labels were assigned by a rule, not by reading the text. For a repository the rule reads the dependency manifest: "`requirements.txt` declares `torch`" makes an item genuine AI, "no machine-learning dependency in `go.mod`" makes it non-AI. Inventory entries take the agency's own classification, and court cases take how the record characterises the system. The analyser never sees a manifest; it sees only the excerpt.

3.4 What we do not measure

The measure says whether a stage *distinguishes*, not whether it is *right*. A stage can separate systems cleanly and be confidently wrong, and the limiting case is a stage whose every element is a fair coin flip: it reaches maximal capacity and distinguishes nothing. Capacity is a necessary condition, not a sufficient one.

4 Results

Setup. We ran the analyser with three models: `gemini-2.5-flash`, `gemini-3.7-flash` and `gpt-5.6-luna`. Decoding is at temperature 0 with a fixed seed where the model accepts one, and each item was analysed once. Ten calls on one item at these settings returned between five and nine properties, and that variation between draws is not in the intervals below, which resample items. The experiment calls each stage directly, with

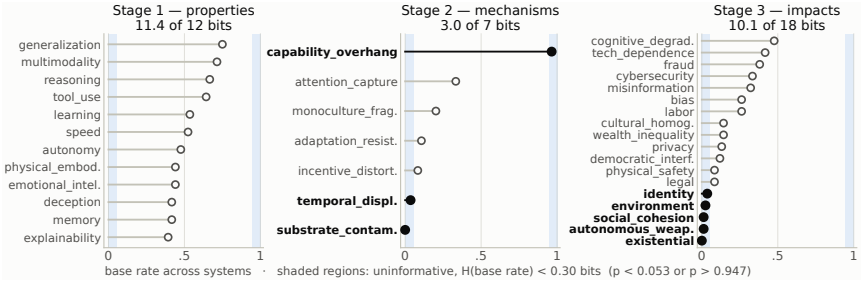

Figure 1: How often each element fires, over the 84 in-scope AI systems. Elements in the shaded regions fire for more than 94.7% or fewer than 5.3% of systems and carry under 0.30 bits each.

Table 1: Checking the screen against a result established with labels. *in-scope*: fraction of the 84 in-scope systems (Section 4) for which Stage 1 asserts the property (`reasoning` is 56 of 84, 0.667 unrounded, under the 0.67 cut); *non-AI*: the same over the 102 non-AI items. *lift* is in-scope minus non-AI, on unrounded rates.

property	in-scope	non-AI	lift
`generalization`	0.75	0.51	+0.24
`multimodality`	0.71	0.42	+0.29
`reasoning`	0.67	0.45	+0.22
`tool_use`	0.64	0.65	0.00
`learning`	0.54	0.40	+0.13

no validator run between stages, so no output was sent back for correction. Stage 1 is measured on all 298 items with all three models. Stages 2 to 5 are measured with `gemini-2.5-flash` only, on the 84 of the 106 genuine AI systems for which Stage 1 asserted at least two properties, which we call the in-scope systems; 10 had none asserted and 12 only one. The study used some 17,000 metered model calls, $41.53 in total.

4.1 Checking the screen against a labelled result

Stage 1 is the one stage with labels, so it is the one place the screen can be checked against a known answer. An earlier labelled analysis had found that `tool_use` does not separate AI from non-AI items on any of the three models. Table 1 gives the five properties Stage 1 asserts most often for the 84 in-scope systems, their rate over the 102 non-AI items, and the difference. `generalization` means the system performs beyond the data it was built on, `multimodality` that it accepts more than one kind of input, `reasoning` that it carries out multi-step inference, `tool_use` that it calls external tools or services, and `learning` that it updates from new data. The analyser decides each from the excerpt alone. Under the same two-property cut on the non-AI side the positive differences shrink to 0.11 or less, so part of the lift is selection.

The screen does not reproduce the labelled finding. It ranks `tool_use` fourth by base rate, at 0.64, below any cut we use, and the two elements the base-rate rule flagged are the two with the largest positive lift. A base rate alone cannot tell a common-and-informative element from a common-and-empty one: `generalization` is common among AI systems and rarer outside them, `tool_use` is common in both, and only the labels separate the two cases. The screen finds the elements worth inspecting; labels, where they exist, say which of them are empty.

Table 2: Capacity in bits by stage, 84 in-scope systems, as defined in Section 3.2. *uninf.*: elements with $H(p) < 0.30$ bits. *p-rule*: elements the base-rate rule fixed before the run would flag ($p \geq 0.67$ or $p \leq 0.05$). (ii) names the clause that fires.

stage	bits	uninf.	p-rule	verdict
1 properties	11.4/12	0	2	not dominated
levels 2–5 only	6.5/12	5	5	not dominated
2 mechanisms	**3.0/7**	3	3	**dominated (ii)**
3 impacts	10.1/18	5	5	not dominated

This is also why our prediction for Stage 1 was wrong. We expected it to be dominated because of the `tool_use` finding, and it is not. Telling AI from non-AI is a job the property taxonomy was never designed for and, with no scope step, was pressed into (Section 4.4); telling one AI system from another is the job it was built for.

4.2 Capacity by stage

Table 2 and Figure 1 give the result. `capability_overhang` fires for 81 of 84 systems and carries 0.22 of a possible bit. Two mechanisms fire for three systems or none. The stage carries 3.0 of 7 bits and is dominated under clause (ii). Of the 4.0 bits missing, 0.78 are lost to `capability_overhang`, 1.78 to the two mechanisms that almost never fire, and 1.44 to the remaining four; recovering either of the first two losses alone would flip the verdict. Put plainly, the 84 systems receive only 17 distinct sets of mechanisms, and 37 of them receive `capability_overhang` alone. Stage 1 is not dominated: 11.4 of 12 bits, and no element is uninformative under the entropy screen. Counted on levels 2 to 5 only, the assertions that say a property is present, Stage 1 carries 6.5 of 12 bits with five elements uninformative. It is still not dominated, but two thirds of what Stage 1 commits to the later stages is level 1, an assertion that a property is absent or minimal, which those stages cannot tell apart from a level 5. Stage 3 is not dominated either, at 10.1 of 18 bits, but five of its 18 axes never fire, so its shortfall sits at the opposite end from Stage 2's. That figure counts an impact at any severity. Of the 272 impacts asserted across the 84 systems, 237 are rated low and only 7 are backed by a quote from the excerpt; counted on moderate and above, the stage would carry 1.5 bits.

4.3 Uncertainty in the capacity estimates

A low capacity is not what noise produces, since elements varying at random push it towards the maximum; the plug-in estimate is biased downward by 0.06 bits for Stage 2, which runs against our

verdict, not for it. Bootstrapping the 84 systems with percentile intervals gives 95% intervals of [11.06, 11.53], [2.41, 3.45] and [9.07, 10.74] bits. Stage 2's interval lies below its half-maximum of 3.5 under the percentile method, and reaches [2.54, 3.60] under the bias-corrected method, so clause (ii) is secured by the point estimate, not by the interval, and the element carries the rest of the case: `capability_overhang` sits at 0.96, Wilson 95% [0.90, 0.99]. Over 615 combinations of the two base-rate cuts, 86.7% reproduce all three verdicts and Stage 2's never changes.

4.4 An implicit scope threshold and an explicit scope gate

The analyser had no step that decides whether a submission is in scope: a Stage 1 that asserted no property left nothing to analyse, and one that asserted any property sent the submission through every stage. Whether a system counted as an AI system was therefore a side effect of the citation rule, not a decision anyone made. `afero` was analysed because Stage 1 asserted two properties for it.

We compared two ways of making the decision on the 208 labelled items. The direct question puts the excerpt to the model in one call, with no taxonomy and no property list, and asks whether it describes an AI system on a five-point scale from "definitely not" to "definitely"; the three models, in the order named above, score 0.939, 0.954 and 0.968 AUC, the area under the receiver operating characteristic (ROC) curve. The implicit rule as it stood, any property at any level, scores 0.510, 0.749 and 0.609. The strongest rule we could build from the property count, which counts only properties at levels 2 to 5 and only those selected on held-out folds for their lift, scores 0.765, 0.797 and 0.726, and the paired comparison below is against that stronger rule. The paired bootstrap differences are +0.174 [+0.123, +0.229], +0.157 [+0.107, +0.211] and +0.241 [+0.185, +0.298]. So we added an explicit scope stage before Stage 1. On `gemini-2.5-flash` it cuts the non-AI items analysed from 81 of 102 to 5, at a cost of three more refusals of genuine systems, 13 of 106; the other two models improve on both counts.

5 Discussion

Being true and being useful are different, and only the first gets checked. Every element measured here is defensible: `capability_overhang` really is something deployed capabilities do. An element enters a taxonomy for being true and clearly defined. A third question would have caught the fault: across the systems the element will be applied to, will it ever fire for some and not for others? Answering that costs one pass over a corpus and no labels.

Being consistent is not being right. The six validators check something real: no identifier and no quote is invented. Each compares one stage with the stage above it; none compares any stage with the system that was submitted. The missing check compares a stage's output against the system it describes; the measure here cannot say whether a stage is right, but it can say when a stage could not be.

6 Limitations

The base rates belong to this corpus, mostly README openings taken as they were found; another corpus would move them. The other two models return an empty Stage 1 profile for 124 and 113 of the 298 items, because a README opening says what a project *is* while the taxonomy asks what a system *does*; on five purpose-written descriptions those two models fill in four to five times as many properties, and with `gemini-2.5-flash` the two mechanisms that almost never fire on the corpus each fired once there, in a single run. Stages 2 to 5 are therefore measured on the one model that asserts often enough, and a model that asserts more pushes the common elements towards $p = 1$, so the choice runs towards the dominated verdict, not against it. The 84 systems those stages are measured on are the genuine AI items for which Stage 1 asserted at least two properties, a stricter form of the implicit scope rule Section 4.4 replaces; Stage 1's own base rates are therefore measured on a sample selected by Stage 1, which raises them. A later audit of the excerpts flags 22 of the 84 as changelogs, install notes or link lists rather than descriptions of a system; on the other 62, Stage 2 carries 2.9 of 7 bits and `capability_overhang` fires for 60. Because most labels come from a dependency manifest, any method that reads library names in the excerpt, the direct question included, is partly reading the label. A regular expression over AI vocabulary scores 0.89 on the same 208 items, so the direct question's margin over reading vocabulary is small, although on the 160 of the 208 whose excerpt names no machine-learning library it still scores 0.91–0.95. What we claim is its margin over the property rule in either form. Finally, all three models disagree with the label on 10 of the 208 non-borderline items, 7 of them federal inventory entries; they stay in every number.

7 Conclusion

A taxonomy filled in by a language model can be accurate, internally consistent and still uninformative, and no check internal to the pipeline can tell. Element base rates detect the near-constant elements without labels. In the analyser studied here the mechanism stage carries 3.0 of a possible 7 bits because one mechanism is asserted for 81 of 84 systems and two for almost none, and the scope decision that let a filesystem library in was never made at all. The measure can show that a stage cannot distinguish systems, never that it can. What remains open is checking whether the effects a pipeline predicts are right, the same measurement on other frameworks (OECD [5], the EU AI Act [2], the NIST AI Risk Management Framework [8]), and the direct test: repair `capability_overhang` and measure again.

References

[1] Linda Crocker and James Algina. 1986. *Introduction to Classical and Modern Test Theory*. Holt, Rinehart and Winston, New York.

[2] European Parliament and Council. 2024. Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act), article 3(1). OJ L, 2024/1689, 12.7.2024, http://data.europa.eu/eli/reg/2024/1689/oj. (2024).

[3] Saibo Geng, Hudson Cooper, Michał Moskal, et al. 2025. JSONSchemaBench: a rigorous benchmark of structured outputs for language models. (2025). arXiv: 2501.10868. DOI: 10.48550/arXiv.2501.10868.

[4] Ziwei Ji et al. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55, 12, 1–38. DOI: 10.1145/3571730.

[5] OECD. 2024. Explanatory Memorandum on the Updated OECD Definition of an AI System. OECD Artificial Intelligence Papers 8. OECD Publishing, Paris. DOI: 10.1787/623da898-en.

[6] Claude E. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27, 3, 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[7] Peter Slattery et al. 2024. The AI risk repository: a comprehensive meta-review, database, and taxonomy of risks from artificial intelligence. (2024). arXiv: 2408.12622v1. DOI: 10.48550/arXiv.2408.12622.

[8] Elham Tabassi. 2023. Artificial Intelligence Risk Management Framework (AI RMF 1.0). Tech. rep. NIST AI 100-1. National Institute of Standards and Technology. DOI: 10.6028/NIST.AI.100-1.