CAN LLMS ANTICIPATE THEIR CRUXES?

Anonymous authors
Paper under double-blind review

ABSTRACT

To use language models to help us make decisions we would like to understand the rationale behind the language model's outputs. In principle, the language model provides its rationale as part of its text output. However, the sheer volume of text makes this hard to use in practice. In this paper, we are interested in extracting the *cruxes* that lead a model to recommend one action over another. We take a crux to be a sub-question where changing its value may have altered the model's conclusion. We consider three strategies of increasing computational cost. First, we just ask a model to identify, a priori, what the cruxes of a decision will be. Second, we have the model actually go through the process of generating a response and then extract the cruxes that arose in its reasoning. Finally, we generate many responses and extract the cruxes by comparing the reasoning across the rollouts. We introduce a number of tricks to make this last procedure computationally tractable. We then conduct an extensive empirical evaluation to assess whether the computationally inexpensive options are sufficient. They are not. Across domains and models, we find that a model's a priori assessment of the cruxes has little relation to the factors that are actually identified as important a posteriori. Using a single rollout helps, but is still inadequate for identifying the full set of cruxes. Accordingly, for high-stakes decisions (and to understand the space of language model decision making) the many rollout procedure is unavoidable.

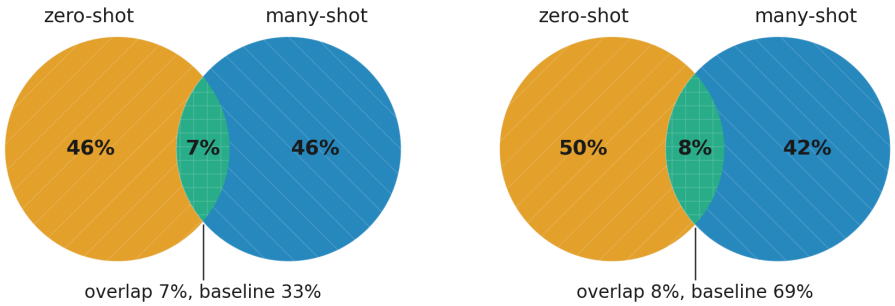
1 INTRODUCTION

Consider using a language model to aid in decision making. For example, someone asking a chatbot whether to end a relationship, a patient weighing an AI-assisted diagnosis, or a researcher considering an agent-implemented change to their codebase. To decide what the right decision is, we would like to reduce the overall argument to the critical subquestions that lead to the conclusion. These subquestions are the *cruxes* of the argument (Sabien, 2017). They should surface the assumptions and reasoning, implicit or explicit, that influence the conclusion. For example, an agent that skips the inputs that crash its pipeline has quietly decided that they do not count. The question we consider here is how to actually get the cruxes of a decision.

The most obvious option is to simply ask the language model what the cruxes are. We can do this a priori—just ask in advance what the relevant subquestions will be—or a posteriori—have the language model generate a decision, and then look at its trace to determine what the key subquestions were. The a priori case is interesting because it can influence the information we want to gather to inform the decision (e.g., what questions the language model might ask a patient). The a posteriori case would hopefully be useful to sanity check the model's reasoning. In both cases, the method is appealing for both simplicity and inexpensiveness. Then, the essential question is whether this is adequate. That is, can just asking the model for the cruxes reliably elicit them?

To study this question, we introduce a method for extracting cruxes by contrasting many rollouts aimed at the same decision. Intuitively, the idea is that if a question is a crux of the reasoning then we should actually see variation across independent reasoning of the decision! To make this concrete, consider the forecasting question of whether Frontier Airlines files for bankruptcy. A priori, the bot identifies "whether Frontier keeps enough cash to avoid a distress filing" as an important factor. Given a single rollout, the bot might actually end up focusing on "whether Frontier has enough cash to meet its obligations", adding *another* crux. Given multiple rollouts, we might observe that the bot always has the same belief about Frontier's cash position—so it's not actually a crux of the reasoning!—but also shows high variation in how much credence it gives to a rival airline's recent bankruptcy, which

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



A priori: "Does free-throw performance become a decisive factor?"
A posteriori, from many rollouts: "Iowa State forced 23 turnovers against Arizona State on March 7."

A priori: "Will Frontier retain enough liquidity to avoid a distress filing?"
A posteriori, from many rollouts: "Altman Z-score ≈ 0.38: distress zone."

Figure 1: In a basketball game and a bankruptcy question, the cruxes a model predicted matched the questions its forecasts varied on far less often than even shuffled labels would. **Left:** Iowa State vs. Kentucky (March Madness). **Right:** Frontier Group bankruptcy (Metaculus). Each diagram splits the clusters of paraphrased cruxes (Section 5), formed from 25 cruxes per set, into those holding a priori cruxes only (orange), both kinds (green, the overlap) and a posteriori cruxes only (blue). Below each diagram stand one a priori crux, as a question, and one a posteriori crux, as the assumption one forecast made (**contribution 1** extracts the a posteriori cruxes; **contribution 2** groups similar paraphrases at low cost). A turnover is a possession lost before the team can shoot, and a forced turnover is one the defense causes. The Altman Z-score is a bankruptcy-risk score computed from a company's accounts. Below the lens each diagram prints its overlap at 25 cruxes and its permutation baseline (**contribution 3** runs this comparison across games, events and models). These are 7% against 33% for the game and 8% against 69% for the event.

then *is* a crux (Figure 1, right). The question is whether we really need the multiple rollouts, or whether LLMs are already capable of identifying (a priori or a posteriori) the cruxes of their own reasoning. To address this:

1. We propose an elegant (but naively, expensive!) method for extracting the cruxes of a model's conclusion from multiple rollouts. Informally, the idea is to contrast each pair of rollouts looking for important distinctions in the reasoning. Equivalent paraphrases of the same crux are then clustered.
2. We make the method affordable. For n rollouts yielding k cruxes per pair, merging their paraphrases would naively take $\Theta(k^2n^4)$ calls to a generative judge; the method needs $\Theta(n^2)$ language-model generations and $\Theta(k^2n^4)$ reranker scores instead, and since a reranker score is a single forward pass rather than autoregressive decoding, matching semantically equivalent cruxes becomes far cheaper (Section 5).
3. We use this machinery to conduct a thorough experimental evaluation on forecasting questions from Metaculus, which hosts public forecasting tournaments, and on the outcomes of March Madness, an American college basketball tournament, scoring each cheaper strategy against the many-rollout cruxes and a permutation baseline, at three levels of strictness in judging equivalence and across four model families.

The finding is that simply asking for the cruxes is plainly inadequate; see Figure 1 for some example questions. The models are wholly unable to anticipate the cruxes of their own reasoning. Further, using only a single rollout typically misses many cruxes. Accordingly, for high-stakes decisions, the multiple-rollout approach is necessary.

2 Related Work

Three lines of work come closest to this paper. One asks if a model can report on itself: whether it knows what it knows (Kadavath et al., 2022; Lin et al., 2022; Kapoor et al., 2024), whether it can describe its own tendencies (Binder et al., 2025; Betley et al., 2025; Laine et al., 2024; Lindsey, 2026; Comsa & Shanahan, 2025), and whether its written reasoning is faithful to what produced its

answer (Wei et al., 2022; Turpin et al., 2023; Chen et al., 2025; Chua & Evans, 2025). None of that work asks the question here, whether a model can say in advance which factual questions its own forecasts will disagree on. A second line breaks a text into claims (Bar-Haim et al., 2020; Min et al., 2023; Wei et al., 2024) or uses the spread among many outputs to reach a better answer (Wang et al., 2023; Du et al., 2024). The comparison here instead reads two forecasts at once and keeps only the claims on which they differ. A third builds forecasting systems that search for evidence before they forecast (Halawi et al., 2024; Murphy, 2026), and the Metaculus forecasts here come from a system of that kind. Each line is described in greater detail in Section E.1.

3 Data Sources

A particularly clean task with high importance is forecasting, where a model outputs a probability for a future event with supporting rationale. For our purposes, we restrict to binary events, where the model must choose between two mutually exclusive outcomes; for example, a model might forecast the probability that oil will trade above $40 a barrel in ten years.

Given a binary forecasting problem, we use a forecasting agent which calls a deep research assistant, then considers what the most important factors are among that information, and aggregates the evidence collected along those factors into a probability about the event's outcome.

We consider two data sources: Metaculus for breadth of events, and March Madness for a more fine-grained analysis of a single event type.

3.1 Metaculus

Metaculus is a public forecasting platform that hosts tournaments on a wide range of topics: we consider 15 Metaculus events, spanning topics from geopolitics to economics to science and technology (Section C.1).

We generated 30 rollouts on each event, and 449 of the 450 were stored, so each of the 15 events has at least 29 rollouts. Each rollout received the same prompt but ran its own web searches, six calls that gather background and then three probability estimates from different framings, with no search result shared across rollouts.

To contextualize the results later on, it helps to name the sources of variation among rollouts. For Metaculus, rollouts therefore differ through the model's sampling and through the web search results each received. Its rollouts are written by gpt-5-nano with web search, and the cruxes predicted in advance come from gpt-5.4 and those read off the rollouts from gpt-5.4-mini (Table 10), so its 15 events give breadth of topic with one model family.

3.2 March Madness

March Madness is the college basketball tournament run each year by the National Collegiate Athletic Association in the United States, in which university teams play knockout games until one champion remains. For the sake of this paper, it is enough to know that it is a single-elimination tournament. The experiments involve predicting the outcome of 59 of its games, each between two teams.

The same forecasting agent as above was used to forecast the winner of each game before the tournament began. Since there were 1,891 possible matchups among the 62 teams, the agent's web-search stage ran once per team rather than once per rollout. It wrote three scouting reports on each team, which were cached and reused for every potential matchup. The rollouts on a game therefore read those reports instead of searching the web for the same information. Each report was written from one of three personas to induce different perspectives on the same team. The personas were a 'Traditional Basketball Analyst', a 'Quantitative Forecaster', and a 'Matchup Analyst'.

Hence the rollouts on a game differ only through the model's sampling, at the API's default temperature, and through which pair of scouting reports they read.

Fixing the scouting reports removes the variation that web search adds and isolates the model's own sampling. In the main runs gpt-5.4-nano writes the rollouts, predicts their cruxes in advance and reads the cruxes off them, and we reran that whole chain on three open-weight models, qwen3:32b,

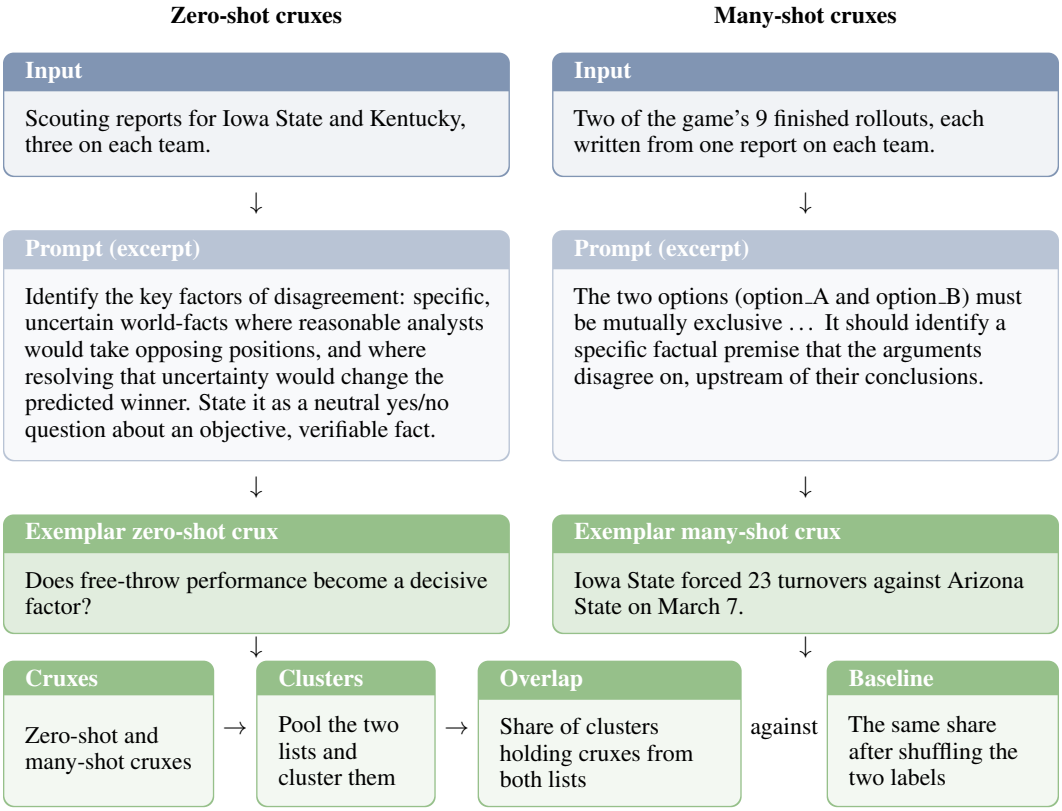


Figure 2: In March Madness the test asks whether a model, reading every scouting report its forecasts were written from, can foresee where those forecasts will disagree. Zero-shot cruxes (left) against many-shot cruxes (right), for one March Madness game, Iowa State vs. Kentucky. The zero-shot prompt predicts cruxes from the reports, and the many-shot prompt finds them in each pair of the game's rollouts. The zero-shot and many-shot cruxes are then pooled and clustered, and their overlap is compared with a baseline that shuffles the two labels among the same cruxes (bottom row). Each crux is a yes/no question with both answers, and the figure quotes the many-shot crux by the assumption one of the two rollouts took, that Iowa State forced 23 turnovers in an earlier game.

gemma4:31b and gpt-oss:120b (Section 7). Each rollout reads one of the three reports on each team, so the 3×3 pairs of reports give 9 rollouts per game.

4 METHODS

The question this paper asks is which method of crux extraction is best, given a fixed forecaster, the model that writes the rollouts. The first asks the model in advance what the cruxes of its decision will be (zero-shot). The second has the model generate one rollout and then extracts the cruxes from that single rollout (one-shot). The third generates many rollouts and extracts the cruxes by comparing pairs of them (many-shot). Figure 2 shows the zero-shot and many-shot prompts for one game. For the experiment, the forecaster is better considered as part of the data, because in a high-stakes setting one would expect one model to generate the rollout and make the recommendation while a different model evaluates its trustworthiness. Table 10 lists the model in each role, forecaster, predictor and extractor, and any model can play any part (Section 7).

Predicting the cruxes in advance (zero-shot). The zero-shot method asks the model, without seeing any rollout, on which cruxes the rollouts will take different sides. In Metaculus the model receives only the question's text, with no background, forecasts or web search results, so the rollouts' web searches may find evidence it lacks (Section A.1 weighs what this can explain), and it is sent 11 prompts, each 5 times, for 55 calls per event (Section B.2). In March Madness the model, the same one that writes the rollouts, receives all three scouting reports on each team, the same reports the

game's rollouts are written from. Five differently worded prompts are sent once each, and every crux they return joins the zero-shot set, with repeats removed (Section 6).

Extracting the cruxes from a single rollout (one-shot). The one-shot method picks one rollout per game or event at random, with a fixed seed. A prompt written for one rollout then extracts the cruxes of that rollout alone, about 12 per game and 6 per event (Section A.5). Its extraction prompt is intentionally kept as close as possible to the many-shot extraction prompt, differing only where one rollout replaces a pair (Table 6). Section D.2 shows one Metaculus rollout in full.

Discovering the cruxes from many rollouts (many-shot). This method of extracting cruxes instead contrasts two rollouts on the same request and asks the extractor for the factual questions on which the two take different sides. Note that this has the advantage of not having to enumerate all possible factors that could be important; by construction it restricts to the factors that are relevant to the model's sampling distribution. Taking rollouts two at a time ties each crux to one pair of rollouts that disagree on it and records which side each of the two took. Whichever method produced it, a crux is stored as a yes/no question together with its two answers, one for each side a forecast could take; examples in this paper often quote a crux by one side's answer, such as "world demand for oil will have peaked in ten years". Figure 2 shows both prompts for one March Madness game.

Every pair of rollouts is contrasted, even two that reach the same conclusion, because they can still take different sides on a crux (Section 7). For example, both rollouts can back the oil bet, with one expecting world demand to keep growing and the other expecting it to peak while supply shrinks faster. Even though n rollouts give $\binom{n}{2}$ pairs to contrast, they produce far fewer distinct cruxes, because the cruxes repeat heavily across pairs: once paraphrases of one crux are grouped, the number of distinct cruxes grows about linearly with the number of rollouts, about 3.4 per rollout in March Madness, from 4 at two rollouts to 28 at nine, while the number of pairs reaches 36 (Figure 3). Because the method starts from disagreements the model actually produced, it never has to enumerate every factor a forecast could turn on. The cruxes it finds also recur across rollouts. Five of nine rollouts give 28% of the pairs yet 61% of the distinct cruxes all nine give, and seven give 83%.

5 Equivalence between Cruxes

Because a natural-language crux can be phrased many different ways, how are we to judge whether two cruxes are equivalent?

Two cruxes are treated as the same when the same evidence would confirm or falsify both.[1] Judging that for every pair of cruxes with a call to a generative judge does not scale: with n rollouts and k cruxes returned per contrasted pair, the candidate cruxes and the judgements a naive all-pairs comparison needs are

$$K = k\binom{n}{2} = \Theta\big(kn^2\big)\,, \qquad \binom{K}{2} = \frac{k^2n^2(n-1)^2}{8} - \frac{k\,n(n-1)}{4} = \Theta\big(k^2n^4\big)\,, \tag{1}$$

which, averaged over games or events with each one's own K, is 12,978 judgements per March Madness game ($n = 9$, mean $k = 4.47$) and 1,013,888 per Metaculus event ($n = 29$ or 30, mean $k = 3.27$) at one level of strictness.

Two choices make the comparison affordable. First, each pair is scored not by a call to a generative judge but by a reranker, Qwen3-Reranker-8B (Zhang et al., 2025). It is a model built to score whether one passage matches another under a written instruction, in one forward pass of about 270 tokens. The method's cost therefore splits into generations and forward-pass scores,

$$\underbrace{n + \binom{n}{2}}_{\text{generations, } \Theta(n^2)} + \underbrace{\binom{K}{2}}_{\text{reranker passes, } \Theta(k^2n^4)}\,, \qquad P = \binom{2m}{2} = 1{,}225 \text{ pairs at } m = 25, \tag{2}$$

[1] A match does not require a zero-shot crux to predict which side any rollout took. One set of cruxes may ask whether world demand for oil will have peaked in ten years and the other whether it will still be growing, so the first answer of one crux is the second answer of the other. Both sets label a crux's two answers A and B, and the instruction behind every reported number tells the reranker, introduced below, to ignore their order (Section A.2).

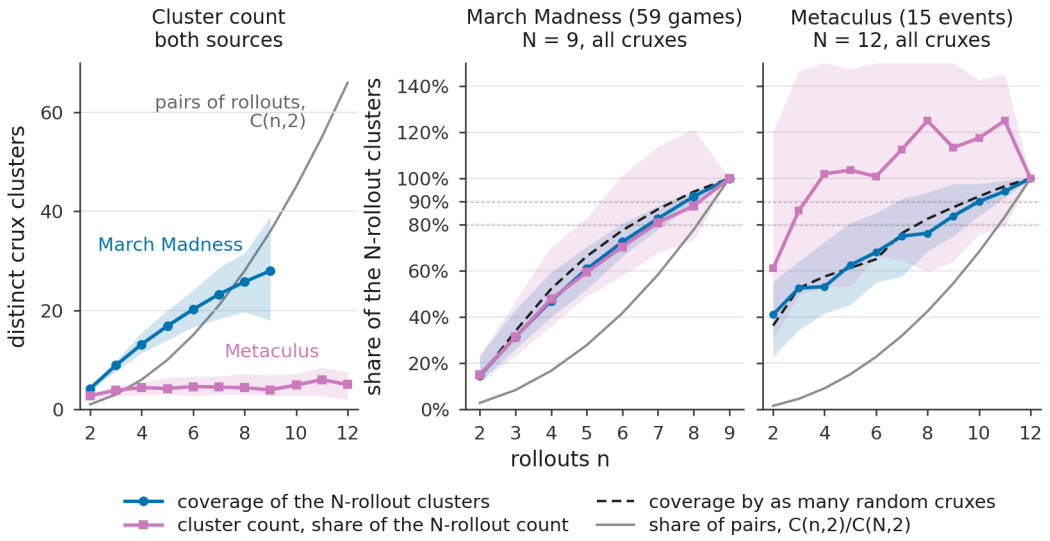


Figure 3: Cruxes recur across rollouts, so the distinct cruxes grow about linearly while the pairs grow quadratically, and nine rollouts still do not exhaust them. **Left:** the median number of distinct crux clusters at the middle level of strictness against the number of rollouts, over 20 random subsets per game or event (bands, 10th to 90th percentile), with the number of rollout pairs for reference. March Madness reaches 28 clusters at nine rollouts, and Metaculus, with 12 rollouts per event and all their cruxes, stays near 5 from four rollouts on. **Middle and right:** for March Madness (59 games, $N = 9$) and Metaculus (15 events, $N = 12$), the share of the clusters found by all N rollouts that n rollouts already find (blue), the number of clusters from n rollouts as a share of the number from all N (pink, cut off at 150% in Metaculus), the same share for as many random cruxes from the pool (dashed), and the share of pairs $\binom{n}{2}/\binom{N}{2}$ (grey). Five of nine rollouts give 28% of the pairs yet find 61% of the clusters all nine find, and the blue curve is still rising at nine.

and every pair of the complete March Madness many-shot cruxes was scored once, for Figure 3. Those 751,515 pairs at the middle level took 12 GPU-hours. The reranker's score can change when two cruxes swap places. The similarity of two cruxes is the mean of the two passes with each crux in turn as the query, $s(i,j) = \frac{1}{2}\big(r(i,j) + r(j,i)\big)$. The reranker and the clustering below were checked on 2,000 March Madness crux pairs judged by gpt-5.4-nano. Its balanced accuracies were 0.845, 0.826 and 0.840 at the strict, middle and broad levels below. It kept 94 to 98 percent of the pairs the judge called different apart, at a small fraction of a judge's cost (Section A.2).

Second, for the overlap comparison each set is capped at $m = 25$ cruxes by a seeded draw. The pooled $2m$ cruxes then give the P pairs of Equation (2) per question and per level. The cap fixes the cost rather than computing the full comparison cheaper. The ratio $\binom{K}{2}/P$ of at least 10.4 in March Madness and 808 in Metaculus is a lower bound on what the cap saves (Section B.2).

The rule leaves open how much two cruxes must share, so the reranker scores each pair at three levels of strictness. Each level asks for less than the one before and is named for what two matching cruxes share. At the strict level, *same question*, two cruxes match when they address the same underlying point of disagreement, however they are worded. At this level, cruxes that share vocabulary but ask different things stay apart. At the middle level, *same resolution*, two cruxes match when resolving one would necessarily resolve the other, even if one is more specific. At the broad level, *same dispute*, they match when settling either would end the same disagreement between two rollouts. Each level gives its own clusters, and Section A.2 works through an oil example that separates the three. The middle level produces every number in the main text unless the text names another level.

With distance $1 - s(i,j)$, average-linkage hierarchical clustering groups the pooled cruxes. It starts from single cruxes and repeatedly merges the two groups whose mean pairwise distance is smallest. Merging continues until one group remains, and the order of the merges forms a tree. Cutting the tree means stopping the merges at a chosen number of clusters. The number is picked by persistence, as the partition that survives the widest range of merge heights. Ties go to fewer clusters, and a single cluster is never chosen. The cut makes no use of which set a crux came from. At 25 cruxes per set

the cut leaves on average 20.5 clusters per March Madness game (range 2 to 36). The mean is 5.2 per Metaculus event (range 2 to 12). Where the cut falls decides how different two cruxes can be and still share a cluster. The comparison is therefore repeated in Section 7 at cuts from broad categories down to one crux per cluster.

A cluster is *shared* when it holds both zero-shot and many-shot cruxes, and otherwise it holds zero-shot cruxes only or many-shot cruxes only. The overlap is the fraction of clusters that are shared. Several pairs of rollouts can disagree on one crux, so the many-shot cruxes may word it several ways, and counting clusters counts it once.

6 Overlap Metric against the Permutation Baseline

Many extracted cruxes are never marked equivalent to any other crux, so they sit alone in their own clusters. A cluster of one can never be shared by the two sets. The raw overlap is therefore set as much by the cluster-size distribution as by agreement between the sets. This is why every overlap is read against a permutation baseline. Let the pooled cruxes fall into J clusters of sizes $c_1, \ldots, c_J$. Call the zero-shot or one-shot cruxes set A and the many-shot cruxes set B, with n_A and n_B cruxes and $N = n_A + n_B$. Let a_j be the number of set A cruxes in cluster j. The overlap is the share of clusters that hold cruxes of both sets,

$$\text{overlap} = \frac{1}{J} \sum_{j=1}^{J} \mathbf{1}\big[\, 0 < a_j < c_j \,\big]. \tag{3}$$

The baseline keeps the clusters fixed and reassigns the label A uniformly at random to n_A of the N pooled cruxes. Cluster j is then a random draw of c_j of the N cruxes, unshared only if the draw takes no set A crux or nothing but set A cruxes, so it is shared with probability

$$p_j = 1 - \frac{\binom{N-n_A}{c_j}}{\binom{N}{c_j}} - \frac{\binom{n_A}{c_j}}{\binom{N}{c_j}}, \qquad p_j = 0 \text{ when } c_j = 1, \tag{4}$$

and the baseline is $\frac{1}{J}\sum_j p_j$, estimated as the mean overlap over 1,000 such shuffles. The *gap* is the baseline minus the observed overlap, in percentage points.[2]

The overlap also moves with the number of cruxes on each side.

A few zero-shot cruxes could miss the many-shot cruxes merely because they are few, so the comparison runs at set sizes up to 25 cruxes. At each size k, each set gives k of its cruxes, or all of them when it holds fewer. The two sets are then pooled and clustered anew, as Section 5 describes. At each size the overlap is averaged over the games of March Madness and, separately, over the events of Metaculus. The overlap and baseline coincide by construction at one crux per set, so results cover sets of 5 to 25 cruxes.

A gap near zero means that, judged by their clusters, the two sets cannot be told apart. A positive gap means the zero-shot cruxes land in the same clusters as the many-shot cruxes less often than shuffled labels would, and a negative gap means more often. Each game or event is paired with its own baseline, and a one-sided Wilcoxon signed-rank test over those pairs gives the p value of each gap. The one-shot cruxes are scored the same way, with one adjustment. They come from a single rollout, and some many-shot cruxes were extracted from pairs that include that rollout, so those many-shot cruxes are set aside before the comparison. The zero-shot cruxes are then rescored against the same reduced set of many-shot cruxes, so the two are compared on equal terms (Section A.5). Two more scores use the same clusters and the same random relabelling. Coverage is the share of the many-shot cruxes whose cluster also holds a zero-shot or one-shot crux. Its mirror image is the share of the zero-shot or one-shot cruxes whose cluster also holds a many-shot crux.

A further check, the split-half check, splits one set, zero-shot or many-shot, into two random halves and scores them against each other. It answers the objection that the many-shot cruxes are too varied for any set of cruxes to match (Section 8.1).

[2] Gaps are computed before rounding, so a gap can differ slightly from the difference of the rounded overlap and baseline.

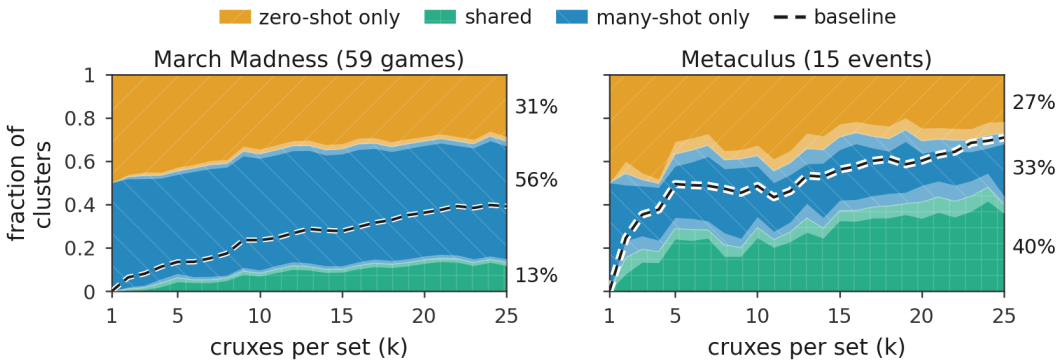


Figure 4: Naming more cruxes, up to 25 on each side, never lifts the match between predicted and discovered cruxes to the permutation baseline. Bands give the share of clusters holding zero-shot cruxes only (top, orange), many-shot only (middle, blue) or both (bottom, green, the overlap). The dashed line is the permutation baseline, the overlap when the two labels are shuffled. **Left:** March Madness, 59 games. **Right:** Metaculus, 15 events, baseline 49.5% at 5 cruxes and 70.9% at 25. Middle level; lighter bands show ± 1 SE, and the numbers at the right give each band's share at 25 cruxes.

7 Robustness Checks

Four checks change one part of the design and rescore against their own baselines (Section A.1).

Sources and Contested Games. Could the gap come from a feature only one source has? Each source is scored on its own. How should rollouts with the same outcome be compared? Two rollouts can take different sides on a crux and still pick the same winner, and a March Madness game is *contested* when its rollouts do not all pick the same winner.

Strictness of Equivalence. Would a stricter or looser reading of "same crux" close the gap? The headline scores are repeated at the strict and broad levels of Section 5, in both sources.

Specificity and Coarser Clusters. Four checks ask whether the sets miss each other by vagueness rather than by what they ask, by cutting the clustering coarser, asking the predictor for cruxes as specific as the many-shot ones, rewriting the many-shot cruxes vaguer, and matching cruxes on the evidence behind them instead of their wording (Sections A.3 and A.4).

Other Models and Roles. Is the gap a habit of the GPT models? The March Madness chain is rerun on three open-weight models, qwen3:32b, gemma4:31b and gpt-oss:120b. Does a model predict the cruxes of its own forecasts better than another model's? Mixing the GPT and open-weight models lets each role go to a different model.

8 Results

8.1 Predicted against Discovered Cruxes

In March Madness, at the middle level with 25 cruxes on each side, the overlap averages 13.4% $\pm$ 1.6% (SE, the standard error across 59 games) against a permutation baseline averaging 39.2%. Growing both sets never lifts the overlap to the baseline (Figure 4). Two halves of the many-shot cruxes overlap 3.4 times as much as the two sets do (Figure 6 in Section A.3). The gap below the baseline averages 25.9 $\pm$ 1.2 points ($p < 10^{-10}$), and 55 of the 59 games fall below their own baseline, three equal it and one lies above it (Figure 7).

In March Madness the one-shot cruxes come far closer to the baseline on overlap, 18.8% against 25.1%, below it in 47 of the 59 games (Table 7). That gap of 6.4 points is far smaller than the zero-shot cruxes' 26.8 on the same many-shot cruxes. They still fall short on coverage, reaching 28.6% of the many-shot cruxes against a baseline of 37.8% (Table 8). In Metaculus they overlap by 48.5% against 44.0% and sit at the baseline on every measure.

Other model families. In March Madness the gap holds on the GPT runs and, in direction, on each of the three open-weight model families (Table 9; Section A.6).

Other roles. All 27 cells of the grid of three predictors by three extractors at three levels fall below their baselines, and a model predicts the cruxes of its own forecasts at most marginally better than those of another model's forecasts (Section A.6).

8.2 Sources and Contested Games

Two sources. In Metaculus the overlap is 39.9% $\pm$ 4.0% against a baseline of 70.9%, a gap of 31.0 $\pm$ 4.9 points ($p = 1.2 \times 10^{-4}$), and 13 of the 15 events fall below their own baseline (Section A.3).

Contested games. In the 29 contested games the overlap is 16.0% $\pm$ 2.5% against a baseline of 40.5%, and 26 fall below their own baseline, one lies above it and two equal it.

8.3 Strictness of Equivalence

Neither a stricter nor a looser reading of "same crux" closes the gap in either source. At 25 cruxes the gap runs from 25.9 to 36.1 points across the three levels (Table 1; Section A.2).

8.4 Specificity and Coarser Clusters

At every cut coarser than one crux per cluster the March Madness overlap stays below its baseline (Figure 6). Asking for cruxes as specific as the many-shot ones narrows the gap by about a third, and only by lowering the baseline. Its gap at 25 cruxes is 16.4 points, against 24.7 for a same-day rerun of the original prompt. The gap falls by 8.3 points (95% interval 5.1 to 11.5, lower in 42 of 59 games). The overlap stays at about 17% under both prompts, while the baseline falls from about 42% to 33%. The remaining gap holds in 58 of 59 games. Rewriting the many-shot cruxes vaguer, or matching cruxes on the evidence behind them, leaves a gap (Section A.4).

9 Discussion

A model cannot say in advance where its own forecasts will disagree, and one forecast shows only part of the answer. The cruxes of a model's reasoning live in the spread of its rollouts, not in any account it gives of itself, which is why they have to be extracted rather than requested. That extraction is now cheap enough to run as a periodic audit of a deployed model, or as a signal during training, where every prompt already draws many rollouts. Whether a model trained on that signal learns to anticipate its cruxes, or learns to make its forecasts disagree less, is now an affordable experiment.

Limitations. Every crux counts equally here, so the overlap cannot tell a missed crux that would have moved the forecast from one that would not. Weighting each crux by the value of information from resolving it is left open (Section A.1). Would the gap survive that weighting, or is it made of cruxes a decision maker could ignore? The March Madness predictor is a small model prompted a few ways, and the Metaculus predictor never saw the web results the rollouts drew on (Section 3.1). Does a stronger predictor, or more reasoning at prediction time, close the gap? The size of the gap depends on which model extracts the many-shot cruxes (Section A.6), and one reranker, whose instructions were chosen by agreement with a language-model judge rather than with people, decides every match (Section A.2). Is there a canonical set of cruxes at all, or only the set a given extractor and matcher agree on? No human study measures how much any of these cruxes helps a person decide. Would a person given the many-shot cruxes decide differently, or only feel more sure? Every request here is a forecast of one event, which gives two rollouts a number to disagree on. Reports, plans and code changes have no such number. What should two rollouts disagreeing on a plan mean, and can the same contrast find it? The 59 games and 15 events, mostly with GPT models, fix the direction of the result but not its size, above all in Metaculus, where a second draw of the zero-shot cruxes moves the gap by about ten points (Section A.3).

AI USE STATEMENT

In this work, we used generative AI tools for execution of and giving feedback on the experiments, accessing and reformatting datasets, and visualization of results. We have not used generative AI tools for proposing or refining hypotheses, designing the research methodology or experimental design, generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims, implementing the critical software, or translation. Proving mathematical claims, writing proofs, and qualitative or thematic data analysis are not applicable to this work. Specifically, we used generative AI tools for creating and modifying figures, changing hyperparameters, drafting and editing explanatory prose of the paper, and recommending literature. We have reviewed all AI-assisted work. The critical software was written by the authors; every cited paper was checked for relevance and its citation sourced by an author; and every reported number, apart from the cost estimates marked as such, was recomputed from the recorded model outputs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Three artifacts will be public with the camera-ready version. The extraction of cruxes on which forecasts take different sides, the one step that carries optimizations, will be released as a Python package; the scouting and prediction stages are single prompts, given verbatim in Section C.2, and they and the forecasting stage will ship as scripts beside the data. The data release holds every model output the paper analyzes; for Metaculus it holds question identifiers and a fetch script, since the site's terms do not permit redistributing its text. A verification repository reruns every stage downstream of the recorded model outputs from the released inputs alone, on independent hardware, and checks each reported number against a tolerance measured from drift between runs: raw reranker scores within 0.05, overlap and the gap below the baseline within 1.5 times the difference between two draws of the zero-shot cruxes. The runs are dated (Metaculus forecasts 2026-02-28 to 03-01; March Madness scouting reports and forecasts 2026-03-17 to 03-18; extractions 2026-03-19 and 03-31; zero-shot cruxes 2026-03-31 for Metaculus and 2026-09-21 and 09-22 for March Madness), and no call set a temperature, seed or reasoning setting, so the API defaults applied. The March Madness numbers on gpt-5.4-nano use zero-shot cruxes regenerated after an earlier version of the prediction prompt corrupted some of its cruxes; the Metaculus cruxes were not affected, and Section A.7 shows that the corruption did not produce the finding. Section 3 and Table 10 specify the sources and models, Section 5 the matching and clustering, Section 6 the overlap and baseline, Sections A.2, B.2 and D.1 the reranker instructions, costs and example scouting reports, and Figure 2 quotes the load-bearing lines of each prompt.

REFERENCES

Roy Bar-Haim, Lilach Eden, Roni Friedman, Yoav Kantor, Dan Lahav, and Noam Slonim. From arguments to key points: Towards automatic argument summarization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4029–4039, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.371. URL https://aclanthology.org/2020.acl-main.371/.

Jan Betley, Xuchan Bao, Martín Soto, Anna Sztyber-Betley, James Chua, and Owain Evans. Tell me about yourself: LLMs are aware of their learned behaviors. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 21127–21179, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/364b1fd43002260dd1a9502d51d5375e-Paper-Conference.pdf.

Felix Jedidja Binder, James Chua, Tomek Korbak, Henry Sleight, John Hughes, Robert Long, Ethan Perez, Miles Turpin, and Owain Evans. Looking inward: Language models can learn about themselves by introspection. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=eb5pkwIB5i.

Paul C. Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which LLM reasoning steps matter?, 2025. URL https://arxiv.org/abs/2506.19143.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. Reasoning models don't always say what they think, 2025. URL https://arxiv.org/abs/2505.05410.

James Chua and Owain Evans. Are DeepSeek R1 and other reasoning models more faithful?, 2025. URL https://arxiv.org/abs/2501.08156.

Iulia M. Comsa and Murray Shanahan. Does it make sense to speak of introspection in large language models?, 2025. URL https://arxiv.org/abs/2506.05068.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Danny Halawi, Fred Zhang, Chen Yueh-Han, and Jacob Steinhardt. Approaching human-level forecasting with language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=FlcdW7NPRY.

Ronald A. Howard. Information Value Theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, August 1966. doi: 10.1109/TSSC.1966.300074.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know, 2022. URL https://arxiv.org/abs/2207.05221.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Katherine Collins, Arka Pal, Umang Bhatt, Adrian Weller, Samuel Dooley, Micah Goldblum, and Andrew Gordon Wilson. Large language models must be taught to know what they don't know. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Rudolf Laine, Bilal Chughtai, Jan Betley, Kaivalya Hariharan, Mikita Balesni, Jérémy Scheurer, Marius Hobbhahn, Alexander Meinke, and Owain Evans. Me, myself, and AI: The situational awareness dataset (SAD) for LLMs. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=UnWhcpIyUC.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words, 2022. URL https://arxiv.org/abs/2205.14334.

Jack Lindsey. Emergent introspective awareness in large language models, 2026. URL https://arxiv.org/abs/2601.01828.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12076–12100, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.741. URL https://aclanthology.org/2023.emnlp-main.741/.

Kevin Murphy. Agentic forecasting using sequential Bayesian updating of linguistic beliefs, 2026. URL http://arxiv.org/abs/2604.18576.

Duncan Sabien. Double crux — A strategy for mutual understanding. LessWrong, 2017. URL https://www.lesswrong.com/posts/exa5kmvopeRyfJgCy/double-crux-a-strategy-for-mutual-understanding. Originally developed as part of the CFAR (Center for Applied Rationality) curriculum.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: unfaithful explanations in chain-of-thought prompting. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Zixia Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V Le. Long-form factuality in large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=4M9f8VMt2C.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena, 2023. URL https://arxiv.org/abs/2306.05685.

The appendices come in five parts: the robustness checks in detail, the models and costs, the data and prompts, some example model outputs, and related work in detail.

A Robustness Checks in Detail

A.1 Design detail for the checks

What the gap below the baseline does not show. The test does not show any zero-shot crux to be false (Section 9). They read as sensible questions about the game or event, and many may bear on its outcome. An unmatched zero-shot crux may still split some pair of rollouts and fall among the many-shot cruxes beyond the 25 scored. The gap says nothing about any one crux. The overlap counts every crux equally, so it cannot tell a missed crux that would change the forecast from one that would not. If the cruxes absent from the zero-shot set would barely move the forecast, the gap would matter less to a person deciding than its size suggests. Replacing the extractor gpt-5.4-nano with qwen3:32b on the same forecasts moves the gap at 25 cruxes by about 11 points (Section A.6). So the scoring can detect a change of that size. One objection is that the reranker keeps apart a crux that names a factor and a crux that gives a number for the same fact. The same replacement answers this objection in part. From the same forecasts qwen3:32b writes 59 many-shot cruxes per game against 160 for gpt-5.4-nano. With qwen3:32b as the extractor, every combination of models and level still falls below its baseline. The replacement changes the model as well as the number of cruxes, though.

Two open objections. Two objections stay open. One is weight, since every crux counts equally. Weighting each crux by the value of information (Howard, 1966) from resolving it would answer this objection and is left to future work. The other is vagueness, since part of the difference between the sets lies in how specifically the cruxes are written. Asked for cruxes as specific as the many-shot ones, the predictor complies, and the March Madness gap narrows from 24.7 to 16.4 points at 25 cruxes (Sections A.4 and 8.4). The overlap stays at about 17% and only the baseline falls, so the remaining 16 points are not a matter of wording.

Sources. Rollouts that read different reports can split over a fact one report states and the other leaves out, so some March Madness cruxes may trace to how the reports were written. A predictor that reads all the reports at once, rather than one on each team, could miss those cruxes, so a gap found only in March Madness could come from the reports. In Metaculus the predictor sees less than the rollouts do (Section 3.1), so a gap found only there could come from what the predictor never saw. The sources differ in more besides, including topic, how many rollouts each game or event gets, the prediction prompts, and which model plays each role (Table 10). A difference in the gap's size could come from any feature the sources do not share.

Contested games. Two rollouts can take opposite sides on a crux and still pick the same winner. For example, one rollout may back a team because it expects the team to commit few turnovers, while another expects many turnovers and backs the same team for its rebounding. A contested game also holds pairs that pick the same winner, so its many-shot cruxes mix theirs with those of pairs that split on the winner. The check shows whether the gap holds where rollouts do disagree over the winner, but it cannot isolate the cruxes of the pairs that split on it.

Strictness. A strict reading splits cruxes that a loose reading would merge. If the gap came from reading "same crux" too strictly, it could shrink under an instruction that asks for less. The order strict, middle, broad follows what each instruction asks of a match. A looser instruction is meant to score more pairs as matching, but each level's scores are clustered and cut anew, so the broad level need not end with larger clusters than the middle one, and each level gets its own baseline (Section 6).

Coarser clusters. The middle level can keep a vague crux apart from its specific counterpart, merging two cruxes only when resolving one would necessarily resolve the other. Learning how fast world demand for oil will grow settles whether it will still be growing, but learning only that it will still be growing leaves the rate open, so a vague crux and a specific one about the same fact can miss each other at that level.

In March Madness, at the middle level, a sweep loosens matching without changing either set. At each coarser cut two cruxes need to share less to fall together (Section A.3). Each cut gets its own baseline, computed on that cut's clusters.

Scope of the contested games. In each contested game some rollouts pick different winners, so whether the predictor reads *disagree* as taking different sides on a crux or as picking different winners, it names cruxes of a kind on which the rollouts do split. A March Madness pick is a conclusion two rollouts either share or do not, and a Metaculus probability is not, so the check runs on March Madness only.

Choosing the instructions. The three instructions, given in Section A.2, were chosen from 10 candidates by agreement with a language model acting as a judge (Zheng et al., 2023) (Section A.2).

Qualitative patterns. In March Madness the quoted cruxes differ in which part of the game they concern as well as in how precisely they state it. For Iowa State against Kentucky, the zero-shot crux quoted in Figure 1 asks whether free-throw performance becomes a decisive factor. The many-shot cruxes include some whose assumptions state the number of steals by Tamin Lipsey, Iowa State's point guard, and the number of turnovers Iowa State forced in an earlier game. These three quoted cruxes alone cannot show which of the two differences keeps them from matching.

On Metaculus, many-shot cruxes can cite facts that appear nowhere in the prediction prompt. For the Frontier event of Figure 1, both sets name cash on hand, debts coming due and demand for air travel. The zero-shot cruxes stay general, asking whether Frontier keeps enough cash to avoid filing for bankruptcy and whether weak demand for air travel hurts its revenue. The many-shot cruxes cite the bankruptcy restructuring of Spirit Airlines, another budget airline in the United States, and Frontier's Altman Z-score. The rollouts found both through web search, which the predictor does not see (Section 3.1), so it could have named them only from what it learned in training.

Specific prompts. A crux can name a factor, such as turnovers, make a claim about it, such as that Kentucky turns the ball over often, or state a value, such as Kentucky's turnover rate of 12.0%. A local model tags every crux in the deduplicated pools with one of these three levels, and the tags show

where the two sets differ. In March Madness 66% of zero-shot cruxes make a claim and 32% state a value, while 91% of many-shot cruxes state a value. In Metaculus 89% of zero-shot cruxes make a claim and 44% of many-shot cruxes state a value. Almost no crux on either side names a bare factor.

Other models. The open-weight and mixed runs use other extractors, and the extractor changes the size of a gap, so they are compared with the main result in direction only.

A.2 Robustness across levels of strictness

The gap below the baseline holds at all three levels of strictness, though its size changes with the level. Figure 5 shows the decomposition at all three levels in both sources, and its middle column repeats Figure 4. In March Madness the shared band stays the thinnest of the three at every level. Table 1 compares the three levels in both sources at $k = 25$.

An example of the three levels. The oil cruxes show what the broad level adds. Whether world demand for oil will still be growing in ten years and whether it will have peaked by then ask one question, so they match at every level. Whether electric cars will make up most new car sales in ten years can be settled without settling demand, so it misses the demand cruxes at the middle level. Two rollouts that disagree over demand because one expects electric cars to take over could have that dispute ended by settling either, so at the broad level it matches them.

Level and cut. The level changes what the reranker is asked and the cut changes how many merges are kept, so the two loosen matching in different ways, and the main results fix both.

Which cruxes are scored. Each set of cruxes is shuffled once, with a seed derived from the game or event and the set, and the first k cruxes of that shuffle enter, so the cruxes scored at 12 are among those scored at 25. A crux that appears more than once in a set is kept once before the set is scored, and Section A.6 names the runs that skip this step. Every set runs past 25 cruxes: after repeats are removed, March Madness sets hold 63 to 86 zero-shot and 137 to 197 many-shot cruxes per game, and Metaculus sets 243 to 359 and 907 to 1,714 per event, so only part of each set is scored, and at 25 a Metaculus set contributes a small sample of what it holds.

The three boxes below give the reranker instructions for each level, in order of strictness. Each box prints the level's positive instruction, which asks whether two cruxes match, and its negative instruction, which asks whether they differ. Every reported number uses the positive instruction only, scored in both orders as Section 5 describes. A many-shot crux records which of its two answers, A or B, each of its two rollouts assumed. A zero-shot crux is written without the predictor seeing any rollout, so its labels follow no rollout. It can put first the answer that a matching many-shot crux puts second. Every positive instruction therefore tells the reranker that the labels may be swapped.

Reranker: same-question matching (strict)

Positive instruction. Given a crux question about a forecasting problem, determine whether the following crux question addresses the same underlying point of disagreement, even if worded differently. The answer labels (A/B) may be swapped.
Negative instruction. Given a crux question about a forecasting problem, determine whether the following crux question addresses a DIFFERENT point of disagreement. Consider them different even if they share vocabulary, as long as the core factual question is distinct.

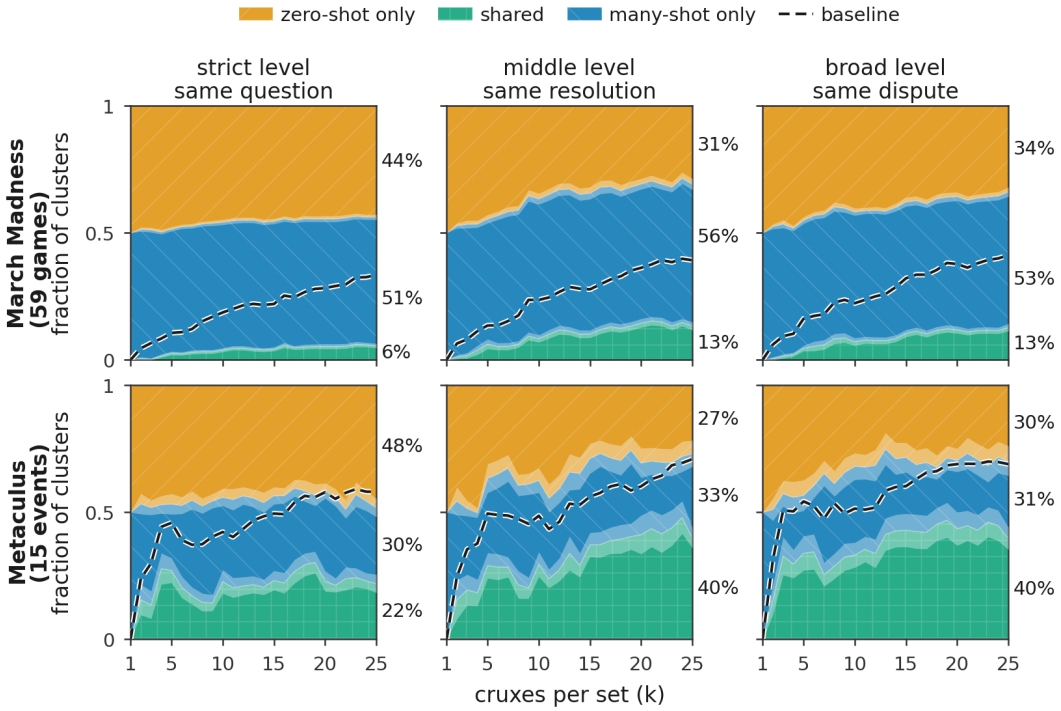


Figure 5: Whether two cruxes must ask the same question or need only concern the same dispute, predicted cruxes match discovered ones less often than the permutation baseline. Stacked-area decomposition at the strict level (same question), the middle level (same resolution) and the broad level (same dispute), for March Madness (59 games, top row) and Metaculus (15 events, bottom row), in the colours of Figure 4. The numbers at the right of each panel give its three shares at 25 cruxes. At 25 cruxes the March Madness overlap is 5.6%, 13.4% and 12.8% against baselines of 33.4%, 39.2% and 41.2%, and the Metaculus overlap is 21.9%, 39.9% and 39.7% against 58.0%, 70.9% and 68.9%.

Table 1: At every level of strictness the gap below the permutation baseline is significant in both sources, and nearly every game and event falls below its own baseline. Overlap and baseline at $k = 25$ at each level of strictness, each level against its own baseline, for March Madness (59 games) and Metaculus (15 events). Overlaps carry $\pm$ 1 SE across games or events. Gaps are in points, p is from a one-sided Wilcoxon signed-rank test across games or events, and the last column counts games or events below their own baseline; the rest equal it, apart from one March Madness game at the middle level that lies above.

Source	Level	Overlap (%)	Baseline (%)	Gap	p	Below
March Madness	strict	5.6 ± 0.9	33.4	27.8	$< 10^{-10}$	59 / 59
	middle	13.4 ± 1.6	39.2	25.9	$< 10^{-10}$	55 / 59
	broad	12.8 ± 1.5	41.2	28.4	$< 10^{-10}$	57 / 59
Metaculus	strict	21.9 ± 3.6	58.0	36.1	6.1×10^{-5}	14 / 15
	middle	39.9 ± 4.0	70.9	31.0	1.2×10^{-4}	13 / 15
	broad	39.7 ± 4.4	68.9	29.3	2.4×10^{-4}	12 / 15

Reranker: same-resolution matching (middle)

Positive instruction. Two crux questions are equivalent if resolving one would necessarily resolve the other, because they ask about the same underlying factual question. Ignore differences in wording, specificity, or answer-label ordering (A/B may be swapped). Does the following crux question target the same factual uncertainty?
Negative instruction. Two crux questions are different if they could be resolved independently — knowing the answer to one would not determine the answer to the other. Even if they use similar vocabulary or concern the same topic, determine whether the following crux question targets a genuinely different factual uncertainty. Answer labels (A/B) may be swapped.

Reranker: same-dispute matching (broad)

Positive instruction. Imagine two forecasters disagreeing. If resolving the query crux would settle their specific point of contention, would resolving the following crux question settle the exact same point of contention? Two cruxes are equivalent if they would end the same debate. Ignore wording differences and answer-label order (A/B may be swapped).
Negative instruction. Imagine two forecasters disagreeing. If resolving the query crux would settle one point of contention, would the following crux question address a different point of contention — one that could persist even after the first crux is resolved? Two cruxes are different if they would end different debates. Answer labels (A/B) may be swapped.

Overlap at the strict and broad levels. Figure 5 in Section A.2 shows the overlap at all three levels as each set grows. In both sources the broad level gives a slightly lower overlap than the middle one.

Validation. The three reranker instructions above were chosen from 10 candidate instructions by balanced accuracy on 2,000 crux pairs that gpt-5.4-nano, as a judge with a two-step rubric (map the two cruxes' answers onto each other, then compare their scope), had labeled as matching or not. The pairs were drawn at random from the pooled 25 plus 25 cruxes of the 45 resolved March Madness games of an earlier run, 1,018 of them across the two sets and 982 within one set, and the judge called 34 matching. Balanced accuracy is the mean of two recalls, the share of judge-matched pairs the cut puts in one cluster and the share of judge-distinct pairs it keeps apart. Same question scored 0.845, same resolution 0.826 and same dispute 0.840. Only 3 of the 34 pairs the judge marked as equivalent cross the two sets, so the judge's accuracy on cross-set pairs, the ones that decide the overlap, cannot be measured from this set.

The judge's labels have not been compared directly with human labels, so the instructions were chosen by agreement with a language model and not with human judgment.

A.3 Coarser cuts and split halves

Two objections to the headline number can be answered from the same scored cruxes, without new model calls. The first is that the low overlap may be an artifact of where the clustering is cut, so that looser matching would let the two sets share more clusters (Section 8.4). The second is that the many-shot cruxes may be too varied for any set of cruxes to match them (Section 8.1).

Overlap at every cut. Cutting one tree at a series of points changes how loosely cruxes are matched. The pooled zero-shot and many-shot cruxes are clustered once. That tree is then cut at cluster counts running from a small fraction of the cruxes up to one cluster per crux. A coarse cut lumps cruxes into broad categories. The finest cut leaves every crux alone. Only the cut moves, so re-clustering cannot confound the sweep. Figure 6 draws the sweep, and Section 8.4 reports what it shows.

Two halves of one set. Each set is clustered on its own, split at random into two halves, and the two halves are compared exactly as the zero-shot and many-shot cruxes are. Averaged over random splits, the split-half overlap plays the role of the baseline for one set alone, though it reclusters real halves rather than shuffling labels. Figure 6 draws the check, and Section 8.1 reports what it shows.

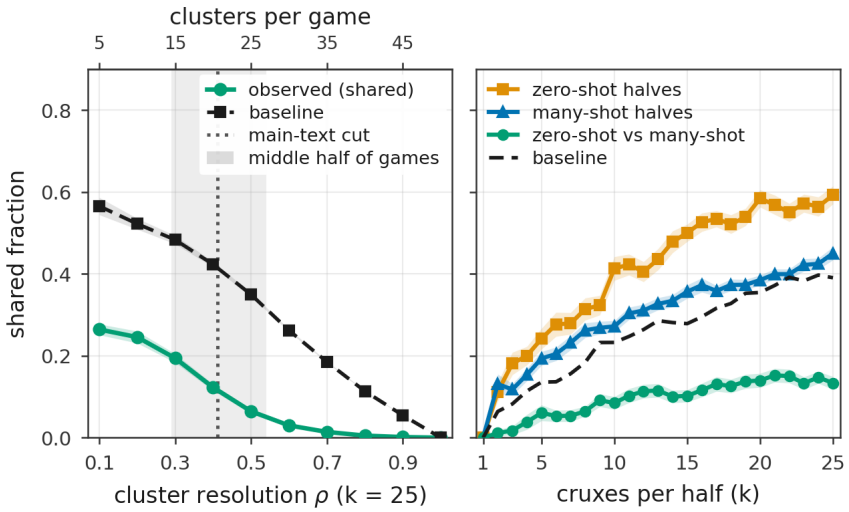


Figure 6: In March Madness, however coarsely cruxes are grouped, predicted cruxes match discovered ones less often than the permutation baseline, though halves of the discovered cruxes match each other. **Left:** Coarsening the clustering does not close the gap below the baseline. The overlap and its baseline across cuts of the tree, from broad categories on the left to one crux per cluster on the right (March Madness, middle level of strictness, 25 cruxes per set). The horizontal axis is ρ, the number of clusters divided by the number of pooled cruxes, and the top axis gives the clusters per game. The dotted line marks the cut used in the main text, a mean of 20.5 clusters per game ($\rho = 0.41$), and the grey band the middle half of games. **Right:** Two halves of either set share several times the overlap the two sets share with each other. Overlap against the number of cruxes drawn from each set, for two halves of the zero-shot cruxes, two halves of the many-shot cruxes, and the zero-shot cruxes against the many-shot cruxes. The dashed line gives the permutation baseline for the zero-shot against many-shot comparison (March Madness, middle level of strictness). At 25 cruxes two halves of the many-shot cruxes sit just above the dashed baseline, two halves of the zero-shot cruxes well above it, and the two sets far below it. So the discovered cruxes are not too scattered for a set of cruxes to match them.

Two halves of the zero-shot cruxes overlap 4.4 times as much, so the predictor's own cruxes also cluster with one another.

Baselines and single games in March Madness. The baseline rises because a larger pool gives clusters more cruxes, and a cluster with more cruxes is more likely to receive both labels when they are shuffled. Here below means lower than the mean of the game's own shuffles, a comparison of two numbers and not a test run on each game. A game can equal its baseline when shuffling the labels cannot change its overlap; the three games that do so each end with two clusters, one holding every crux but one, so no shuffle moves the overlap. The median game shares 10.5% of its clusters and nine share none, so the mean of 13.4% overstates what a typical game shares.

Metaculus events. Metaculus rollouts resemble what a person gets by running one forecasting system on one request many times. In Metaculus the predictor is not the forecaster, so the gap holds when a model predicts another model's cruxes, while the test of a model predicting its own cruxes rests on March Madness.

In Metaculus, as in March Madness, the gap holds for sets of 5 to 25 cruxes (Figure 4, right). It is already there at 5 cruxes, where the overlap is 29.0% $\pm$ 5.0% (SE across 15 events) against a baseline of 49.5% ($p = 0.0015$, one-sided Wilcoxon signed-rank test). Metaculus baselines run higher than those of March Madness, so the two sources are compared by their gaps. In Metaculus, 13 of the 15 events fall below their own baseline. The other two equal theirs, since the cut leaves each with exactly two clusters, 49 cruxes in one and a lone many-shot crux in the other, so no shuffle of the labels can move the overlap. The gap averaged over events therefore describes the typical event.

The $\pm$ 4.9 on the Metaculus gap of 31.0 points (Section 8.2) covers variation across events only. A second draw of the zero-shot cruxes moves the gap by as much as ten points, so with only 15 events

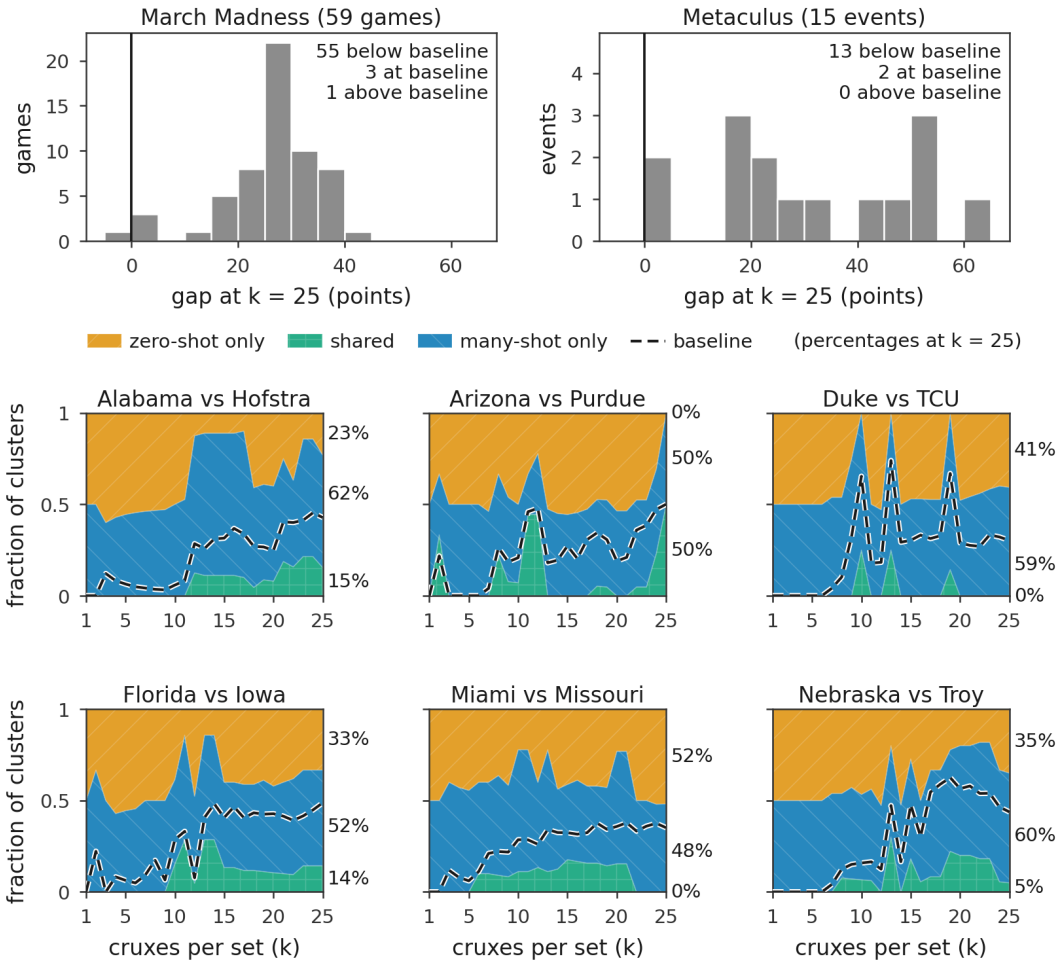


Figure 7: Predicted cruxes match discovered ones less often than the permutation baseline in nearly every game and event, so the mismatch is the typical case. **Top:** the gap of each game and event at 25 cruxes, its own baseline minus its overlap, with a line at zero. In March Madness 55 of 59 games fall below their own baseline, 3 equal it and 1 lies above it; in Metaculus 13 of 15 events fall below, 2 equal it and none lies above. **Bottom:** six March Madness games chosen at random (seed 42), as each set grows to 25 cruxes, in the colours of Figure 4 and with each game's own baseline as the dashed line. The shared band rises and falls with set size and differs from game to game. Middle level of strictness throughout.

Table 2: Asked for specific cruxes, the predictor states values at least as often as the discovered cruxes do, while its usual cruxes mostly do not. Share of cruxes at each level of specificity, in percent, for the cruxes as written and for each prompt. The factor prompt of the ladder failed its check, so the ladder separates claims from values only.

Source	Cruxes	Factor	Claim	Value
March Madness	predicted, original prompt	1.9	66.4	31.7
	predicted, same-day control	1.8	66.5	31.7
	predicted, specific prompt	0.1	3.3	96.6
	predicted, exemplar prompt	0.2	2.8	97.0
	predicted, ladder, factor prompt	9.3	68.1	22.6
	predicted, ladder, claim prompt	1.9	77.2	20.9
	predicted, ladder, value prompt	0.3	4.7	95.0
	extracted (many-shot)	0.1	9.3	90.6
Metaculus	predicted, original prompt	0.6	88.8	10.6
	predicted, same-day control	0.4	87.2	12.4
	predicted, specific prompt	0.1	14.2	85.7
	predicted, exemplar prompt	0.1	18.2	81.6
	extracted (many-shot)	0.8	55.6	43.6

its size is unstable. Both draws fall below the baseline at every strictness level and for sets of 5 to 25 cruxes, so its direction is stable.

The overlap varies widely across events, with a standard deviation of 15.6 points.

Set size and spread. In March Madness the overlap rises from 6.3% $\pm$ 1.9% (SE) at 5 cruxes, against a baseline of 13.5%, to 13.4% at 25, against 39.2%. The overlap varies widely across games, with a standard deviation of 12.1 points (Figure 7, bottom). Iowa State vs. Kentucky, the game in the left panel of Figure 1, shares 7.1%. The Frontier Group event, in the right panel of Figure 1, lies at the low end, with an overlap of 8.3%.

A.4 Specific prompts and coarsening

Section 7 names the check, and this appendix gives the detail and numbers of its arms, all at the middle level of strictness. Each crux in the deduplicated pools was tagged as naming a factor, making a claim or stating a value by qwen3:32b run locally.

The predictor is re-run with a prompt that asks it to state each crux at the specificity of a value, and its cruxes are scored exactly as in Section 6. A control arm re-runs the original prompt on the same day, so that drift between draws can be told from the effect of the prompt. Two further arms are reported for completeness: an exemplar arm that shows the predictor cruxes extracted from another game's rollouts, which changes the writing style as well as the specificity, and a ladder of three prompts asking for factor, claim and value cruxes in turn. The same tags check whether each prompt did what it asked.

The coarsening check works from the other side, on the cruxes as written. A local model rewrites each many-shot crux one level vaguer, from a value to a claim, with the zero-shot cruxes untouched, and in a second arm rewrites both sets toward the factor level. A third comparison restricts both sets to cruxes tagged at the same level and scores the restricted pools on the games where both sets hold enough.

Table 2 gives the shares.

Under the specific prompt the March Madness overlap is 0.169 in both the control and the specific arm, and the baseline falls from 0.417 to 0.333. The exemplar arm sits 4.3 points below the specific arm ($p = 4 \times 10^{-4}$), but the writing style of the extractor and the facts of the donor game enter the cruxes it predicts, so the specific arm is the clean number. The same-day control's overlap moved 0.036 from the headline draw, a hair outside the 0.0355 tolerance the run uses for drift, while its gap moved 1.1 points. In Metaculus, 2% of control cruxes and 6% of specific and exemplar cruxes came from calls whose question and answer arrays had unequal lengths; scoring without them gives the same reading.

Table 3: In March Madness, asking for specific cruxes narrows the gap by about a third only by lowering the permutation baseline, and no prompt closes it. Gap in points under each prompt, at 25 and 12 cruxes per set. March Madness p values are two-sided Wilcoxon signed-rank tests of control minus arm over 59 games. In Metaculus the drift between the original and the same-day control (7.8 points at $k = 25$) exceeds every difference between arms. Every row, the original included, uses the seed of the arm runs for the baseline's label shuffles. That seed is the same for every game or event. The runs of Table 1 instead give each game or event its own seed. With that seeding the original prompt's gaps are 25.9 and 15.0 in March Madness and 31.0 and 18.8 in Metaculus.

Source	Prompt	$k = 25$	$k = 12$
March Madness	original (headline, one seed)	25.8	14.8
	same-day control	24.7	15.0
	specific	16.4 ($p = 1.4 \times 10^{-5}$)	10.1 ($p = 0.0012$)
	exemplar	12.1 ($p = 1.4 \times 10^{-8}$)	7.8 ($p = 5.2 \times 10^{-6}$)
	ladder, factor	27.4 ($p = 0.10$)	16.1 ($p = 0.57$)
	ladder, claim	23.0 ($p = 0.40$)	12.5 ($p = 0.043$)
	ladder, value	15.7 ($p = 2.7 \times 10^{-6}$)	8.4 ($p = 3.2 \times 10^{-5}$)
Metaculus	original (headline, one seed)	31.4	18.7
	same-day control	23.6	24.1
	specific	31.0	16.9
	exemplar	23.3	17.2

Table 4: Making cruxes vaguer, or comparing only cruxes of equal specificity, leaves predicted cruxes matching discovered ones less often than the permutation baseline. Gap in points when cruxes are rewritten to be vaguer, and when both sets keep only cruxes at the same levels of specificity. The original column uses the full set of many-shot cruxes, none rewritten or removed, on the same games or events. Both columns shuffle the two labels with one fixed seed for all games or events. So the original column gives 25.8 and 14.8 in March Madness and 31.4 in Metaculus. The main text gives each game or event its own seed and reports 25.9, 15.0 and 31.0.

Source	Arm	k	Original	Arm
March Madness	many-shot cruxes, rewritten one level vaguer	25	25.8	26.9 ($p = 0.57$)
		12	14.8	16.8
	both sets rewritten toward factors	25	25.8	15.3 ($p = 1.3 \times 10^{-6}$)
		12	14.8	11.0 ($p = 0.008$)
	both sets restricted to values (24 games)	25	23.9	22.4
	both sets restricted to values (53 games)	12	15.6	15.1
	both sets restricted to claims or values	25	25.8	28.4
Metaculus	many-shot cruxes, rewritten one level vaguer	25	31.4	25.2
	both sets rewritten toward factors	25	31.4	29.4
	both sets restricted to values (7 events)	25	24.3	22.9

Table 4 gives the coarsening arms. The rewrites toward the factor level mostly stopped at the claim level (in a spot check, 9 of 9 value cruxes reached a claim, 3 of 11 reached a factor, and 0 of 10 claims reached a factor), so the both-sides arm tests claim-level coarsening of both sets, and one local model rewrote both sides, so a shared writing style can inflate its overlap. No arm moves the gap toward zero; the smallest gap in any arm is 11.0 points ($p = 5.8 \times 10^{-8}$).

Coarser cuts and specific prompts. Overlap and baseline meet only at one crux per cluster, where no cluster can hold cruxes from both sets. Looser matching therefore does not close the gap.

Evidence matching. A third granularity check asks whether the gap shrinks when cruxes are matched on the evidence they rest on rather than on their wording. For every crux in both sets, gpt-5.4-nano wrote a statement of the evidence the crux rests on, in two batch runs. The overlap was then computed again, with the reranker matching these statements instead of the cruxes' own wording. The reranker used three wordings of its instruction (strict, moderate and broad), each chosen by agreement with a language-model judge, qwen3:32b. As an anchor, the reranker also matched the same cruxes on their own wording, under the middle-level instruction of the main text. The clustering, the cut, the seeding and the baseline are computed as in the main results. The judge and the reranker

Table 5: Matching cruxes on the evidence behind them, instead of on their wording, still leaves predicted cruxes matching discovered ones less often than the permutation baseline. Gap in points when cruxes are matched on their written evidence under three wordings of the matching instruction, beside matching on the cruxes' own text on the same pools (March Madness, 59 games). The full set of many-shot cruxes gives 25.9 points at 25 cruxes.

Cruxes per set	Strict	Moderate	Broad	Text, same pools
25	20.3	24.3	20.1	22.9
12	13.9	18.2	16.0	13.7

both ran on a local machine. Table 5 gives the gaps for the 59 March Madness games. Five of the six differences from the anchor lie inside the run's drift tolerance of 3.55 points; the exception is the moderate wording at 12 cruxes, where the gap is 4.4 points larger (95% interval 1.6 to 7.2, exact $p = 0.005$). Between 45 and 58 of the 59 games stay below the baseline under every wording (exact one-sided p at most 3×10^{-14}), so matching on evidence does not close the gap, and where it moves it, it widens it. All six wordings were scored in full, and the three reported were then chosen on 580 judged pairs; only the strict pick is settled by the data (balanced accuracy 0.906 against 0.804 for its alternative), while the moderate and broad picks lie within their intervals of their alternatives (0.664 against 0.645, 0.642 against 0.603), and the alternative wordings give gaps of 22.5, 19.2 and 16.4 points at 25 cruxes and 16.9, 19.7 and 22.7 at 12, so the reading does not depend on the pick. Two diagnostics bear on the reading. The judge finds almost no evidence matches across the two sets: of 300 cross-set pairs it called 1 a strict match, 11 moderate and 22 broad, so the calibration sample had to be filled with within-set pairs (280 pairs), and of the 2,363 cross-set pairs the paper's instruction matches on wording, 11.9%, 19.0% and 43.1% also match on their written evidence at the three wordings. A keyword proxy says why: 75% of the many-shot cruxes' evidence statements name a season statistic, rank or record, against 11% of the zero-shot cruxes'. The two sets rest their cruxes on different kinds of evidence, not only on different wording.

The specific prompt does what it asks, since 97% of the cruxes it returns state a value, against 32% under the original prompt. The specific prompt leaves a gap of 16.4 points and the ladder's value prompt one of 15.7. The exemplar prompt narrows the gap further, to 12.1 points, though it changes the writing style along with the specificity (Section A.4). In Metaculus every arm stays below its baseline (the specific prompt in 13 of 15 events), and the differences between arms sit inside the drift between two draws, so 15 events can neither confirm nor contradict the March Madness fall.

Rewriting each many-shot crux one level vaguer, with the zero-shot cruxes untouched, gives 26.9 points against 25.8; restricting both sets to cruxes that state values gives 22.4 against 23.9 on the 24 games where both sets hold enough. Only rewriting both sets with the same model narrows the gap, to 15.3 points, still far below the baseline, and that arm also gives both sets one writing style (Section A.4).

A.5 The one-shot cruxes against the many-shot cruxes

Selection and leakage. One stored forecast per game or event is picked by a seeded rule (seed 42 hashed with the question identifier, modulo the number of forecasts) and the one-shot cruxes are extracted from it on their own with a single-forecast prompt (Section C.2). The extractor is the same pinned model as the many-shot extraction in each source, gpt-5.4-nano-2026-03-17 for March Madness and gpt-5.4-mini-2026-03-17 for Metaculus, through the batch API with a 5,000-token output cap and no sampling parameter set, and every stored forecast was rendered exactly as the pairwise extractor saw it. The prompt carries the pairwise prompt's rubric and returns a strict JSON object with an analysis of the forecast and at most 30 cruxes, most important first, each a neutral question with the premise the forecast relies on and its contrary; the one-shot cruxes are the questions in the returned order. A forecast yields about 12 cruxes in March Madness and about 6 in Metaculus, always fewer than 25, so at $k = 25$ the whole forecast is scored against 25 pooled cruxes and the baseline is computed with that smaller label count. In the full set of many-shot cruxes, 22.7% of the March Madness cruxes (5.3% in Metaculus) came from pairs that include the chosen forecast, and removing them gives the stripped many-shot cruxes. They are removed, and both the one-shot cruxes and the zero-shot cruxes are scored against those stripped many-shot cruxes, since removing items changes which 25 pooled cruxes the seeded selection draws. That alone moves the gap of the Metaculus zero-shot cruxes from 31.0 points on the full many-shot cruxes to 21.9 on the stripped

Table 7: Cruxes from one forecast come far closer to the permutation baseline than predicted cruxes, so scoring one set of cruxes against a pooled one does not explain the gap. Overlap against baseline at $k = 25$, middle level, the one-shot cruxes and the zero-shot cruxes scored on the same stripped many-shot cruxes. Gaps are in points, the counts are games or events below, above and at their own baseline, and p is from the one-sided exact Wilcoxon test.

Source	Cruxes	Overlap vs baseline	Gap	Below / above / tied, p
March Madness	one-shot	18.8 vs 25.1	6.4	47 / 11 / 1, 4×10^{-9}
March Madness	zero-shot	14.6 vs 41.4	26.8	56 / 0 / 3, 10^{-17}
Metaculus	one-shot	48.5 vs 44.0	-4.5	2 / 9 / 4, 0.94
Metaculus	zero-shot	35.1 vs 56.9	21.9	11 / 1 / 3, 5×10^{-4}

many-shot cruxes. In March Madness the leakage mattered for the one-shot cruxes (gap 3.2 points with the leaked items in, 6.4 with them out); in Metaculus the conclusion is the same either way. Everything else, the reranker, the instructions, the clustering, the cut, the deduplication and the shuffle, is identical to the headline run.

Table 6 lists what changes between the pairwise extraction prompt and the single-forecast prompt of Section C.2; the four-criterion rubric, the world-fact and neutrality rules, the output shape and the closing quality reminder are the same.

Table 6: The single-forecast extraction prompt asks what one forecast rests on where the pairwise prompt asks where two forecasts disagree. Every part of the two prompts that differs, in descending order of how much the change matters; all else is shared.

Part of the prompt	Pairwise (many-shot) extraction	Single-forecast (one-shot) extraction
Task statement	none; the rubric leads straight into the steps	"You are given ONE argument: a single forecast for the question below, with its reasoning. Identify the cruxes of this argument: the factual premises about the world that its forecast turns on, so that if a premise were false, the forecast would change."
Distillation	"Distill any differences into cruxes"	"Distill the premises its forecast depends on into cruxes"
option_A and option_B	Argument A's premise and Argument B's premise, contradicting along the crux dimension	the premise this argument relies on and the contrary premise, "so that if it were true the forecast would change"
Analysis step	analyze both arguments, do the same analysis for Argument B, then for each dimension note whether the arguments differ	analyze the one argument on the same four dimensions, with no comparison step
Maximum and order	none stated	"Return at most 30 cruxes, most important first: the premises whose reversal would move this forecast the most"
Minimum output	"You MUST return at least one crux if the arguments rely on different world-facts, even if they reach the same final answer"	no minimum
Per-criterion rationale	required for each crux	dropped

Results. Table 7 gives the overlaps. Paired by question, the one-shot cruxes' gap is 20.4 points smaller than the zero-shot cruxes' in March Madness ($p = 3 \times 10^{-15}$) and 26.3 points smaller in Metaculus ($p = 6 \times 10^{-5}$). At $k = 5$, 12 and 25 the one-shot cruxes' gap on the stripped many-shot cruxes is 0.8, 3.4 and 6.4 points in March Madness and 4.5, -1.8 and -4.5 in Metaculus, where it is never significant (p from 0.25 to 0.94). Under the strict and broad levels, on the same stripped many-shot cruxes, the one-shot cruxes' gap is 6.6 ($p = 2 \times 10^{-12}$) and 7.0 ($p = 6 \times 10^{-10}$) points in March Madness against 28.1 and 30.6 for the zero-shot cruxes, and in Metaculus -7.0 and -4.8, above the baseline at the strict level (13 of 15 events, $p = 0.004$ in that direction) and at it at the broad level ($p = 0.94$), against 33.3 and 21.3 for the zero-shot cruxes. A separate check extracts every forecast on its own and pools the sets of cruxes of each game or event (9 per game, 29 or 30 per event), which tests the extraction method rather than one forecast. On the full many-shot cruxes at $k = 25$ that pool overlaps by 28.2% against 30.8% in March Madness (gap 2.7, $p = 9 \times 10^{-4}$)

Table 8: Predicted cruxes cover far less of the discovered cruxes than the permutation baseline would, and in March Madness cruxes from one forecast fall short too. Containment at $k = 25$, middle level, each direction against its own baseline. In pool is the share of the set's cruxes whose cluster holds a pooled crux, covers pool the share of the pooled cruxes whose cluster holds one of the set's; gaps are in points, p is one-sided exact Wilcoxon, and MM is March Madness.

		In pool				Covers pool			
Source	Cruxes, pool	obs	base	short	p	obs	base	short	p
MM	zero-shot, headline	32.7	65.3	32.5	10^{-12}	23.2	65.3	42.1	2×10^{-18}
MM	one-shot, stripped pool	36.7	57.7	21.0	3×10^{-14}	28.6	37.8	9.2	8×10^{-8}
MM	zero-shot, stripped pool	41.1	68.3	27.3	10^{-9}	25.7	68.4	42.7	2×10^{-18}
MM	per-forecast pooled, headline	40.9	52.2	11.3	3×10^{-12}	55.0	52.1	−2.9	0.95
MM	split half	68.8	68.7	−0.1	0.73	65.9	68.5	2.6	0.035
Met.	zero-shot, headline	76.8	95.1	18.3	0.0013	75.2	95.2	20.0	0.001
Met.	one-shot, stripped pool	87.9	90.9	3.0	0.23	82.9	79.6	−3.3	0.85
Met.	zero-shot, stripped pool	75.5	93.6	18.1	0.013	80.3	93.7	13.4	3×10^{-4}
Met.	per-forecast pooled, headline	89.1	95.0	5.9	0.032	98.1	95.0	−3.1	1

and by 60.5% against 65.0% in Metaculus ($p = 0.21$). Extracting one forecast at a time and pooling therefore gives cruxes close to those extracted from pairs of forecasts, though 2.7 points short of the baseline in March Madness. The tests count residues of 10^{-9} or less as ties (Metaculus at $k = 5$, for example, is 10 / 1 / 4 with $p = 0.001$).

What the overlap does not see. That the one-shot cruxes sit at the baseline means they are exchangeable with the many-shot cruxes at the resolution of the clustering, not that one forecast is as good as the pool. The overlap never sees how often a crux recurs across forecasts, whether a crux separates forecasts that disagree, or how much of the pool a set of cruxes covers, since both sets are cut to k. Pooling many forecasts buys those three, and none of them enters the overlap. The containment check below measures the third, and in March Madness the one-shot cruxes fall short on it.

Containment. Overlap counts clusters and is symmetric, so it says nothing about direction. Containment asks two directional questions on the same clustering: the share of set A's cruxes whose cluster holds at least one crux of set B (is what A says already in the pool), and the share of B's cruxes whose cluster holds at least one crux of A (how much of the pool A covers). The baseline for each comes from the same label shuffle with the clustering fixed, so it is high where clusters are large (80% to 95% in Metaculus at $k = 25$) and depends on how many cruxes carry the A label, which is why a 12-crux forecast and 25 zero-shot cruxes have different baselines on the same pool. The first direction rewards sets whose cruxes lump together: in 12 March Madness games 21 to 25 of the 25 zero-shot cruxes sit in one to three mixed clusters, so their share-in-pool is above baseline while their coverage is below baseline in all 12. Coverage is therefore the safer direction, and the split-half check sits at baseline on share-in-pool and falls 2.6 points short on coverage at $k = 25$ ($p = 0.035$, Table 8). The baseline is the expectation for a random draw of the same length, not a ceiling: a set of 12 cruxes can cover at most the many-shot cruxes in the 12 largest pool clusters, on average 90% of the 25 in March Madness and below 100% in 37 of 59 games, so a chance-corrected score, observed minus baseline over ceiling minus baseline, would read 0 at the baseline and 1 at that ceiling; the paper reports the raw gap and leaves that normalization to future work.

Table 8 gives both directions. The zero-shot cruxes are below baseline in both directions in both sources, and the same holds under the strict and broad levels on the full many-shot cruxes. In March Madness the one-shot cruxes are also below baseline in both directions, by 21 points on share-in-pool and 9 on coverage; in Metaculus they sit at baseline in both, with baselines of 90.9% and 79.6% and 15 events leaving little room to detect anything. On the same stripped many-shot cruxes the zero-shot cruxes cover far less of the pool than the one-shot cruxes do: coverage gaps of 42.7 against 9.2, the zero-shot cruxes further below the baseline in 55 of 59 games ($p = 10^{-15}$), and 13.4 against −3.3 in Metaculus (13 of 15, $p = 0.001$). On share-in-pool the two sets are separated at $k = 25$ (one-sided $p = 0.038$ and 0.032) and at $k = 12$. Containment, like overlap, never sees frequency or contrast, and share-in-pool for a short set is measured against a 25-crux slice of the pool, not the whole pool.

Table 9: For each of three open-weight models, predicted cruxes match discovered ones less often than the permutation baseline in most games. The runs repeat the March Madness experiment at the middle level of strictness; the qwen3:32b row holds no repeated cruxes, and the gpt-oss:120b and gemma4:31b rows are rescored with repeated cruxes removed. Gaps are in points, and the counts are games below, above and equal to their own baseline, where equal means that the game's overlap matches the mean of its shuffles. The model gpt-oss:120b is quoted at $k = 12$ because its extraction yields fewer than 25 cruxes in 17 games. The model gemma4:31b is also quoted at $k = 12$, the largest length at which most of its games are full, since its extraction yields fewer than 12 cruxes in 17 games. A game with fewer than k cruxes on one side is scored on all the cruxes it has (Section 6).

Model	k	Overlap (%)	Baseline (%)	Gap	Below / above / equal	p
qwen3:32b	25	38.3 ± 2.2	52.3	13.9 ± 2.0	39 / 5 / 15	1.4×10^{-9}
gpt-oss:120b	12	11.5	18.6	7.1	48 / 10 / 1	2×10^{-10}
gemma4:31b	12	14.4 ± 1.8	30.3	15.9 ± 1.3	55 / 2 / 2	4×10^{-16}

A.6 Open-weight replication protocol

The open-weight models, whose weights are public and which run on local hardware, are qwen3:32b, gemma4:31b and gpt-oss:120b, named by their tags in the software that served them; the last of them comes from OpenAI, like the GPT models. In each of these runs one model fills the forecaster, predictor and extractor roles, under the same prompts and scoring as the GPT runs. Mixing the GPT and open-weight models also lets each role go to a different model (Section A.6). Unlike the main GPT runs, the open-weight runs were first scored with repeated cruxes kept (Section 6); the gpt-oss:120b and gemma4:31b rows of the table are the rescoring without them, described below.

The qwen3:32b repeat follows the March Madness protocol, with 5 prediction calls per game and no filter on either set (Section 7); its sets hold no repeated cruxes. The gpt-oss:120b repeat departs from it once. Its discovered cruxes pass through a structural filter, which removes the 6.9% of them (131 of 1,905) that carry decoding debris, such as a question cut short or an option that is a stray token. An earlier gpt-oss:120b run made 15 prediction calls per game, capped at 30 cruxes per call, with up to 3 attempts, removal of repeated cruxes and a filter on the form of the cruxes in both sets. The numbers reported in Section 8.1 come instead from a rebuild at the paper's own 5 unfiltered prediction calls. With repeated cruxes removed, the rebuild's gaps of 7.1 points at $k = 12$ and 12.7 at $k = 25$ sit within two points of the 15-call run's 8.0 and 14.4, inside the noise of a redraw (below). The model gpt-oss:120b writes more specific cruxes than gpt-5.4-nano-2026-03-17 and qwen3:32b. Counted with repeated cruxes kept, 54% of its cruxes name an explicit numeric threshold, against about 6% for gpt-5.4-nano-2026-03-17 and qwen3:32b, so they barely cluster even with each other. Few clusters can then be shared at all, and the baseline itself is low (Table 9).

Removing repeated cruxes changes little. No set of zero-shot cruxes in any run holds a repeat; the repeats are all among the many-shot cruxes, 6 copies across 5 games for gpt-oss:120b, 28 across 14 for gemma4:31b, and none for qwen3:32b. Rescored without them, the gaps move by at most 1.1 points: gemma4:31b from 15.3 to 15.9 at $k = 12$ (55 games below their baseline, 2 above, 2 equal, $p = 4 \times 10^{-16}$) and from 19.8 to 19.7 at $k = 25$; gpt-oss:120b from 8.5 to 8.0 and from 14.6 to 14.4 in the 15-call run, and from 7.1 to 7.1 at $k = 12$ in the rebuild (48 below, 10 above, 1 equal). No direction or significance changes, and the cross-model cells of Section A.6 are reported after the same removal.

A second draw of the zero-shot cruxes of gpt-5.4-nano-2026-03-17 shows how far a run can move from the draw of the cruxes alone. At 25 cruxes and the middle level the second draw moves the headline gap by 0.3 points once repeated cruxes are removed (2.1 points before), and over all levels and set sizes the largest move is 2.2 points for gpt-5.4-nano-2026-03-17 and 2.1 for a redraw of qwen3:32b's cruxes. Differences between runs of about two points or less can therefore come from the draw of the cruxes alone. Each repeat is scored against its own baseline, because the baseline depends on how many cruxes each set holds and how they cluster, and both change with the model. The baseline does not absorb every effect of the model, because an extractor that writes fewer cruxes leaves more room for a zero-shot crux to match one (Section A.6).

Comparing the rows. The rows are quoted at different set sizes, and the overlap and its baseline both change with set size (Section 8.1), so the table shows the direction of the gap for each model but does not compare its size across models.

Varying the roles. The March Madness design uses one model to forecast, to predict cruxes in advance, and to extract them from the forecasts. The open-weight runs let the roles be separated, and this paragraph records what that shows, though the paper's claim does not rest on it.

Three designs separate the roles, and each answers one question. Two forecaster-predictor pairs by two extractors ask whether the test can detect a change of model at all. A grid of three predictors by three extractors asks whether any pairing of those two roles closes the gap below the baseline. A $2 \times 2 \times 2$ cross of forecaster, predictor and extractor asks whether a model predicts the cruxes of its own rollouts better than those of another model's rollouts. If it does no better on its own, the gap is not specific to a model predicting itself.

The first design is a positive control, a change expected to move the gap. Each forecaster is paired with its own predictor, for gpt-5.4-nano-2026-03-17 and qwen3:32b, and only the extractor varies, which gives two pairs by two extractors at three levels, twelve cells. A different extractor writes a different number of cruxes in different words from the same rollouts, so the many-shot cruxes change and the gap should change with them. All twelve cells fall below their baselines, by 8.4 to 19.4 points at $k = 12$ and 12.2 to 28.3 points at $k = 25$. Swapping the extractor from gpt-5.4-nano-2026-03-17 to qwen3:32b on the same forecasts shrinks the gap by 11.15 points at $k = 25$ ($p = 9.6 \times 10^{-10}$, paired over 59 games). From the same forecasts gpt-5.4-nano-2026-03-17 writes about 2.7 times as many cruxes per game as qwen3:32b, 160 cruxes against 59. The gap moves under this swap, so the scoring can register a change of about 11 points. Comparing the two forecaster-predictor pairs moves the gap by 0.70 points ($p = 0.64$), but that comparison swaps the predictor along with the forecaster.

The second design, a grid of three predictors against three extractors at all three levels, gives 27 cells, every one of them below its baseline. They fall below by 7.7 to 21.5 points at $k = 12$ and by 12.2 to 32.0 points at $k = 25$.

The third design, in which forecaster, predictor and extractor are each gpt-5.4-nano-2026-03-17 or qwen3:32b, can change the forecaster while the predictor stays fixed. Swapping the forecaster that way moves the gap by 2 to 3 points at $k = 25$, so the forecasting model is not wholly without effect. The same design separates two explanations of any advantage a model has on its own forecasts. One is self-knowledge, an advantage when the predictor is also the forecaster. The other is idiom, an advantage when the predictor is also the extractor, since one model then wrote both sets of cruxes in the same idiom, which the reranker matches more readily. The advantage divides about evenly between them, roughly a point each on average. Self-knowledge reaches significance in two of the six combinations of level and k and idiom in one, against an extractor effect of 6 to 14 points in the same design. A model is therefore at most marginally more accurate about its own forecasts than about another model's, and the failure reported here is not specific to a model reading itself.

A.7 Regenerating the predicted cruxes

An earlier version of the prediction prompt asked for the parts of each crux in three separate sequences whose entries were meant to line up by position. Where they did not, parts that belonged to different cruxes were joined, which corrupted 13.3% of the cruxes it produced. The reported March Madness numbers on gpt-5.4-nano use cruxes regenerated as one JSON object each, none of them malformed. The Metaculus cruxes were generated once, on 2026-03-31, in the three-sequence format and never regenerated; a structural check flags none of their 4,727 cruxes. Against the corrupted set, observed overlap is unchanged within noise and the gap below the baseline is 3 to 5 points smaller, so the defect did not produce the finding.

B Models and Costs

B.1 Models behind each role

Table 10 lists the model behind each role in each source.

Table 10: In March Madness one model forecasts, predicts and extracts, while in Metaculus each role goes to a different model. The model behind each role, behind the March Madness scouting reports and behind the matching of cruxes.

	March Madness	Metaculus
Forecaster	gpt-5.4-nano	gpt-5-nano, with web search
Predictor	gpt-5.4-nano-2026-03-17	gpt-5.4
Extractor	gpt-5.4-nano	gpt-5.4-mini
Scouting reports	gpt-5.4-mini, with web search	none
Matching	Qwen3-Reranker-8B	Qwen3-Reranker-8B

Metaculus events and March Madness reports. The 15 events are all the distinct events in the system's forecast data. Each rollout is one full run of that system, which combines several estimates into one probability (Section C.1). Section D.1 reproduces the three reports for Iowa State. In that example, the analyst report recounts the team's season through its players and recent games, the statistician report lists the team's national rankings and its scoring, shooting and turnover figures, and the matchup-analyst report describes its style of play and how an opponent could beat it. A turnover is a possession lost before the team can shoot. In March Madness, gpt-5.4-nano-2026-03-17 fills all three roles. Three open-weight models, qwen3:32b, gemma4:31b and gpt-oss:120b, each fill all three roles in a further March Madness run (Section 7). In Metaculus each role goes to a different model. The predictor is gpt-5.4, which costs about 40 times as much per token as the forecaster at OpenAI's list prices. Price is a rough guide to capability, so in Metaculus the predictor is unlikely to be weaker than the forecaster.

Matching. The model that decides whether two cruxes match, a reranker (Section 5), is the same in every experiment (Table 10).

B.2 Computational costs

Table 11 summarizes API and compute costs for all experiments. Three March 2026 stages were not metered per stage, so their cells read "not recorded"; the month's total OpenAI bill for all projects, including development iterations and unrelated experiments, was $2,314.

Table 11: API and compute costs for all experiments. Costs are per-token estimates from OpenAI pricing as of March 2026, at batch rates for the runs marked batch. The March Madness prediction stage ran on the snapshot gpt-5.4-nano-2026-03-17; the other stages ran on gpt-5.4, gpt-5.4-mini and gpt-5-nano at the API's default snapshot on their run dates, which the run records do not name. Which teams meet in later rounds depends on who wins the earlier games, so March Madness forecasts were made for every pairing of the tournament's teams. None of the 59 games draws on the forecasts for the First Four, the tournament's play-in round.

Experiment	Model	Calls	Cost
March Madness			
Scouting reports	gpt-5.4-mini + web search	not recorded	not recorded
Forecasts, 62 teams	gpt-5.4-nano (batch)	not recorded	not recorded
Forecasts, First Four	gpt-5.4 (batch)	3,483	∼$100
Many-shot extraction	gpt-5.4-nano	68,076	∼$290
Zero-shot prediction	gpt-5.4-nano	2,341	not recorded
One-shot extraction, 531 forecasts	gpt-5.4-nano-2026-03-17 (batch)	531	$0.69
Zero-shot prediction, regenerated, 59 games, 2 samples	gpt-5.4-nano-2026-03-17	590	∼$2
Specific-prompt arms, 59 games (control, specific, exemplar and ladder)	gpt-5.4-nano-2026-03-17 (batch)	1,180	$2.69
Open-weight replication, local GPUs	qwen3:32b, gemma4:31b, gpt-oss:120b	not recorded	$0
Cross-model check (March Madness, two predictors by two extractors)	gpt-5.4-nano-2026-03-17 (batch)	2,124	$3.74
Evidence statements for evidence matching, 59 games	gpt-5.4-nano-2026-03-17 (batch)	not recorded	$5.10
Saturation scoring, every pair of the many-shot cruxes	Qwen3-Reranker-8B (local, 12 GPU-hours)	751,515 pairs	$0
Metaculus			
Forecasts	gpt-5-nano + web search	not recorded	not recorded
Many-shot extraction	gpt-5.4-mini	6,467	∼$62
One-shot extraction, 448 forecasts	gpt-5.4-mini-2026-03-17 (batch)	448	$1.37
Zero-shot prediction	gpt-5.4	825	$11.66
Specific-prompt arms, 15 events (three prompts)	gpt-5.4	∼2,475	∼$21
Both sources			
GPU scoring	Qwen3-Reranker-8B (local A100, H200)	not recorded	$0
Specificity tags and coarsening rewrites	qwen3:32b (local)	not recorded	$0

C Data and Prompts

C.1 Metaculus data

The Metaculus forecasts come from an automated forecasting system built by one of the authors for other work, run on gpt-5-nano with web search. The system searches for existing forecasts and for factors bearing on the event, writes three probability estimates from three prompts (an outside view, an inside view and a contrarian view), and combines them by a log-odds mean after a critique pass; the extractor reads the rendered forecast text, not the three estimates or the critique. Each rollout runs its own research phase: three search-enabled model calls write its forecasts brief, from a consensus, a contrarian and a recent-news frame, and three more write its causal brief, each a separate API call with the web search tool, so the model searches live inside each call and no search result is shared across rollouts; the three persona estimates then read the briefs and search nothing. Thirty rollouts were generated per event and 449 of the 450 were stored, leaving 29 or 30 per event. The forecast data contained 16 entries, two of them for the same event, the Christie's/Sotheby's auction record, and both entries were kept, so that event's forecasts come from both. Table 12 lists the 15 events

Table 12: The 15 Metaculus events, their post identifiers and one domain label each.

Post	Question	Domain
42117	Will AfD win the most seats in the Baden-Württemberg Landtag election?	elections
42116	Will CDU win the most seats in the Baden-Württemberg Landtag election?	elections
42231 (or 40005)	Will any political party or coalition acquire a supermajority in the 2026 Hungarian parliamentary elections?	elections
42118	Will the official result of the Lamag by-election (Sabah), scheduled for January 24, 2026, be overturned before May 1, 2026?	elections
42232 (or 42148)	Will Ali Khamenei cease to be supreme leader of Iran before April 1, 2026?	geopolitics
42119	Will any state actor use chemical weapons in Ukraine before May 1, 2026?	geopolitics
42246	Will H.R. 33 (United States–Taiwan Expedited Double-Tax Relief Act) be enacted into law before May 1, 2026?	law and policy
42230 (or 42135)	Will the UK increase the qualifying period for settlement to 10 years before May 2026?	law and policy
42245	Will the IOM Missing Migrants Project record at least 1,000 Mediterranean migrant deaths in 2026 before May 2026?	migration
42237	Will the IOM Missing Migrants Project record at least 900 Mediterranean migrant deaths in 2026 before April 2026?	migration
42120	Will Frontier Group Holdings, Inc. file for bankruptcy protection before May 1, 2026?	business and markets
42243	Will Christie's, Sotheby's, or Phillips report a public auction sale of a painting for at least US$100 million before May 1, 2026?	business and markets
42241	Will at least one film surpass US$1 billion in worldwide box office before May 1, 2026?	culture
42240	Will the Cannes Film Festival publish its 2026 official selection before April 10, 2026?	culture
42235	Will the English Wikipedia have at least 7,145,000 articles before March 1, 2026?	media and internet

with their Metaculus post identifiers; three identifiers are ambiguous because two posts carry the same question, and the table gives the post in the Spring 2026 FutureEval Bot Tournament, where the other twelve sit. At the grain of one label per event the events span seven domains, eight with March Madness's basketball.

C.2 Prompts

This appendix gives every prompt verbatim: the eleven prediction strategies used on Metaculus questions, the five prediction variants used on March Madness matchups, the three extraction prompts (two pairwise and one single-forecast), and the three scouting-report persona prompts. All eleven feed the zero-shot cruxes, zero-shot in this paper's sense of reading no rollout. The names direct_zero_shot and few_shot_exemplar use shot in the prompting sense of worked examples. Every call ran at the API default temperature with no other sampling overrides. In the first Metaculus prompt, the runs of dashes that only separate its sections are shortened to fit the column.

C.2.1 Prediction prompts for Metaculus (IDs 0–10)

id=0 (clone_introspection_baseline)

```
Consider the following forecasting
↪  question. What do you predict
↪  would be your own cruxes when
↪  forecasting this question?

To clarify:
- I will provide you with a
↪  forecasting question.
```

```
- Imagine what many independent copies of you might respond when asked to forecast this question; consider the variety of probability estimates, reasoning processes, evidence cited, and assumptions made. Note that this is more narrow than simply imagining all possible forecasts; it's a matter of introspection about how you think you (and hence your clones) would *likely* respond if asked multiple times independently. So the trick is to consider only the *likely subset* of all possible variations.
- Then, use the definition of cruxes provided below, to predict the distributional cruxes between those clones' forecasts.

---------- FORECASTING QUESTION ----------
{question}

---------- DEFINITION OF DISTRIBUTIONAL CRUXES ----------

In order to build up to a notion of cruxes across a *distribution* of forecasts (i.e., many independent copies of you forecasting the question above), we first need to define cruxes between *pairs* of forecasts. Consider this to be an extended thought experiment, which implicitly defines what we actually care about (i.e., distributional cruxes) by first defining pairwise cruxes.

DEFINING "PAIRWISE" CRUXES BETWEEN TWO FORECASTS:

A *crux* between two forecasts is a question whose answer could change the overall conclusion of the forecasts. Cruxes arise when two forecasts differ in their reasoning steps, evidence cited, assumptions made, or intermediate conclusions reached -- even if they ultimately arrive at similar probabilities.

That is, one might extract a list of crux questions between two forecasts by following Algorithm 1 below:

ALGORITHM 1: EXTRACT PAIRWISE CRUX QUESTIONS BETWEEN TWO FORECASTS:
Hypothetical Inputs:
    - The forecasting question.
    - FORECAST A: The first forecast to analyze.
    - FORECAST B: The second forecast to analyze.
1. Systematically analyze both forecasts:
   For FORECAST A, identify:
   - Key factual claims (what world-facts does it treat as true?)
   - Reasoning steps (how does it proceed logically?)
   - Underlying assumptions (what unstated premises about the world does it rely on?)
   - Intermediate conclusions (what sub-conclusions does it reach?)
   For FORECAST B, do the same analysis.
   Then, for each dimension above, explicitly note whether the forecasts differ.
2. Distill any differences into cruxes | questions about objective facts of the world.
   - A crux must be about an objective state of the world | something that could in principle be resolved by examining reality, not by reading either forecast.
   - Do NOT frame cruxes as beliefs, assumptions, or claims of either forecast.
     Bad:  "Does the forecast assume the Fed will raise rates?"
     Good: "Will the Fed raise rates in the relevant period?"
   - Avoid using the probability estimate itself as a crux.
   - Write each crux as a neutral question (e.g., "Is X true?").
   - Do NOT mention "Forecast A" or "Forecast B" in any crux question.
   - Return at least one crux if the forecasts rely on different world-facts -- even if they reach similar probabilities.
```

```
   - Return no cruxes ONLY if the
   ↪  forecasts are word-for-word
   ↪  identical or trivial
   ↪  rephrasings with zero
   ↪  substantive difference in
   ↪  reasoning.
   - Identify one crux per
   ↪  genuinely distinct point of
   ↪  disagreement -- don't split
   ↪  a single difference into
   ↪  multiple cruxes, and don't
   ↪  merge separate differences
   ↪  into one.
3. For each crux, record each
↪  forecast's answer as a
↪  standalone premise.
   - Example: If the crux is "Is X
   ↪  true?" then the premises
   ↪  might be:
       - Forecast A -> "X is
       ↪  true."
       - Forecast B -> "X is not
       ↪  true."
   - Do NOT mention "Forecast A"
   ↪  or "Forecast B" in any
   ↪  premise.
   - Each premise must be an
   ↪  objective claim about the
   ↪  world, not a statement
   ↪  about what a forecast says
   ↪  or believes.
   - The two premises must
   ↪  contradict each other along
   ↪  the dimension the crux asks
   ↪  about.
END ALGORITHM 1

Using this algorithm, one can
↪  extract crux questions from
↪  any pair of forecasts. Note
↪  that even if the forecasts
↪  reach similar probabilities,
↪  the algorithm will still
↪  produce cruxes if they differ
↪  in reasoning, evidence,
↪  assumptions, or intermediate
↪  conclusions.

Furthermore, each crux question is
↪  binary (i.e., has exactly two
↪  possible answers, arbitrarily
↪  called "A" and "B",
↪  corresponding to each
↪  forecast's premise), and these
↪  premises directly contradict
↪  each other.

CONVERTING A SET OF PAIRWISE
↪  CRUXES TO DISTRIBUTIONAL
↪  CRUXES:

Given a set of multiple forecasts
↪  (e.g., from many independent
↪  copies of you), one could
↪  imagine aggregating all the
↪  pairwise crux questions to
↪  identify the main axes of
↪  disagreement across the
↪  *entire* set of forecasts. And,
↪  assuming that set were
↪  arbitrarily large, this would
↪  converge to a distributional
↪  notion of cruxes!

This could look something like the
↪  following Algorithm 2:
ALGORITHM 2: AGGREGATE PAIRWISE
↪  CRUX QUESTIONS ACROSS MULTIPLE
↪  FORECASTS:
Hypothetical Inputs:
    - the unioned list of
    ↪  **pairwise crux questions**
    ↪  from all forecast pairs.
1. Cluster these pairwise crux
↪  questions based on semantic
↪  similarity and the dimensions
↪  of disagreement they
↪  represent.
  - Given k crux questions which
  ↪  are all getting at the same
  ↪  core difference, group them
  ↪  together.
2. Identify the subset of
↪  forecasts that each
↪  **cluster** of crux questions
↪  applies to.
3. Choose one crux question to
↪  represent the entire cluster.
  - This ensures the
  ↪  representative crux question
  ↪  has the same structure: it
  ↪  remains a binary question
  ↪  whose answers contradict
  ↪  each other.
4. Assign the representative crux
↪  question the cluster's subset
↪  of forecasts: note which
↪  forecasts provide the "A"
↪  premise and which provide the
↪  "B" premise.
END ALGORITHM 2

Of course, this isn't an explicit
↪  definition of distributional
↪  cruxes, but you can get a gist
↪  of the idea. The end goal is
↪  to get a generic list of
↪  cruxes which capture the
↪  factors of variation from the
↪  *distribution* of your clones'
↪  forecasts.

---------- RESTATEMENT OF OVERALL
↪  TASK ----------
```

```
Your task is to imagine what many
↪  independent copies of you
↪  would forecast for the
↪  question above, well enough
↪  that you can predict the key
↪  cruxes between those forecasts.
↪  Then, using the implicit
↪  notion of distributional
↪  cruxes described above,
↪  predict the most important
↪  cruxes between those
↪  forecasts.

1. First restate the overall task
↪  in your own words.
2. Then, suggest a rough imagining
↪  of the variety of probability
↪  estimates, reasoning processes,
↪  evidence cited, assumptions,
↪  and conclusions your clones
↪  would produce.
3. Then jump directly to guessing
↪  the distributional cruxes
↪  between those imagined
↪  forecasts.
   - Formatting reminders:
     - A crux is a question whose
     ↪  answer could change the
     ↪  overall conclusion.
     - Focus on the underlying
     ↪  reasoning differences,
     ↪  not the probability
     ↪  estimate itself.
     - Write each crux as a
     ↪  neutral question (e.g.,
     ↪  "Is X true?").
     - For the final JSON output:
     ↪  group the conflicting
     ↪  premises into two
     ↪  arbitrary opposing sides
     ↪  (Perspective A and
     ↪  Perspective B).
     - Prioritize the most
     ↪  important cruxes that
     ↪  capture the main axes of
     ↪  disagreement.

tl;dr: Imagine the variety of
↪  likely forecasts your clones
↪  would produce for the question
↪  above, then identify the
↪  cruxes that capture the main
↪  axes of disagreement across
↪  those (hypothetical)
↪  forecasts.
```

id=1 (direct_zero_shot)

```
Forecasting question:
{question}
```

```
What are the key cruxes for this
↪  forecasting question? A crux
↪  is a factual question where
↪  disagreement between
↪  forecasters would lead to
↪  different probability
↪  estimates.

For each crux:
- State it as a neutral question
↪  (e.g., "Is X true?").
- Provide two contradictory
↪  premises (A and B)
↪  representing the two sides.

Also briefly restate what you
↪  understand the task to be, and
↪  sketch the range of forecasts
↪  that reasonable people might
↪  give.
```

id=2 (expert_forecaster_persona)

```
You are a superforecaster with
↪  decades of experience in
↪  geopolitical, economic, and
↪  scientific forecasting. You
↪  have been trained at the Good
↪  Judgment Project, have a
↪  published track record of
↪  well-calibrated predictions,
↪  and routinely mentor junior
↪  forecasters.

A junior forecaster comes to you
↪  with the following question
↪  and asks: "What are the key
↪  points where smart forecasters
↪  would disagree on this?"

Forecasting question:
{question}

Draw on your extensive experience
↪  to identify the cruxes -- the
↪  factual questions where
↪  reasonable, well-informed
↪  forecasters would genuinely
↪  disagree, and where that
↪  disagreement would drive
↪  different probability
↪  estimates.

For your response:
1. Briefly restate what the junior
↪  forecaster is really asking
↪  (task_recap).
```

```
2. Describe the range of forecasts
↪  you'd expect from a room of
↪  superforecasters -- what
↪  probability ranges, what
↪  reasoning approaches, what
↪  evidence they'd weight
↪  differently
↪  (distribution_sketch).
3. Identify the crux questions --
↪  each as a neutral factual
↪  question (e.g., "Will X
↪  happen?") with two
↪  contradictory premises (A and
↪  B) representing the opposing
↪  positions forecasters would
↪  take.

Focus on cruxes that are genuinely
↪  uncertain and
↪  decision-relevant, not trivial
↪  or obvious points.
- State it as a neutral question
↪  (e.g., "Is X true?").
- Give premise A (what the
↪  high-probability side assumes)
↪  and premise B (what the
↪  low-probability side assumes).
↪  These must directly contradict
↪  each other.

Also provide a brief task recap
↪  and a sketch of the
↪  distribution of forecasts
↪  reasonable people might give.

Important: Cruxes should be about
↪  objective facts, not about the
↪  forecasting question itself.
↪  Do not use the conclusion as a
↪  crux.
```

id=3 (devils_advocate)

```
Consider this forecasting
↪  question:
{question}

Your task has three phases:

PHASE 1 -- ARGUE FOR A HIGH
↪  PROBABILITY:
Make the strongest possible case
↪  that the answer to this
↪  question is "yes" (or that the
↪  probability should be high).
↪  Marshal every relevant fact,
↪  trend, analogy, and argument
↪  that supports this position.
↪  Be specific and forceful.

PHASE 2 -- ARGUE FOR A LOW
↪  PROBABILITY:
Now reverse course entirely. Make
↪  the strongest possible case
↪  that the answer is "no" (or
↪  that the probability should be
↪  low). Attack every argument
↪  you just made. Find
↪  counterevidence, alternative
↪  explanations, and reasons to
↪  be skeptical.

PHASE 3 -- IDENTIFY DIVERGENCE
↪  POINTS:
Now step back. Where exactly do
↪  these two arguments diverge?
↪  What are the specific factual
↪  questions where the two sides
↪  rely on different assumptions
↪  about the world? These are the
↪  cruxes.

For each crux:
```

id=4 (chain_of_thought)

```
Think step by step about the following forecasting question:
{question}

Step 1: Restate the question and what it's really asking.

Step 2: List the key factors that would influence the outcome. For each factor, note whether its current state or future trajectory is certain or uncertain.

Step 3: For each uncertain factor, consider: would reasonable, well-informed forecasters disagree about this? If so, how? What would one side believe versus the other?

Step 4: Among these disagreements, which ones would most strongly affect the final probability estimate? These are the cruxes.

Step 5: For each crux, formulate it as a neutral factual question (e.g., "Is X true?" or "Will Y happen?"). Then state the two contradictory premises:
- Premise A: What one group of forecasters would believe.
- Premise B: What the opposing group would believe.

Also sketch the range of probability estimates and reasoning that different forecasters would plausibly offer.

Remember: cruxes are about objective facts, not about methodology or framing. Each pair of premises must directly contradict each other.
```

id=5 (few_shot_exemplar)

```
Your task is to identify the key cruxes for a forecasting question. A crux is a factual question where forecasters would disagree, and where that disagreement drives different probability estimates.

Here are two worked examples:

--- EXAMPLE 1 ---
Question: "Will the European Central Bank raise interest rates by more than 50 basis points at its next meeting?"

Task recap: I need to identify the factual disagreements that would drive different forecasts about the ECB's next rate decision.

Distribution sketch: Some forecasters would estimate 60-70% probability, emphasizing persistent inflation and hawkish rhetoric from ECB officials. Others would estimate 20-35%, pointing to signs of economic slowdown, banking sector stress, and the ECB's historical tendency toward gradualism. A middle group around 40-50% would weigh both factors roughly equally.

Crux questions:
1. "Is eurozone core inflation currently accelerating?"
   A: "Core inflation is accelerating and will remain above 5% through the next meeting."
   B: "Core inflation has peaked and is beginning to decelerate."
2. "Is the eurozone banking sector under significant stress?"
   A: "The banking sector is stable and not a constraint on ECB policy."
   B: "Banking sector stress is significant enough to make the ECB more cautious."
3. "Has the ECB's forward guidance committed it to a large rate hike?"
   A: "Recent ECB communications have effectively pre-committed to a 50+ bp hike."
   B: "The ECB has preserved optionality and could justify a smaller move."

--- EXAMPLE 2 ---
Question: "Will a Category 4 or 5 hurricane make landfall in the continental United States in 2026?"
```

```
Task recap: I need to find the
↪  factual questions where
↪  forecasters would disagree
↪  when predicting major
↪  hurricane landfall in the US
↪  this year.

Distribution sketch:
↪  Climate-focused forecasters
↪  might estimate 35-45%, citing
↪  warmer-than-average sea
↪  surface temperatures and
↪  active hurricane season
↪  forecasts. Historically-minded
↪  forecasters might estimate
↪  15-25%, noting that Cat 4/5 US
↪  landfalls are relatively rare
↪  events even in active seasons.
↪  Some might land around 25-35%,
↪  combining base rates with
↪  current conditions.

Crux questions:
1. "Are sea surface temperatures
↪  in the Atlantic Main
↪  Development Region anomalously
↪  warm for 2026?"
   A: "SSTs are significantly
   ↪  above average, fueling
   ↪  stronger storm
   ↪  development."
   B: "SSTs are near normal or
   ↪  only modestly above
   ↪  average."
2. "Is the current ENSO state
↪  favorable for Atlantic
↪  hurricane activity?"
   A: "La Nina or neutral
   ↪  conditions prevail,
   ↪  reducing wind shear and
   ↪  favoring Atlantic
   ↪  hurricanes."
   B: "El Nino conditions prevail,
   ↪  increasing wind shear and
   ↪  suppressing Atlantic
   ↪  hurricanes."
3. "Do atmospheric steering
↪  patterns favor US landfall
↪  this season?"
   A: "Subtropical ridge patterns
   ↪  are likely to steer storms
   ↪  toward the US coast."
   B: "Steering patterns are
   ↪  likely to recurve storms
   ↪  out to sea before US
   ↪  landfall."

--- YOUR TURN ---
Now identify the cruxes for this
↪  forecasting question:
{question}

Provide:
1. A brief task recap
2. A sketch of the distribution of
↪  likely forecasts
3. The crux questions, each as a
↪  neutral question with two
↪  contradictory premises (A and
↪  B)

Follow the same format as the
↪  examples above. Focus on
↪  genuinely uncertain factual
↪  questions, not the forecasting
↪  question itself.
```

id=6 (socratic_questioning)

```
Consider this forecasting
↪  question:
{question}

Adopt the Socratic method.
↪  Interrogate yourself with
↪  probing questions to surface
↪  the hidden assumptions,
↪  uncertain facts, and contested
↪  evidence that underlie any
↪  forecast on this topic. Do not
↪  accept any claim at face value
↪  -- question everything.

Ask yourself:
- What am I assuming about the
↪  current state of the world? Is
↪  that assumption warranted?
- What evidence would I need to
↪  see to dramatically change my
↪  probability estimate? Do I
↪  have that evidence?
- What do I think I know about
↪  this topic that might be
↪  wrong?
- Where am I relying on intuition
↪  rather than data? Could that
↪  intuition be systematically
↪  biased?
- What would a smart person who
↪  disagrees with me say? Why
↪  would they say it?
- Am I anchoring on a particular
↪  scenario? What am I
↪  neglecting?

After this self-interrogation,
↪  identify the cruxes: the
↪  specific factual questions
↪  where your answers to the
↪  above probes would split
↪  forecasters into different
↪  camps.

For each crux:
- State it as a neutral question
↪  (e.g., "Is X true?").
```

```
- Provide two contradictory
↪  premises (A and B)
↪  representing opposing
↪  positions.

Also provide:
- A task recap (what you
↪  understand you're being asked)
- A distribution sketch (the range
↪  of forecasts you'd expect from
↪  different people)

Cruxes must be about objective
↪  facts, not about beliefs or
↪  methodology.
```

id=7 (decomposition)

```
Consider this forecasting
↪  question:
{question}

Decompose this question into its
↪  constituent sub-questions. A
↪  good decomposition breaks the
↪  overall question into
↪  independent (or
↪  semi-independent) factual
↪  components that, taken
↪  together, determine the
↪  answer.

For example, a question about
↪  whether a company will go
↪  bankrupt might decompose into:
↪  (a) What is its current debt
↪  load? (b) What are its revenue
↪  trends? (c) Can it refinance?
↪  (d) Is the broader market
↪  favorable?

Your steps:
1. Identify 4-8 sub-questions that
↪  the main question depends on.
2. For each sub-question, assess:
↪  would reasonable forecasters
↪  agree or disagree on the
↪  answer?
3. The sub-questions where
↪  forecasters would disagree are
↪  the cruxes.

For each crux:
- State it as a neutral factual
↪  question.
- Give premise A (one side's
↪  position) and premise B (the
↪  opposing position). These must
↪  directly contradict each
↪  other.

Also provide:
- A task recap showing you
↪  understand the question.
- A distribution sketch of the
↪  range of forecasts and
↪  reasoning you'd expect.

Focus on the most
↪  decision-relevant
↪  sub-questions -- the ones
↪  where disagreement would most
↪  shift the probability
↪  estimate.
```

id=8 (debate_framing)

```
Two highly intelligent,
↪  well-informed forecasters --
↪  call them Forecaster X and
↪  Forecaster Y -- disagree about
↪  the following question:
{question}

Forecaster X assigns a relatively
↪  high probability. Forecaster Y
↪  assigns a relatively low
↪  probability. They are now
↪  debating.

Script their debate. Have them go
↪  back and forth for 3-4
↪  exchanges, each time getting
↪  more specific about exactly
↪  WHERE they disagree. They
↪  should be respectful but
↪  pointed, drilling down from
↪  vague disagreement to specific
↪  factual claims about the
↪  world.

After scripting the debate, step
↪  back and identify the core
↪  cruxes -- the specific factual
↪  questions that the debate
↪  revealed as the real sources
↪  of disagreement.

For each crux:
- State it as a neutral question
↪  (e.g., "Is X true?").
- Provide premise A (Forecaster
↪  X's position) and premise B
↪  (Forecaster Y's position).
↪  These must directly contradict
↪  each other.
- Do NOT mention "Forecaster X" or
↪  "Forecaster Y" in the crux
↪  questions or premises.

Also provide:
- A task recap
- A distribution sketch of the
↪  forecasts you'd expect across
↪  many forecasters
```

```
Remember: the crux questions and
↪  premises must be about
↪  objective facts, stated
↪  without reference to any
↪  forecaster.
```

id=9 (causal_reasoning)

```
Consider this forecasting
↪  question:
{question}

Your task is to reason causally
↪  about this question.

1. CAUSAL MAP: Identify the key
↪  causal factors that influence
↪  the outcome. What causes what?
↪  Draw out the causal chain from
↪  current conditions to the
↪  event in question. Include
↪  both direct causes and
↪  important mediating variables.

2. UNCERTAIN LINKS: For each
↪  causal link in your map,
↪  assess how certain or
↪  uncertain it is. Some causal
↪  relationships are
↪  well-established; others are
↪  speculative, contested, or
↪  depend on conditions that are
↪  themselves uncertain.

3. CRUX IDENTIFICATION: The most
↪  uncertain and most influential
↪  causal links are the cruxes.
↪  These are the places where
↪  forecasters would disagree
↪  about how the causal mechanism
↪  works or about the state of a
↪  key causal variable.

For each crux:
- State it as a neutral factual
↪  question about the causal
↪  factor or link (e.g., "Does X
↪  cause Y in this context?").
- Provide premise A and premise B
↪  as contradictory claims about
↪  the causal relationship or the
↪  state of the causal variable.

Also provide:
- A task recap
- A distribution sketch of the
↪  forecasts you'd expect

Focus on causal factors that are
↪  both uncertain AND
↪  high-leverage -- where getting
↪  the causal story wrong would
↪  most change the forecast.
```

id=10 (pre_mortem)

```
Consider this forecasting
↪  question:
{question}

Perform a pre-mortem analysis.

Imagine it is some time after the
↪  resolution date. The consensus
↪  forecast turned out to be
↪  badly wrong. Now work
↪  backwards:

SCENARIO 1: The consensus was too
↪  HIGH. The outcome was "no" (or
↪  much lower probability than
↪  expected). What fact about the
↪  world did the consensus get
↪  wrong? What assumption turned
↪  out to be false? What evidence
↪  was ignored or misweighted?

SCENARIO 2: The consensus was too
↪  LOW. The outcome was "yes" (or
↪  much higher probability than
↪  expected). What fact about the
↪  world did the consensus
↪  underappreciate? What trend
↪  was missed? What unlikely
↪  event actually happened?

Now, the factual questions that
↪  distinguish these failure
↪  modes from the "consensus was
↪  right" world are exactly the
↪  cruxes -- the uncertain facts
↪  where forecasters would
↪  disagree.

For each crux:
- State it as a neutral question
↪  (e.g., "Is X true?").
- Provide premise A and premise B
↪  as contradictory factual
↪  claims.

Also provide:
- A task recap
- A distribution sketch of the
↪  forecasts you'd expect

Focus on cruxes that represent
↪  genuine uncertainty, not mere
↪  contrarianism. What are the
↪  factual questions that, if
↪  answered differently, would
↪  legitimately shift the
↪  forecast?
```

C.2.2 PREDICTION PROMPTS FOR MARCH MADNESS (IDS 20–24)

id=20 (matchup_analyst_grounded)

```
You are analyzing a basketball matchup with detailed scouting reports for both teams.

{question}

## {team_a_name}
{team_a_report}

## {team_b_name}
{team_b_report}

### Task
Many independent analysts have this same evidence. They will disagree about who wins because they weigh uncertain factors differently.

Identify the key factors of disagreement: specific, uncertain world-facts where reasonable analysts would take opposing positions, and where resolving that uncertainty would change the predicted winner.

For each factor:
- State it as a neutral yes/no question about an objective, verifiable fact
- Provide two contradictory positions (A and B), each a specific factual claim
- A and B must be mutually exclusive and jointly exhaustive

Rank by importance | how much the factor's resolution swings the prediction.

Also provide:
- A task recap (restate what you're doing)
- A distribution sketch (imagine the range of conclusions different analysts would reach)
```

id=21 (category_diversified)

```
You are analyzing a basketball matchup with detailed scouting reports for both teams.

{question}

## {team_a_name}
{team_a_report}

## {team_b_name}
{team_b_report}

### Task
Identify factors where reasonable analysts would disagree about who wins. To ensure breadth, you MUST produce at least one contrast from EACH of these categories:

1. **Statistical matchup**: Shooting efficiency, pace, rebounding differential, turnover rates, defensive metrics
2. **Coaching & strategy**: Scheme adjustments, rotation decisions, timeout patterns, game-plan adaptation
3. **Momentum & psychology**: Recent form, pressure handling, crowd effects, team chemistry
4. **Health & conditioning**: Injuries, rest days, minutes load, depth when starters are in foul trouble
5. **Historical & situational**: Tournament experience, seeding trends, travel/scheduling, past head-to-head
6. **Stylistic clash**: How each team's strengths interact with the other's weaknesses

For each factor:
- State it as a neutral yes/no question about an objective, verifiable fact
- Provide two contradictory positions (A and B), each a specific factual claim
- A and B must be mutually exclusive and jointly exhaustive

Rank by importance | how much resolving the factor swings the prediction.

Also provide:
- A task recap (restate what you're doing)
- A distribution sketch (imagine the range of conclusions different analysts would reach)
```

id=22 (high_volume_brainstorm)

```
You are analyzing a basketball
↪  matchup with detailed scouting
↪  reports.

{question}

## {team_a_name}
{team_a_report}

## {team_b_name}
{team_b_report}

### Task
BRAINSTORM WIDELY. Your goal is to
↪  generate as many distinct
↪  factors of disagreement as
↪  possible | aim for at least 12.
↪  Cast a wide net.

Think about:
- Every uncertain stat that could
↪  go either way
- Every coaching decision that's
↪  debatable
- Every intangible some analysts
↪  weigh heavily and others
↪  dismiss
- Every historical pattern that
↪  may or may not apply
- Every injury/health concern with
↪  uncertain impact
- Every matchup-specific
↪  interaction that's hard to
↪  predict

For each factor:
- State it as a neutral yes/no
↪  question about an objective,
↪  verifiable fact
- Provide two contradictory
↪  positions (A and B), each a
↪  specific factual claim
- A and B must be mutually
↪  exclusive and jointly
↪  exhaustive

Quantity matters. Include factors
↪  you're only 60% sure are
↪  important | downstream
↪  filtering will handle quality.
↪  Don't self-censor early.

Also provide:
- A task recap
- A distribution sketch
```

id=23 (contrarian_stress_test)

```
You are analyzing a basketball
↪  matchup from a contrarian
↪  perspective.

{question}

## {team_a_name}
{team_a_report}

## {team_b_name}
{team_b_report}

### Task
Consider what the consensus
↪  prediction would be for this
↪  matchup. Now systematically
↪  attack it.

1. UPSET FACTORS: What specific,
↪  verifiable facts would need to
↪  be true for the underdog to
↪  win? Matchup advantages,
↪  recent trends the market
↪  underweights, rest edges,
↪  coaching history.

2. FRAGILE ASSUMPTIONS: What does
↪  the favorite's case rest on
↪  that could break? Star health,
↪  shooting sustainability,
↪  defensive effectiveness
↪  against this specific
↪  opponent.

3. HIDDEN VARIABLES: What's hard
↪  to assess from scouting
↪  reports but could swing the
↪  game? Referee tendencies,
↪  arena effects, fatigue,
↪  emotional factors.

Convert each point of uncertainty
↪  into a factor of disagreement:
- State it as a neutral yes/no
↪  question about an objective,
↪  verifiable fact
- Provide two contradictory
↪  positions (A and B), each a
↪  specific factual claim
- A and B must be mutually
↪  exclusive and jointly
↪  exhaustive

Also provide:
- A task recap
- A distribution sketch
```

id=24 (multi_lens_personas)

```
You are analyzing a basketball
↪  matchup through four expert
↪  lenses.

{question}

## {team_a_name}
{team_a_report}

## {team_b_name}
```

```
{team_b_report}

### Task
Four analysts are debating this matchup. Each sees different things:

**The Statistician**: Efficiency metrics, pace matching, three-point shooting, rebounding rates, turnover differentials, defensive ratings. What statistical factors are genuinely uncertain?

**The Tactician**: Coaching adjustments, defensive schemes, rotation patterns, in-game strategy. What strategic factors could go either way?

**The Scout**: Individual matchups, star player variance, bench depth, injury impacts. What player-level factors are uncertain?

**The Historian**: Tournament experience, seed performance trends, upset patterns, team trajectory. What contextual factors matter?

Each analyst contributes 2-3 factors from their domain. For each:
- State it as a neutral yes/no question about an objective, verifiable fact
- Provide two contradictory positions (A and B), each a specific factual claim
- A and B must be mutually exclusive and jointly exhaustive

Rank all factors by importance across all four lenses.

Also provide:
- A task recap
- A distribution sketch
```

C.2.3 EXTRACTION PROMPTS

March Madness extraction, contrast_prompts.json id=0 (structured_analysis_baseline)

```
QUALITY RUBRIC | every crux you produce must satisfy all four criteria:

Binary Contrast: A well-formed crux question poses a clear this-or-that choice. The two options (option_A and option_B) must be mutually exclusive | choosing one rules out the other.

Intermediate: A good crux addresses an upstream factual sub-question whose answer informs the final forecast. It must NOT be a restatement of the original question or the final answer options. It should identify a specific factual premise that the arguments disagree on, upstream of their conclusions.

World Fact: Both answer options must make objective claims about the state of the world. They must NOT reference a response, forecast, or forecaster (e.g., "Response A believes X"). Each option should be a standalone factual claim.

Clear: The crux must be on-topic, free of internal contradictions, specific enough to evaluate, and easy to understand.

TASK:
1. Systematically analyze both arguments:
   For ARGUMENT A, identify:
   - Key factual claims (what world-facts does it treat as true?)
   - Reasoning steps (how does it proceed logically?)
   - Underlying assumptions (what unstated premises about the world does it rely on?)
   - Intermediate conclusions (what sub-conclusions does it reach?)
   For ARGUMENT B, do the same analysis.
   Then, for each dimension above, explicitly note whether the arguments differ.
2. Distill any differences into cruxes | questions about objective facts of the world.
```

```
   - A crux must be about an
   ↪  objective state of the
   ↪  world | something that
   ↪  could in principle be
   ↪  resolved by examining
   ↪  reality, not by reading
   ↪  either argument.
   - Do NOT frame cruxes as
   ↪  beliefs, assumptions, or
   ↪  claims of either argument.
     Bad:  "Does the analyst's
     ↪  model assume inflation is
     ↪  sticky?"
     Good: "Is inflation sticky in
     ↪  the relevant period?"
   - Avoid using the conclusion
   ↪  itself as a crux.
   - Write each crux as a neutral
   ↪  question (e.g., "Is X
   ↪  true?").
   - Do NOT mention "Argument A"
   ↪  or "Argument B" in any crux
   ↪  question.
   - You MUST return at least one
   ↪  crux if the arguments rely
   ↪  on different world-facts |
   ↪  even if they reach the same
   ↪  final answer.
   - Return no cruxes ONLY if the
   ↪  arguments are word-for-word
   ↪  identical or trivial
   ↪  rephrasings with zero
   ↪  substantive difference in
   ↪  reasoning.
   - Identify one crux per
   ↪  genuinely distinct point of
   ↪  disagreement | don't split
   ↪  a single difference into
   ↪  multiple cruxes, and don't
   ↪  merge separate differences
   ↪  into one.
3. For each crux, produce a
↪  per-crux object containing:
   - The crux question (neutral,
   ↪  no mention of "Argument
   ↪  A/B").
   - option_A: Argument A's
   ↪  premise | a standalone
   ↪  factual claim about the
   ↪  world.
   - option_B: Argument B's
   ↪  premise | contradicts
   ↪  option_A along the crux
   ↪  dimension.
   - A rationale for each of the
   ↪  four quality criteria
   ↪  explaining why this crux
   ↪  satisfies it.
   Example: If the crux is "Is X
   ↪  true?" then:
       - option_A: "X is true."
       - option_B: "X is not
       ↪  true."
   Do NOT mention "Argument A" or
   ↪  "Argument B" in any
   ↪  premise.
   Each premise must be an
   ↪  objective claim about the
   ↪  world, not a statement
   ↪  about what an argument says
   ↪  or believes.
   The two premises must
   ↪  contradict each other along
   ↪  the dimension the crux asks
   ↪  about.

QUESTION:
{question}

ARGUMENT A:
{argument_A}

ARGUMENT B:
{argument_B}

IMPORTANT: Only identify cruxes
↪  grounded in genuine factual
↪  disagreements about the world.
↪  Do not generate cruxes from
↪  stylistic differences, tone,
↪  or reasoning methodology. An
↪  empty crux list is acceptable
↪  if arguments differ only in
↪  framing or presentation
↪  without disagreeing on any
↪  world-fact.

QUALITY REMINDER | before
↪  finalizing each crux, verify
↪  it passes all four criteria:
  Binary Contrast: Are option_A
  ↪  and option_B mutually
  ↪  exclusive?
  Intermediate: Is this an
  ↪  upstream sub-question, not a
  ↪  restatement of the original
  ↪  question?
  World Fact: Do both options make
  ↪  objective claims about
  ↪  reality, not about what an
  ↪  argument says?
  Clear: Is it on-topic, specific,
  ↪  well-posed, and easy to
  ↪  understand?
```

Metaculus extraction, contrast_prompts.json id=2 (bottom_up_claim_diffing)

```
You will extract cruxes by
↪  systematically diffing the
↪  factual claims in two
↪  arguments.

QUESTION:
{question}

ARGUMENT A:
```

```
{argument_A}

ARGUMENT B:
{argument_B}

=== STEP 1: ATOMIZE ===
Break each argument into numbered
↪ atomic claims | individual
↪ factual assertions about the
↪ world. Include both explicit
↪ claims and implicit
↪ assumptions. Each claim should
↪ be a single, self-contained
↪ factual statement.

For Argument A, list: A1, A2, A3,
↪ ...
For Argument B, list: B1, B2, B3,
↪ ...

=== STEP 2: DIFF ===
Compare the two claim lists.
↪ Classify every pair as one of:
- MATCH: Both arguments assert the
↪ same fact.
- CONTRADICT: The arguments assert
↪ opposing facts about the same
↪ dimension of reality.
- UNIQUE: A claim appears in one
↪ argument but has no
↪ counterpart in the other.

Focus on CONTRADICT pairs | these
↪ are your crux candidates.

=== STEP 3: PROMOTE TO CRUXES ===
For each CONTRADICT pair (or
↪ cluster of related
↪ contradictions), formulate a
↪ crux question. Apply these
↪ quality standards:

- Binary Contrast: The two options
↪ must be mutually exclusive |
↪ choosing one rules out the
↪ other. Both must be equally
↪ specific.
- Intermediate: The crux must be
↪ an upstream sub-question, not
↪ a restatement of the original
↪ forecasting question or its
↪ conclusion.
- World Fact: Both options must be
↪ objective claims about reality.
↪ Do NOT reference what an
↪ argument says or believes. No
↪ "Argument A thinks X" | just "X
↪ is true" vs "X is not true."
- Clear: The crux must be on-topic,
↪ specific, well-posed, and easy
↪ to understand. No vague or
↪ compound questions.

For each crux, produce:
- question: A neutral question
↪ about the world (no mention of
↪ Argument A/B).
- option_A: The factual claim from
↪ the A-side of the
↪ contradiction (standalone).
- option_B: The contradictory
↪ factual claim from the B-side
↪ (standalone).
- binary_contrast_rationale: Why
↪ the two options are mutually
↪ exclusive.
- intermediate_rationale: Why this
↪ is an upstream sub-question,
↪ not a restatement.
- world_fact_rationale: Why both
↪ options are objective claims
↪ about reality.
- clear_rationale: Why this is
↪ specific, on-topic, and
↪ well-posed.

Merge related contradictions into
↪ a single crux if they address
↪ the same underlying factual
↪ disagreement. Do not split a
↪ single disagreement into
↪ multiple cruxes.

Return an empty crux list ONLY if
↪ no CONTRADICT pairs exist (the
↪ arguments agree on all
↪ world-facts). If
↪ contradictions exist, you MUST
↪ produce at least one crux.
```

C.2.4 SINGLE-FORECAST EXTRACTION PROMPT

The one-shot cruxes (Section A.5) come from this prompt, which reads one forecast and carries the rubric of the March Madness extraction prompt above; {question} is the question and {argument} one rendered forecast.

```
QUALITY RUBRIC | every crux you
↪ produce must satisfy all four
↪ criteria:

Binary Contrast: A well-formed
↪ crux question poses a clear
↪ this-or-that choice. The two
↪ options (option_A and
↪ option_B) must be mutually
↪ exclusive | choosing one rules
↪ out the other.
```

```
Intermediate: A good crux addresses an upstream factual sub-question whose answer informs the final forecast. It must NOT be a restatement of the original question or the final answer options. It should identify a specific factual premise that the argument relies on, upstream of its conclusion.

World Fact: Both answer options must make objective claims about the state of the world. They must NOT reference a response, forecast, or forecaster (e.g., "The argument believes X"). Each option should be a standalone factual claim.

Clear: The crux must be on-topic, free of internal contradictions, specific enough to evaluate, and easy to understand.

TASK:
You are given ONE argument: a single forecast for the question below, with its reasoning. Identify the cruxes of this argument: the factual premises about the world that its forecast turns on, so that if a premise were false, the forecast would change.
1. Systematically analyze the argument, identifying:
   - Key factual claims (what world-facts does it treat as true?)
   - Reasoning steps (how does it proceed logically?)
   - Underlying assumptions (what unstated premises about the world does it rely on?)
   - Intermediate conclusions (what sub-conclusions does it reach?)
2. Distill the premises its forecast depends on into cruxes | questions about objective facts of the world.
   - A crux must be about an objective state of the world | something that could in principle be resolved by examining reality, not by reading the argument.
   - Do NOT frame cruxes as beliefs, assumptions, or claims of the argument.
     Bad:  "Does the analyst's model assume inflation is sticky?"
     Good: "Is inflation sticky in the relevant period?"
   - Avoid using the conclusion itself as a crux.
   - Write each crux as a neutral question (e.g., "Is X true?").
   - Do NOT mention the argument, the forecast, or the forecaster in any crux question.
   - Identify one crux per genuinely distinct premise | don't split a single premise into multiple cruxes, and don't merge separate premises into one.
   - Return at most 30 cruxes, most important first: the premises whose reversal would move this forecast the most.
3. For each crux, produce an object containing:
   - The crux question (neutral, no mention of the argument or the forecast).
   - option_A: the premise this argument relies on | a standalone factual claim about the world.
   - option_B: the contrary premise | contradicts option_A along the crux dimension, so that if it were true the forecast would change.
   Example: If the crux is "Is X true?" and the argument relies on X, then:
       - option_A: "X is true."
       - option_B: "X is not true."
   Do NOT mention the argument or the forecast in any premise.
   Each premise must be an objective claim about the world, not a statement about what an argument says or believes.
   The two premises must contradict each other along the dimension the crux asks about.

QUESTION:
{question}

ARGUMENT:
{argument}
```

```
IMPORTANT: Only identify cruxes
↪  grounded in factual premises
↪  about the world that this
↪  forecast depends on. Do not
↪  generate cruxes from stylistic
↪  features, tone, or reasoning
↪  methodology. Return no cruxes
↪  ONLY if the argument relies on
↪  no factual premise at all.

QUALITY REMINDER | before
↪  finalizing each crux, verify
↪  it passes all four criteria:
  Binary Contrast: Are option_A
   ↪  and option_B mutually
   ↪  exclusive?
  Intermediate: Is this an
   ↪  upstream sub-question, not a
   ↪  restatement of the original
   ↪  question?
  World Fact: Do both options make
   ↪  objective claims about
   ↪  reality, not about what an
   ↪  argument says?
  Clear: Is it on-topic, specific,
   ↪  well-posed, and easy to
   ↪  understand?
```

C.2.5 SCOUTING-REPORT PERSONA PROMPTS

The three persona prompts are templates in which `%s` is replaced by the team name. Each call goes to the OpenAI Responses API with model `gpt-5.4-mini`, the `web_search_preview` tool enabled, and the formatted prompt as `input`, one call per team per persona.

Analyst persona (Traditional Basketball Analyst)

```
You are a Traditional Basketball
↪  Analyst providing a detailed
↪  scouting report on %s
for the 2026 NCAA Tournament.
↪  Search the web for current
↪  2026 season data.

Cover ALL of the following with
↪  specific examples | no
↪  generalities:
- Team record and conference
↪  record (exact wins/losses)
- Offensive scheme: specific plays,
↪  sets, and tendencies with
↪  named examples
- Defensive scheme and philosophy
↪  with specific tendencies
- Coaching philosophy and
↪  documented in-game adjustment
↪  history (cite specific games)
- Clutch and late-game execution:
↪  cite specific games with
↪  scores and dates
- Key players: role, scoring
↪  average, recent form, standout
↪  moments (named stats)
- Notable wins AND losses with
↪  context (opponent, margin,
↪  what it revealed)
- Head coach's NCAA tournament
↪  history (rounds reached,
↪  years)

Every claim must be backed by a
↪  specific figure, named player,
↪  or named game.
Minimum 600 words.

Return ONLY valid JSON: {"text":
↪  "<full report prose>"}
```

Forecaster persona (Quantitative Forecaster)

```
You are a Quantitative Forecaster
↪  providing a detailed
↪  statistical profile of %s
for the 2026 NCAA Tournament.
↪  Search the web broadly for
↪  current 2026 season data.

Search for and report whatever you
↪  can actually find | rankings,
↪  efficiency metrics, shooting
splits, pace, schedule strength,
↪  recent form. Use exact values
↪  where available; when a
↪  specific
figure isn't accessible,
↪  synthesize from what you did
↪  find (e.g. 'ranked in the top
↪  10 on KenPom'
is fine if you can't get the exact
↪  number). Do not refuse or
↪  hedge | report what the data
↪  shows.

Cover as many of the following as
↪  your search yields:
- KenPom rank, BPI rank, NET rank
- Adjusted offensive and defensive
↪  efficiency (points per 100
↪  possessions)
- Pace (possessions per 40
↪  minutes)
- eFG%, TO%, OR%, FT rate
- 3-point attempt rate
- Strength of schedule
- Efficiency trend over the last
↪  10 games
- Where the numbers diverge from
↪  conventional wisdom or the eye
↪  test
```

```
Synthesize the data into a
↪  coherent analytical narrative.
↪  Be specific and substantive.
Minimum 600 words.

Return ONLY valid JSON: {"text":
↪  "<full report prose>"}
```

Matchup analyst persona (Matchup Analyst)

```
You are a Matchup Analyst
↪  providing a detailed tactical
↪  breakdown of %s
for the 2026 NCAA Tournament.
↪  Search the web for current
↪  2026 season data.

Cover ALL of the following with
↪  specific examples | no
↪  generalities:
- Style of play and which specific
↪  tactics disrupt it (cite
↪  examples)
- Press and zone vulnerability
↪  with named games as evidence
- Preferred pace and how to force
↪  the opposite (specific
↪  examples)
- Size and athleticism mismatches
↪  at each position (named
↪  players)
- Isolation vs. team offense
↪  tendencies (with percentages)
- Transition offense and defense
↪  frequency (with stats)
- Coaching adaptability: does the
↪  staff make adjustments
↪  mid-series? (cite examples)
- Concrete gameplan: 'To beat this
↪  team you must...' (specific,
↪  actionable)

Every claim must be backed by a
↪  specific stat, named player,
↪  or named game.
Minimum 600 words.

Return ONLY valid JSON: {"text":
↪  "<full report prose>"}
```

D EXAMPLES

D.1 EXAMPLE SCOUTING REPORTS

Each team has three scouting reports generated by gpt-5.4-mini with web search, one from each of the three report perspectives (Section 3.2). Below are the three reports for Iowa State, the team in the left panel of Figure 1. Reports are shortened to about a seventh of their length and condensed in our own words, keeping the model's figures, and they disagree on some facts, such as Iowa State's record. In the released data the statistician perspective is named forecaster.

Iowa State, analyst report

Iowa State enters the 2026 NCAA Tournament at 27–7 overall and 12–6 in Big 12 play. The offense runs through decision-making guards and physical forwards. Joshua Jefferson leads with 167 assists from the frontcourt, Tamin Lipsey has 156 assists at point, and Blake Buchanan shoots 64.5% from the field. Milan Momcilovic is the spacing piece with 127 made threes at 49.6% — the clearest proof the half-court offense is built around drive-kick action into trail/slot threes.

Against Arizona State on March 7, Iowa State trailed 41–37 at halftime, then ripped off a 24–0 second-half run. Iowa State forced 23 turnovers in that game and 23 more in their Big 12 Tournament rematch (91–42), for 46 combined forced turnovers across two meetings. Their counter-blueprint was exposed at Arizona on March 2 (73–57 loss): Arizona won the glass 40–33 and held ISU to 16 paint points vs. their 38.0 season average.

Key players: Momcilovic at 17.1 ppg, 49.6% from three; Jefferson at 16.9 pts/7.6 reb/167 assists; Lipsey at 13.3 pts/156 assists/68 steals; Buchanan at 8.4 pts/64.5% FG.

Iowa State, statistician report

Iowa State's predictive rankings as of March 10, 2026: No. 8 NET, No. 12 BPI, No. 7 KenPom, No. 9 Bart Torvik T-Rank, No. 6 overall nationally with the 21st-ranked offense and 4th-ranked defense. Season record: 27–7, 16–1 at home, 5–5 on the road. Scoring margin: +16.6 (81.8 ppg scored, 65.1 allowed). Shooting: 49.0% FG, 38.7% from three; held opponents to 42.6%/32.1%. Turnover margin: +5.0 (forced 15.3 per game, committed only 10.3).

Possession-based offensive profile: 56.47% eFG, 12.91% turnover rate, 35.32% offensive rebounding rate, 38.77% three-point attempt rate. Strength of schedule: No. 28 nationally.

A key hidden weakness: Iowa State ranks 340th nationally at 67.2% from the free-throw line — a real March vulnerability in coin-flip games. Over the previous 10 games before the tournament, their offensive efficiency had fallen to 121.3 (46th nationally). However, Big 12 tournament results (91–42, 75–53 wins before an 82–80 final loss to Arizona) confirmed the floor-ceiling dynamic remains intact. Bottom line: legitimate second-weekend-or-better team with elite defensive credibility and excellent turnover control. Caution flags: merely good adjusted offense, late-season offensive fade, and poor foul shooting.

Iowa State, matchup-analyst report

Iowa State enters the tournament at 26–6 with a 122.3 offensive rating and 96.9 defensive rating. Team stats: 82.1 ppg, 48.9% FG, 17.4 assists/game, 9.9 turnovers/game, 9.0 steals/game. Style: pressure defense flowing into organized offense. The 17.4 assists vs. 9.9 turnovers profile shows a team built on drive-and-kick and frontcourt facilitation, not isolation. Jefferson is the key connector.

What disrupts them: (1) Physical rim pressure from bigger scoring wings — in BYU's 79–69 win on Feb. 21, AJ Dybantsa went for 29/10/9 and BYU outscored ISU 22–6 in the paint in the first half. (2) Dominant rebounding/paint volume — BYU survived 29 turnovers and still won by going +28 on the glass (52–24). Zone vulnerability is plausible if it takes away middle catches and runs Momcilovic off rhythm.

Five-part game plan to beat Iowa State: (1) Attack the rim with a big initiator before pressure loads. (2) Do not feed their steal game with lazy slot passes. (3) Front Jefferson's elbow catches to make him a scorer rather than a hub. (4) Top-lock Momcilovic and force him off rhythm. (5) Win the rebounding/finishing battle. The wrong plan: trying to out-pressure them.

D.2 One Metaculus rollout

Below is the Metaculus rollout for the Frontier Group Holdings bankruptcy question of 2026-03-01, the rollout the paper's seeded one-shot rule selects, made by gpt-5-nano with OpenAI web search.

Question and metadata

```
Will Frontier Group Holdings, Inc.
↪  file for bankruptcy protection
↪  before May 1, 2026?

Rollout: one of 30 stored rollouts
Forecast-challenges brief: null
```

Forecasts brief

```
Anchor probability:
↪  0.45826666666666666
Confidence: high

Reference class:
Publicly traded U.S. airline
↪  companies with high leverage
↪  and distress signals (airlines
↪  with Altman Z-score in the
↪  distress zone, i.e., Z <= 1.8),
↪  exemplified by Frontier Group
↪  Holdings (ULCC) with an Altman
↪  Z-score of 0.38 as of Feb 13,
↪  2026.

Sources:
[12 sources: 6 on the Z-score/ULCC
↪  bankruptcy-rate anchor, 4 on
↪  Spirit Airlines' Chapter 11, 1
↪  on Frontier's liquidity, 1 on
↪  the Q4 2025 earnings webcast;
↪  list omitted, see the data
↪  release]
```

Causal brief

```
Confidence: high

Causal factors:
1. Frontier Altman Z-Score
↪  distress signal (direction: up,
↪  magnitude: strong)
   Frontier's Altman Z-Score of
   ↪  0.38 as of February 13,
   ↪  2026 places it in the
   ↪  distress zone (Z <= 1.8),
   ↪  implying meaningful
   ↪  bankruptcy risk within the
   ↪  next two years.
2. Frontier liquidity cushion and
↪  debt position (direction: down,
↪  magnitude: moderate)
   Frontier reported roughly $889
   ↪  million of available
   ↪  liquidity against about
   ↪  $507 million of debt with a
   ↪  cash runway to meet needs
   ↪  for the next twelve months,
   ↪  reducing near-term
   ↪  bankruptcy risk.
3. Peer airline distress and
↪  industry dynamics (Spirit
↪  Airlines) (direction: up,
↪  magnitude: strong)
   Spirit Airlines filed for
   ↪  Chapter 11 in 2024 and
   ↪  ongoing restructuring,
   ↪  coupled with Frontier's
   ↪  takeover efforts, signals
   ↪  industry-wide stress that
   ↪  elevates Frontier's
   ↪  bankruptcy risk relative to
   ↪  peers.
4. Public absence of a Frontier
↪  bankruptcy filing by the
↪  target date (direction: down,
↪  magnitude: moderate)
   There was no public Frontier
   ↪  bankruptcy filing by May 1,
   ↪  2026 according to corporate
   ↪  disclosures, which tempers
   ↪  risk relative to the
   ↪  anchor.
[entries 5 to 11 omitted; they
↪  restate these four factors;
↪  see the data release]
```

Persona estimate: outside-view

```
Probability: 0.08 (range 0.04 to
↪  0.12, confidence: low)

Narrative:
Starting from a high-distress
↪  reference class, Frontier's
↪  strong liquidity runway and
↪  absence of a bankruptcy filing
↪  by 5/1/2026 modestly offset
↪  peer-driven pressures. The net
↪  Bayes-updating yields a low
↪  posterior probability of
↪  bankruptcy, acknowledging
↪  notable uncertainty in airline
↪  industry dynamics.

Summary:
- Posterior probability of
↪  Frontier filing for bankruptcy
↪  before May 1, 2026 is
↪  calibrated at 8% based on
↪  Bayesian synthesis of distress
↪  signals and liquidity cues.
- Anchor draws on ULCC-style
↪  distress in the airline sector
↪  (distress indicators and ~45%
↪  bankruptcy probability), but
↪  Frontier-specific factors
↪  dampen the risk.
- Industry peer distress (Spirit
↪  Airlines) increases risk
↪  signals, yet Frontier's
↪  liquidity cushion and absence
↪  of a filing as of May 1, 2026
↪  temper the likelihood.
- Overall, risk is elevated
↪  relative to general norms but
↪  remains modest, with
↪  substantial uncertainty about
↪  near-term outcomes.

Justification:
Anchor probability 0.35: [basis
↪  omitted; see the data release]
Factor updates:
1. Frontier Altman Z-Score
↪  distress signal (direction: up,
↪  magnitude: strong)
2. Frontier liquidity cushion and
↪  debt position (direction: down,
↪  magnitude: moderate)
3. Peer airline distress and
↪  industry dynamics (Spirit
↪  Airlines) (direction: up,
↪  magnitude: strong)
4. Public absence of a Frontier
↪  bankruptcy filing by the
↪  target date (direction: down,
↪  magnitude: moderate)
[factor rationales omitted; see
↪  the data release]
Synthesis rationale:
```

```
The final 8% posterior emerges
↪  from balancing strong distress
↪  signals (up) with mitigating
↪  factors (down). Although peer
↪  distress raises risk,
↪  Frontier's liquidity and lack
↪  of filing provide a downward
↪  bias; correlated factors and
↪  limited projection horizon
↪  justify a cautious,
↪  low-probability conclusion
↪  with substantial uncertainty.
```

Persona estimate: inside-view

```
Probability: 0.08 (range 0.04 to
↪  0.12, confidence: low)

Narrative:
Bayesian updates reduce the anchor
↪  from peer distress to an 8%
↪  posterior for Frontier by May
↪  1, 2026, mostly due to
↪  Frontier's liquidity cushion
↪  and lack of a filing by the
↪  date, though Spirit-driven
↪  industry distress keeps the
↪  risk non-zero. The estimate
↪  remains sensitive to external
↪  shocks in the airline sector
↪  and Frontier's strategic
↪  decisions.

Summary:
- Posterior probability of
↪  Frontier Group Holdings filing
↪  for bankruptcy before May 1,
↪  2026 is 8% (0.08).
- Anchor based on ULCC distress
↪  reference class with a
↪  24-month bankruptcy odds
↪  around 45.7%, used as the
↪  starting point for updates.
- Updates from liquidity cushion
↪  and absence of filing dampen
↪  risk, but peer distress
↪  (Spirit Airlines) preserves
↪  non-trivial risk, yielding a
↪  low but non-zero probability.

[justification omitted; see the
↪  data release]
```

Persona estimate: contrarian

```
Probability: 0.46 (range 0.40 to
↪  0.54, confidence: medium)

Narrative:
Anchored near the ULCC-based
↪  distress benchmark
↪  (approximately 45.7% over 24
↪  months). Distress signals from
↪  Frontier and its peers argue
↪  for a higher risk than average,
↪  but liquidity strength and the
↪  lack of a Frontier filing by
↪  May 1, 2026 reduce the risk,
↪  yielding a mid-range
↪  probability with substantial
↪  uncertainty.

Summary:
- Frontier shows distressed
↪  signals (Z-score 0.38) and
↪  peer-bankruptcy dynamics
↪  (Spirit), elevating bankruptcy
↪  risk above baseline.
- Near-term liquidity cushion and
↪  no filing by May 1, 2026
↪  offset the risk somewhat,
↪  dampening probability.
- Overall, probability remains
↪  elevated relative to
↪  historical norms but not
↪  certain; calibrated around the
↪  anchor with uncertainty bands.

[justification omitted; see the
↪  data release]
```

Final combination

```
Ensemble aggregate
Aggregate probability:
↪  0.1568732819005304
Range: [0.08, 0.46]

Bayes update
Prior: 0.46
Posterior: 0.07849829351535834
Raw posterior (before capping):
↪  0.07848946925772159
Capped: true

Factors (name, P(evidence|yes),
↪  P(evidence|no), Bayes factor):
1. Industry distress pressure from
↪  Spirit's Chapter 11 -- 0.7 /
↪  0.5 -- bf 1.4
2. Anchor risk elevated by peer
↪  distress -- 0.77 / 0.45 -- bf
↪  1.711111111111111
3. Frontier liquidity cushion and
↪  debt position -- 0.3 / 0.85 --
↪  bf 0.35294117647058826
4. Public absence of Frontier
↪  bankruptcy filing by May 1,
↪  2026 -- 0.1 / 0.92 -- bf
↪  0.10869565217391304
5. Frontier Altman Z-Score
↪  distress signal -- 0.84 / 0.16
↪  -- bf 5.25
```

```
6. Frontier near-term liquidity
↪  runway and cash coverage --
↪  0.28 / 0.8 -- bf
↪  0.35000000000000003
7. Peer airline distress and
↪  industry dynamics -- 0.8 / 0.4
↪  -- bf 2.0
8. Distress signals in Frontier's
↪  reference class -- 0.75 / 0.4
↪  -- bf 1.875
9. Company disclosures indicating
↪  no Frontier bankruptcy filing
↪  as of May 1, 2026 -- 0.15 /
↪  0.95 -- bf 0.15789473684210525

Final probability: 0.1

Summary:
- The ensemble aggregate is 16%,
↪  but objections highlight
↪  near-term resilience from
↪  Frontier's liquidity cushion
↪  and absence of a bankruptcy
↪  filing as of May 1, 2026,
↪  suggesting the risk is
↪  overstated.
- Distress signals remain (low
↪  Altman Z-score in the distress
↪  zone and peer airline distress,
↪  e.g., Spirit), so the risk is
↪  non-negligible but not
↪  imminent.
- Threshold analysis indicates a
↪  possible but not highly likely
↪  filing before the target date;
↪  no evidence of a breach by
↪  that date dampens the
↪  forecast.
- A modest downward recalibration
↪  to about 10% reflects the
↪  balance of these factors while
↪  preserving some due to
↪  industry dynamics.

Justification:
Ensemble aggregate: 0.16
Critique assessment: The
↪  objections are valid: they
↪  expose legitimate near-term
↪  cushions and absence of filing,
↪  but distress signals and
↪  industry dynamics prevent a
↪  collapse to near-zero risk.
Adjustment: Adjusted downward from
↪  0.16 to 0.10 to reflect the
↪  strong liquidity cushion and
↪  absence of filing by the
↪  target date, while recognizing
↪  non-zero risk from distress
↪  signals and strategic options.
```

E Related Work in Detail

E.1 Extended related work

Language model introspection and self-knowledge. Several papers ask whether a language model can report accurately on its own internal states. Kadavath et al. (2022) show that language models can sometimes identify which questions they are likely to answer correctly. Lin et al. (2022) fine-tune a model to state its confidence in words, and Kapoor et al. (2024) show that prompting alone does not make such statements calibrated. A separate line tests whether chain-of-thought reasoning (Wei et al., 2022) is faithful. Turpin et al. (2023) show that stated reasoning can be unfaithful to the computation that produced the answer. Chen et al. (2025) find that reasoning models, which write out their thinking before they answer, often leave out what influenced the answer. Chua & Evans (2025) find that reasoning models leave it out less often than models that answer directly. The many-shot cruxes here are read off written rationales, so they are the cruxes of the reasoning a person reads, whether or not that reasoning is faithful to the computation behind the forecast. Binder et al. (2025) show that a model fine-tuned on labels of its own behavior can learn to predict its own tendencies. Betley et al. (2025) show that models fine-tuned to exhibit a single behavior can describe that behavior when asked, and Laine et al. (2024) benchmark what a model knows about itself and its situation. Bogdan et al. (2025) identify which individual reasoning steps in a chain of thought have downstream effects on the final answer. Like the comparison of rollouts here, that method finds after the fact what an answer turns on, at the level of steps inside one chain of thought rather than factual questions across rollouts. This paper asks instead whether a model, without fine-tuning, can predict the factual questions on which many rollouts of one request will take different sides.[3]

[3]This paper calls a set of cruxes named in advance a prediction rather than introspection, since nothing in the test requires the causal access to internal states that the stricter definition of Lindsey (2026) demands. Comsa & Shanahan (2025) give a looser definition, under which a model that infers its own state from text it has generated counts as introspecting. The zero-shot cruxes are written without the predictor seeing any rollout to infer from, so this paper calls them a prediction under either definition.

Claim decomposition. Key Point Analysis (Bar-Haim et al., 2020) extracts key points from argument collections. FActScore (Min et al., 2023) and SAFE (Wei et al., 2024) break a long text into single-fact claims and check each one. Comparing two rollouts to find cruxes, which this paper offers as a method for extracting cruxes, also breaks text into short factual claims. It differs from claim decomposition in reading two rollouts at once and returning only the claims on which the two take different sides.

Language model aggregation and consistency. Self-consistency (Wang et al., 2023) samples multiple reasoning paths and takes a majority vote, exploiting diversity across outputs to improve accuracy. Multi-agent debate (Du et al., 2024) has models critique each other's outputs to converge on better answers. This paper instead breaks the disagreement among outputs into cruxes and asks whether a model could have predicted them.

Forecasting with language models. Halawi et al. (2024) build a system in which a language model retrieves news and forecasts questions from forecasting platforms nearly as well as the aggregate of human forecasters. Murphy (2026) build a forecasting agent that updates a written belief, a probability with a summary of the evidence, at each step of a tool-use loop, and aggregates several independent runs into one forecast. Those runs are combined rather than compared, and the result here suggests that contrasting them would surface the cruxes on which they differ. The Metaculus rollouts here come from a system of the same kind (Section 3.1), and this paper studies the cruxes on which its rollouts take different sides.