Checking the claims in an AI report cannot find its omissions

28 Sep 2026

For evaluators and applied social researchers who use AI in qualitative analysis, or use or assess work that does.

Summary

- A frontier AI can turn a terms of reference and a pile of transcripts into a convincing evaluation report. A second AI checking it will usually find few errors.
- The checker works from the report back to the transcripts. That measures **precision** (is what the report says right?). It does not measure **recall** (did the report include everything that belongs in it?).
- Reports that commit to little do best: "some farmers" passes, while "only two of 24" can fail.
- Unless someone constrains it, the AI tends to report the findings that are easy to verify. Those are not always the ones the commissioner needed.
- To measure **recall** you work the other way round, from each transcript to the findings. For that you need explicit coding of *all* the relevant material, in a format that shows for every finding who is in it, who goes against it and who said nothing. A checker can then test a random sample of sources.
- Causal mapping codes exhaustively in this way, for causal claims. We code every claim in every passage and answer each question by querying the result, so recall can be checked a chunk at a time. And you only have to do one lot of coding, which can then help answer many of your evaluation questions.
- We still need the professional evaluator. Somebody has to choose the method with its users in mind and know what it did well enough to check it. Somebody also has to vouch for it to those users.

The freestyle evaluation promise

Anyone who wants accurate information about social programmes will by now have met what we call the **freestyle evaluation promise**. In the second half of 2026 you can give a frontier AI model (Claude Opus, GPT-6 Sol, Gemini 3.1 Pro) the evaluation questions from a terms of reference, together with the interviews and reports you have collected. You answer a few clarifying questions, perhaps discuss a coding strategy with it, and it returns what looks like an excellent evaluation report, detailed and often very accurate.

We call this the **soft version** of the freestyle route. You cannot really tell what method the model used, or whether it did anything that might be called coding. If you ask, it may well make up an account of what it did. The **hard version** uses an agent with tools, such as Claude Code, which can run Python in a sandbox and start sub-agents. With an agent you can agree a coding plan first; it then does the coding, consolidates the results and writes the report from them. The soft version only writes something that resembles the outcome of a coding task. We would use it only for very simple tasks. The hard version can produce very convincing results, which you can defend up to a point.

The freestyle checking illusion

In either case you can **hand the report to another AI**, or to the same one, to check its accuracy. Checking a report is much easier than producing one. Reports from both versions usually score well, with few important errors.

Is the AI gorilla going to destroy professional evaluation?

> Problem solved for a desk officer tasked with commissioning evaluation. They can just put the material and the request into a freestyle chatbot and never engage a professional evaluator ever again.

The models write these reports to be accurate, so the reports pass accuracy checks. The bias in this may be what slows the gorilla down. Unless the evaluation question has been specified tightly and the way of answering it agreed in detail (which the hard version partly allows and the soft version does not), the AI picks claims that are probably true and hard to falsify. It steers away from questions and sub-questions whose answers are more equivocal or harder to check, towards a workflow that yields verifiable answers. So a report can score very well without quite answering the questions the commissioner needed answered, in the most valid way. The AI ends up favouring one kind of evaluation workflow, often quite different from the one you would build if you took seriously the task of operationalising the evaluation questions accurately.

The problem: checking an AI evaluation report with a lazy checker

An evaluator, Jo, has 24 interview transcripts. She asks an AI to answer an evaluation question, freestyle. The report she gets back includes sentences such as

> Some farmers credit the training for their bigger harvest (see farmers F3 and F7)

To check it, Jo gives the report and the transcripts to a second AI, or to a colleague, and asks for every sentence to be verified. The checker does as asked. It takes each sentence, finds F3 and F7 in the transcripts and confirms that they credit the training. It might spot a wrong name or a misquotation. Otherwise it passes the report as "zero errors" and Jo congratulates herself.

It does not usually read every transcript looking for farmers the report left out. So it passes the sentence, even if F9 and F15 credit the training too. A report full of weak claims like this can score 100% on the accuracy test.

Jo could have asked for stronger findings, like this:

> Two farmers credit the training for their bigger harvest (farmers F3 and F7)

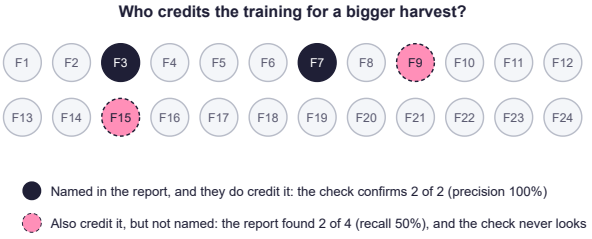
or even like this, making it explicit that the other 22 farmers did not make such a claim:

> Only two out of 24 farmers credit the training for their bigger harvest (farmers F3 and F7)

These claims are stronger because it is clearer how to refute them. Refuting them is more work, though: you have to plough through all the other transcripts to make sure. People are lazy. So are AIs.

The same goes for "Nobody in the northern district mentions drainage as a factor affecting harvests". That is a strong claim, and more work to check.

In the terms used to evaluate search engines and classifiers, the easy check measures **precision**: of the things the report says, how many are right. It does not measure **recall**: of the things in the transcripts that belong in the report, how many the report includes.



What to do about it

Jo's checker gives the weak report full marks. Suppose she asks her AI for strong claims instead, and gets

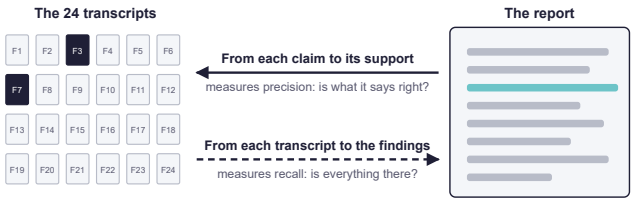
> Only two out of 24 farmers credit the training for their bigger harvest (farmers F3 and F7)

Now the same checker can find an error. If it happens to read F9's transcript it finds a third farmer, and the report loses a mark. **The report that says more scores worse.** If we check for precision alone, bland reports that commit to little come out best.

Without serious extra machinery, an AI checker (or a lazy human) is still much worse at finding omissions than at checking what has been reported. The gap widens with larger corpora (more and longer documents), because thorough checking is expensive. Our usual tricks for large corpora help less here; RAG (retrieval-augmented generation, which fetches only the passages that look relevant) is one.

In our experience, freestyle AI checkers also treat a wrongly named farmer as a serious error and a missing one as a minor slip. When they do find an undercount, they mark it down less.

A human checker has the same problem. Jo's colleague would read the report and look up each claim, exactly as the AI did. **Whoever does it, a check that starts from the report and looks for support in the transcripts measures precision well and recall poorly.**



A claim-by-claim check only opens the transcripts the report names (F3 and F7).

Making recall measurable

To measure recall you work the other way round: start from each transcript and ask which findings that farmer belongs in. Does F9 credit the training? Does F15? Doing that for every farmer and every finding needs plain old-fashioned explicit coding. It is hard to get a consumer chatbot, or even a tool like Claude Code, to do this well. You can ask it to give every finding a small report in a standard format, something like this:

> Credits the training for a bigger harvest: F3, F7, F9 and F15 say so. F12's account goes against it: the harvest fell despite the training. F20 says it only of the neighbours. The other 18 do not mention it.

Every count in the report then has to come from a format like this, so "only" and "nobody" become statements about the format, which anyone can check. Lazy checking gets a bit harder to excuse. Whoever fills in the format can still miss F9. But a checker can now find the misses without ploughing through every transcript. The checker picks four transcripts at random, reads each one in full, places each of those farmers on every finding without looking at the format, and compares. If the sample shows one placement in ten missed, expect roughly the same across all 24. Auditors do the same with a sample of invoices.

To go further, in analysis and in checking, you have to stop just dumping your source texts into a chatbot and playing with the prompt. You might move to a more systematic coding tool. Then of course you lose the amazing flexibility of the frontier models, which can dream up a freestyle method and produce a perfect-looking report with good precision (and dodgy recall).

Sparse and dense tasks

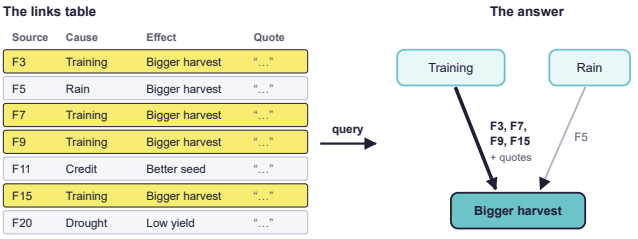
Text coding tasks can be sparse or dense. In a dense task there are several possible hits in a small amount of text. In a sparse one you may search through pages to find a single hit. Dense tasks can be expensive, but the codebook can be easier to write. Sparse tasks seem cheaper, perhaps only because you are skipping loads of text that might contain a hit.



Why causal mapping is a good fit

Causal mapping turns a sparse question into a dense task. Jo's question, who credits the training for a bigger harvest, is sparse. Most passages in the 24 transcripts do not mention it. A reader looking for the few that do starts to skim. In causal coding we code every causal claim in every passage, whatever it is about, as a link from a cause to an effect, with the verbatim quote and the source it came from (minimalist coding). In interviews about change that is a dense task, because nearly every paragraph has something to code. The codebook is short, too: code every claim that one thing influenced another. We also code the text only once, into one model of all the causal narratives, and answer every question from that model.

We then get each finding by querying the links. "Who credits the training for a bigger harvest?" becomes "which sources mention a chain from `Training` to `Bigger harvest`?" The answer lists every such source with its quotes. That gives us the standard format from the previous section without having to ask the model for it. A farmer whose harvest fell despite the training appears as a contrary link (despite-claims). "The other 18 do not mention it" is a statement about the links table: we do not code absences, so it means only that those farmers did not make the claim. The model never chose which findings to report, so it could not drift towards the easy ones. The analyst chooses the questions. The AI does one narrow extraction job.



Every farmer who made the claim is on the arrow. The others did not make it, which is all their absence says.

We can then measure recall a chunk at a time. The AI reads one short chunk of text per request, never the whole corpus, so a person can recode any piece of its work by hand and compare. Precision and recall are what we check at the coding step, on a small, varied sample, before we code the whole corpus. In our coding experiments, a model given a whole 89,000-character document in one request returned 6 links, where chunked runs returned 114. Given a large context, the model in effect codes only a sample of it. We also saw a small-scale version of the bias described at the top of this page. The most expensive model we tested made few errors but missed over half the real content, because it avoided faint or debatable claims. Its selection was no better. We raised recall with an accounting contract: for every numbered segment of a chunk, the model had to return either its claims or an argued "none". So the coding itself works from the transcripts towards the findings.

Causal mapping deals only with claims about what causes what. That is a big part of many terms of reference, though not all of it. For the other questions we have to build the same discipline question by question.

So is the gorilla going to win?

No. Even the best-built workflow needs somebody to choose it, know what it did **and vouch for it**. A commissioner who receives a report needs to know which method was actually used, what was read and what was left out, and whether the method fits the question they needed answered. A chatbot can suggest a method and describe it. A person has to declare that this is the right kind of method for this question and put their name to it. As the Rubicon principles have it, worth is declared by a person.

Part of that job is technical. Evidence is used by people: a programme team, a board, a funder, a community. Each of them needs different evidence and trusts different sources. An evaluator learns who needs what, and why they would believe it, through relationships and through knowing the setting, over time. The users of the evaluation can hold that evaluator to account. As machines do more of the analysis, what evaluators are paid for is increasingly the rest: choosing and operationalising the questions, vouching for the method and taking responsibility for the conclusions.

Punchline

A freestyle AI report can get full marks on an accuracy check and still not answer the question. If you check it by reading the report, you learn how much of what it says is right but nothing about what it left out. Under that check a vague report looks accurate and a committed one looks careless. To measure recall you work from the transcripts towards the findings. For that you need explicit coding of the whole corpus behind the findings, in a format that shows for each count who is in it, who goes against it and who said nothing; a checker can then sample the sources and compare. Causal mapping does this for causal claims. We code every claim in every passage, answer questions by querying the links, and check recall chunk by chunk. Rubicon is our attempt to do the same for evaluation questions of any kind: each question is broken into steps somebody else could check, and every count is stated out of what was read. Whatever the tool, an evaluator still has to choose the method, know what it did and vouch for it to the people who will use the findings.

What the literature calls it

We did not find a single established name for this. Several literatures describe parts of it.

Precision without recall. FActScore, which scores a long AI answer claim by claim, "considers precision but not recall, e.g., a model that abstains from answering too often or generates text with fewer facts may have a higher FActScore" (Min et al. 2023, section 3.1). Wei et al.'s F1@K adds recall, measured against a preferred number of supported facts (2024), so it rewards quantity, which is a different thing from completeness against the source. In summarisation research omission is an error type of its own; Zou et al. built a dataset of omission labels because none was available (2023).

Omission blindness in AI judges. Zheng et al.'s study of models used as judges examined position, verbosity and self-enhancement bias (2023); omission is not among them. A recent preprint names it. Fox et al. call it omission blindness: eight judge designs reading clinical notes against the transcript told a flawed note from its correct twin at 0.79 to 0.94 for added or altered content and 0.50 to 0.63 for omissions, where 0.5 is a coin flip (2026). Detection partly recovered when the task was restructured to start from the source: "list the facts the transcript establishes, then check the note for each". AbsenceBench shows the same weakness in language models generally (Fu et al. 2025). A second model may also share the first model's blind spots: Goel et al. found mistakes growing more similar as models become more capable, "pointing to risks from correlated failures" (2025).

Auditing: the direction of the test. Auditors separate the claim that recorded items are real (the existence or occurrence assertion) from the claim that everything that should be recorded is (the completeness assertion) (PCAOB AS 1105, para .11). They are tested in opposite directions: for completeness, "the procedure should start from the underlying documents and check to the entries in the relevant ledger to ensure none have been missed" (ACCA). Checking from the record back to the evidence is often called vouching; checking from the evidence forward to the record is tracing (Vero 2025). In those terms, a claim-by-claim review is vouching. Where full tracing costs too much, auditors sample and "project the misstatement results of the sample to the items from which the sample was selected" (PCAOB AS 2315, para .26). The analogy has a limit. An invoice is either in the ledger or not, whereas deciding that someone "said X" is itself an act of coding.

Why checkers let undercounts go. Klayman and Ha's positive test strategy is the tendency to test cases expected to have the property in question (1987). When the hypothesised set is a subset of the true set, as when a report names F3 and F7 and F9 also qualifies, people using only positive tests "can never discover that their rule is incorrect" (p. 214). A tester "may simply be more concerned that all chosen cases are true than that all true cases are chosen" (p. 216). Their subjects were testing their own hypotheses, which a checker is not. A more basic finding is the feature-positive effect: adults learn a discrimination more easily when the presence of a feature marks the answer than when its absence does, which the authors take as evidence of "the difficulty organisms experience in using 'nonoccurrence' as a cue" (Newman, Wolff and Hearst 1980). Two terms fit less well. Satisfaction of search, missing a second abnormality once one is found (Adamo et al. 2021), assumes a search for further cases that a checker is never asked to make. Omission bias, judging harmful omissions less bad than harmful commissions (Spranca, Minsk and Baron 1991), concerns people's actions.

Qualitative method and information retrieval. Negative case analysis, one of Lincoln and Guba's techniques for credibility, is "the constant reworking of hypotheses in light of disconfirming evidence" (Phillips et al. 2014, tabulating Lincoln and Guba): the analyst's discipline of reading beyond the claim. In information retrieval the counting problem is explicit. Missed items are absent from the results, so recall is estimated "by assessing a random sample of documents", including documents the search did not return (Webber 2012).

Sources

- Adamo, S. H., Gereke, B. J., Shomstein, S. and Schmidt, J. (2021). From "satisfaction of search" to "subsequent search misses": a review of multiple-target search errors across radiology and cognitive science. *Cognitive Research: Principles and Implications*, 6.
- ACCA. The audit of assertions. Technical article for ACCA students.
- Fox, S., Markham, L., Lail, R. and Karotsieris, M. (2026). LLM judges verify presence, not absence: omission blindness in AI clinical notes and what recovers it. arXiv preprint 2608.31016.
- Fu, H. Y., Shrivastava, A., Moore, J., West, P., Tan, C. and Holtzman, A. (2025). AbsenceBench: language models can't tell what's missing. arXiv 2506.11440.
- Goel, S. et al. (2025). Great models think alike and this undermines AI oversight. arXiv 2502.04313.
- Klayman, J. and Ha, Y.-W. (1987). Confirmation, disconfirmation, and information in hypothesis testing. *Psychological Review*, 94(2), 211–228.
- Min, S. et al. (2023). FActScore: fine-grained atomic evaluation of factual precision in long form text generation. EMNLP 2023.
- Newman, J., Wolff, W. T. and Hearst, E. (1980). The feature-positive effect in adult human subjects. *Journal of Experimental Psychology: Human Learning and Memory*, 6(5), 630–650.
- PCAOB. AS 1105: Audit Evidence.
- PCAOB. AS 2315: Audit Sampling.
- Phillips, C. B., Dwan, K., Hepworth, J., Pearce, C. and Hall, S. (2014). Using qualitative mixed methods to study small health care organizations while maximising trustworthiness and authenticity. *BMC Health Services Research*, 14.
- Spranca, M., Minsk, E. and Baron, J. (1991). Omission and commission in judgment and choice. *Journal of Experimental Social Psychology*, 27, 76–105.
- Vero, Y. (2025). Audit tracing and vouching: a complete guide. Trullion blog.
- Webber, W. (2012). Approximate recall confidence intervals. arXiv 1202.2880; later in *ACM Transactions on Information Systems*.
- Wei, J. et al. (2024). Long-form factuality in large language models. arXiv 2403.18802.
- Zheng, L. et al. (2023). Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. NeurIPS 2023 Datasets and Benchmarks Track.
- Zou, Y., Song, K., Tan, X., Fu, Z., Zhang, Q., Li, D. and Gui, T. (2023). Towards understanding omission in dialogue summarization. *Proceedings of ACL 2023*, 14268–14286.

This page was drafted with the assistance of Claude, an AI model from Anthropic; the arguments and judgements are our own.