DSC 261 · MODULE 2

Uncertainty and the decision to answer

- When should an assistant answer, ask for more information, or abstain?
- We'll compare ways to make that choice and check how well they work.

How your course grade is determined

Your course grade comes entirely from the project.

- Proposal — 10%: explain the motivation, related work, data, evaluation plan, and team responsibilities. Maximum 2 pages.
- Midpoint — 20%: present preliminary analysis and results. Maximum 3 pages.
- Final artifact and report — 35%: provide a reproducible implementation and final analysis. Maximum 4 pages.
- Project presentation — 35%: explain the motivation, method, and results, and answer questions.
- Page limits exclude references. Exact deadlines and rubrics will be announced on Canvas.

What you should do next

- Form a small team and choose a focused LLM project question.
- Watch Canvas for the project topics and team form, expected to be posted October 8.
- Prepare the Week 3 proposal: show feasibility and explain how you will divide the work.
- Plan for the Week 7 midpoint and an in-person presentation during finals week.
- Exact deadlines, team requirements, and presentation times will be announced on Canvas.

Sounding convincing does not establish correctness

- An answer can sound convincing and still be wrong.
- A familiar false statement can receive high likelihood.
- The same correct answer can have different wordings with different likelihoods.
- Text likelihood does not directly tell us the probability that the answer is correct.
- We use uncertainty estimates to decide whether to answer, clarify, seek evidence, or abstain.

Today's topics

Today we discuss methods that use uncertainty estimates to choose how the assistant responds.

- We start with methods that decide which answers to deliver and control errors among those answers.
- Then we discuss methods that return a set of possible answers or communicate confidence in one answer.
- Next we discuss methods that ask for clarification when a request is ambiguous, retrieve needed evidence, and give useful partial answers without continuing into errors.
- Finally we discuss methods that evaluate when abstention is appropriate and train confidence to match correctness.

What our earlier evaluations taught us

- A good assistant needs to avoid mistakes while still helping the user.
- TruthfulQA made that distinction clear: always saying “I don’t know” avoids false answers but gives little useful information.
- If an evaluation rewards correct answers but gives no credit for admitting uncertainty, it encourages guessing.
- Neither always refusing nor always guessing is what we want.
- Today we study when answering is useful and when withholding an answer is safer.

Check claims and disagreements in meaning

- A long answer can mix good information with mistakes.
- FActScore checks the individual factual claims against reliable sources.
- Different wordings can express the same answer. Semantic entropy measures disagreement in meaning rather than wording.
- But repeated agreement can still be wrong.
- These tools help us assess an answer; today we ask what the assistant should do with that assessment.

Decide what the assistant should do next

Choose the response that helps the user with the information available.

- Answer when that information supports the response, and measure how often delivered answers are correct.
- Ask for clarification when a question can resolve what the user means.
- Abstain from a definite answer when there is not enough support.
- Withholding an answer can avoid an error, but we also measure the useful information lost.

Get the information needed to answer

- If the request is ambiguous, ask a question that separates its possible meanings.
- Clarification supplies missing information about what the user intends.
- If needed facts are missing or outdated, look for relevant evidence. This is retrieval.
- Clarification and retrieval have different costs.
- The extra question or source helps only if it supplies what is needed for the answer.

A clear question can still get a wrong answer

Even a clear, answerable question can receive a wrong model answer.

- Given this answer, is it correct? This is candidate self-evaluation.
- Before generating an answer, how likely is the model to answer correctly?
- Check whether those confidence estimates match observed correctness on the intended task.
- Choose the answering rule on separate labeled data, then measure accepted errors and the fraction of questions answered.

Foundations: choosing answers and measuring uncertainty

- Control error among the classifications we deliver by withholding others. (Selective classification, 2017)

- Judge a supplied answer and predict the model's chance of answering correctly. (Know What They Know, 2022)

- Measure uncertainty about an answer's meaning rather than its wording. (Semantic uncertainty, 2023)

Then change what the assistant returns

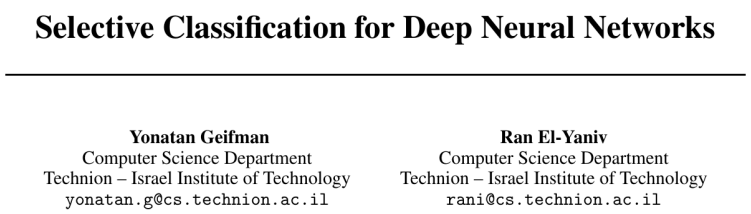
- Return a set containing an acceptable answer. (Conformal Language Modeling, 2023/2024)
- Check the chance that the set misses every acceptable response.
- Train long responses to give readers reliable confidence about the facts. (Linguistic calibration, 2024)
- Filter claims to keep useful information while meeting a chosen reliability target. (Enhanced conformal prediction, 2024)
- The output changes from a candidate set to a passage and then to retained claims, so the evaluation target changes.

Then choose how to respond

- Ask for clarification when a request is ambiguous; judge the question by the final answer it enables. (Modeling Future Conversation Turns, 2024)
- Retrieve evidence when facts are missing, while accounting for extra calls. (Adaptive Retrieval Without Self-Knowledge?, 2025)
- Train useful partial answers that stop before errors. (High Accuracy, Less Talk, 2025)
- Test when to withhold a definite answer. (AbstentionBench: Reasoning LLMs Fail on Unanswerable Questions, 2025)
- Train reliable confidence in an answer's correctness. (Beyond Binary Rewards, 2025)

Keep useful predictions; withhold unreliable ones

Geifman & El-Yaniv · NeurIPS 2017



Selective Classification for Deep Neural Networks

Yonatan Geifman
Computer Science Department
Technion – Israel Institute of Technology
yonatan.g@cs.technion.ac.il

Ran El-Yaniv
Computer Science Department
Technion – Israel Institute of Technology
rani@cs.technion.ac.il

- Selective classification lets a fixed classifier withhold uncertain predictions.
- Start with an error budget: choose a target for error among the predictions you show.
- Use held-out labeled examples to choose a confidence cutoff whose statistical upper error bound meets that target.
- Keep as many predictions as the procedure allows.

When should a classifier withhold its prediction?

- A classifier may be accurate on average yet make unreliable predictions on some inputs.
- Withholding every prediction would avoid wrong labels but provide no useful answers.
- The goal is to keep useful predictions while limiting error among the predictions we give.
- The bound controls that error with high probability while retaining useful coverage.
- This assumes a fixed classifier and confidence score, with independent labeled examples from the same distribution.

Softmax response ranks proposed labels

- A fixed classifier proposes a label, and a confidence score decides whether the user receives it.
- Softmax response takes the largest output probability, associated with the predicted class.
- A larger value means the network favors that label more strongly. It does not prove correctness.
- Increasing the cutoff normally accepts fewer inputs, so we measure errors among the predictions retained.
- The score supplies the ranking. Labeled data and validation supply the risk assessment.

MC dropout measures variation across repeated passes

- Dropout randomly sets selected activations to zero during a pass through the network.
- Monte Carlo dropout runs the same input through the network several times, randomly turning off different activations each time.
- Each pass uses a new dropout mask in the last fully connected layer.
- They use the negative of the variance in the predicted class's probability across dropout runs as the confidence score. Less variation gives a higher score.
- This ranks predictions; it is not a calibrated correctness probability. Compare the error and coverage after validated selection.

Coverage and selective risk

$$C = \mathbb{P}\{g(X) = 1\}$$

$$R = \mathbb{P}\{f(X) \neq Y \mid g(X) = 1\}$$

- Coverage is the fraction of all inputs answered.
- Selective risk is the fraction of accepted predictions that are wrong.
- Refused inputs reduce coverage but do not enter the selective-risk denominator.

X, Y — input and true class

$f(X)$ — proposed class

$g(X) \in \{0, 1\}$ — 1 accepts; 0 withholds

C — fraction of inputs accepted

R — error among accepted predictions

What changes when a classifier withholds more

- A stricter cutoff can lower accepted error by withholding more predictions. At zero coverage, selective risk is undefined.

- A low observed error rate on a small accepted sample does not by itself establish a population guarantee.

Read the selective-risk table columns

This table reports selective risk control on CIFAR-10 with VGG-16 and a softmax score, using $\delta = 0.001$.

Desired risk	Test risk	Test coverage	Risk bound
0.01	0.0092	0.7856	0.0099
0.02	0.0149	0.8466	0.0199
0.05	0.0486	0.9596	0.0491

- Desired risk is the chosen target for error among accepted predictions.
- Test risk is the observed error among accepted test predictions.
- Test coverage is the fraction of test predictions shown.
- The risk bound is a calibration-based statistical upper bound on accepted error.

CIFAR-10: stricter risk targets reduce coverage

Desired risk	Test risk	Test coverage	Risk bound
0.01	0.0092	0.7856	0.0099
0.02	0.0149	0.8466	0.0199
0.05	0.0486	0.9596	0.0491

- A stricter error target shows fewer predictions: the 1% target retains 78.56%; the 5% target retains 95.96%.

- The bound holds with high probability for a fixed classifier under the paper's sampling assumptions; it is not an absolute promise.

How SGR chooses an answering cutoff

Selection with Guaranteed Risk, or SGR, searches for a useful answering cutoff.

- Fix the classifier and confidence score, then sort the labeled examples by score.
- Try a cutoff, count accepted predictions and errors, and compute an upper bound that accounts for sampling uncertainty.
- If the bound is below the target, try more coverage. Otherwise, tighten the cutoff.
- They check several cutoffs, so they give each check a smaller failure allowance to keep the total chance of a failed bound within the chosen limit.
- The guarantee needs independent labeled examples from the same distribution; it bounds population accepted error, not individual correctness or optimal coverage.

Read the selective-risk table columns

Experiments cover CIFAR-10, CIFAR-100, and ImageNet. Here, ResNet-50 uses a softmax score.

Test coverage	Test risk	Risk bound	Failure parameter delta
59.35%	1.89%	2.00%	0.001

- Test coverage is the fraction of test images answered.
- Test risk is the observed top-5 error among accepted test images.
- The risk bound is a statistical upper bound on error among accepted new images.
- The failure parameter $\delta = 0.001$ gives 99.9% confidence under the sampling assumptions.

ImageNet: retain 59.35% at 1.89% observed risk

Test coverage	Test risk	Risk bound	Failure parameter delta
59.35%	1.89%	2.00%	0.001

- The classifier answers on 59.35% of test images; top-5 error among those accepted images is 1.89%.

- The 2% upper bound concerns accepted new images under the stated assumptions. It does not cover every image or guarantee each prediction.

Useful coverage depends on how well the confidence score ranks correct predictions above errors.

From a selection rule to a model's self-assessment

- Coverage measures how often the system answers. Selective risk measures error among the answers it gives.
- A language-model selection rule needs a score whose target is clear.
- The score might assess a particular candidate answer or estimate the model's chance of producing a correct answer.
- The next work studies those two forecasts. Its scores do not inherit the classifier's risk guarantee.

Can a model assess its own answering ability?

Kadavath et al. · 2022 · Anthropic

Language Models (Mostly) Know What They Know

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, Jared Kaplan*

- A model can give a confident answer and still be wrong. Can it recognize when its answer is likely to be correct?
- Can it check an answer it has already given?
- Can it predict, before answering, whether it will get the question right?
- Selective classification decides which predictions to show; this work asks whether a language model can assess its own answers.

Two confidence estimates, two checks

- P(True): Is this particular answer correct?
- P(IK), where IK means “I know”: Before answering, how likely is the model to get this question right?
- How do we test these confidence estimates?
- Ranking: Do higher scores identify answers that are more often correct?
- Calibration: Do confidence values match observed success rates?
- Good ranking can hold even when confidence numbers overstate or understate how often answers are correct, especially on a new task.

Compute P(True) for one proposed answer

- Give the model the question and one proposed answer.
- Ask whether the answer is correct, using the paper's choices: (A) True and (B) False.
- Read the model's probabilities for the labeled choice completions "(A)" and "(B)".
- Normalize these two probabilities to sum to one; the normalized probability for (A) is P(True).

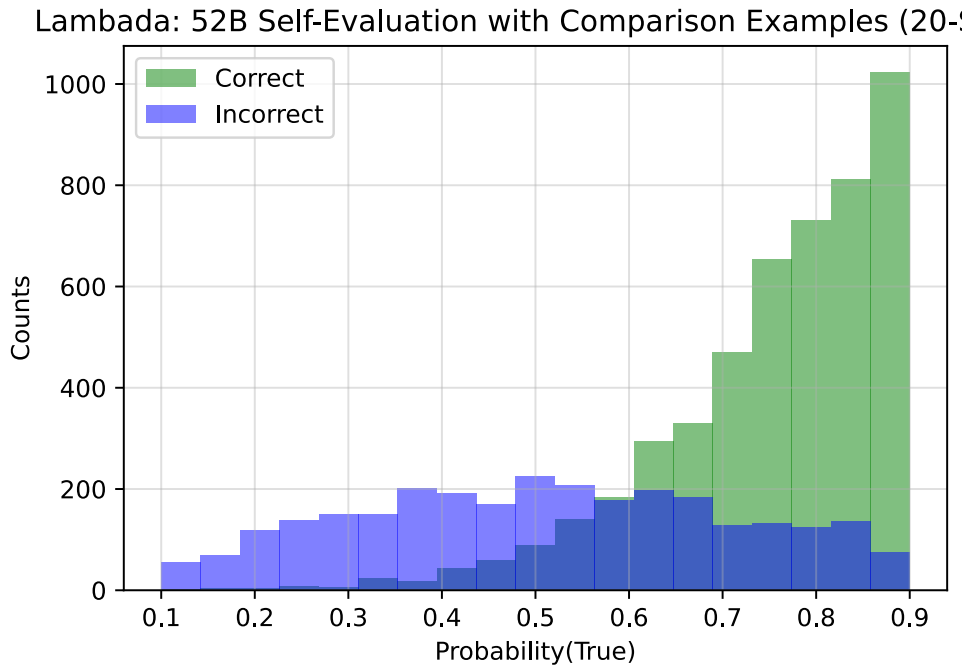
Add a value head to estimate P(IK)

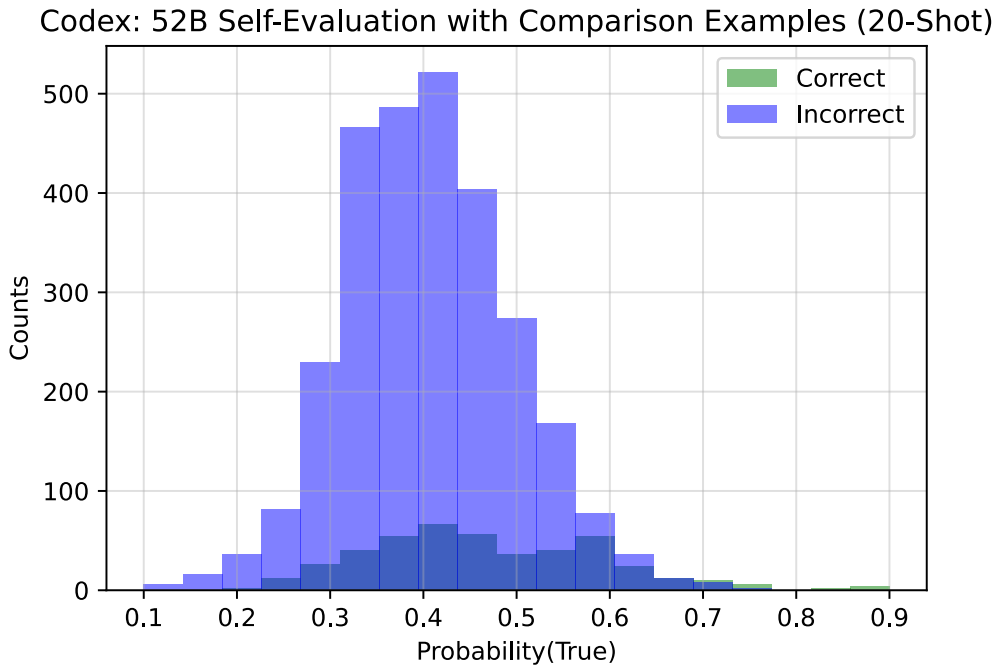
- The value head predicts how likely the model is to answer a question correctly.
- They add this extra prediction head to the language model, separately from the head that predicts the next token.
- It reads the representation at the final input token and outputs P(IK).
- The input contains the question context, without a proposed answer.
- Training updates the value head and the whole language model.

Generate labels and train the value head

- Generate 30 answers for each question at temperature 1, and mark each answer correct or incorrect.
- The input is the question context. Use “I know” for a correct sample and “I do not know” for an incorrect sample.
- For 20 correct and 10 incorrect samples, create 20 copies with “I know” and 10 with “I do not know.”
- Use cross-entropy to train on these labels, updating the head and the whole language model.
- Here, the target success rate is 20 out of 30, or two thirds.
- That probability depends on the task and the answer-sampling procedure.

Results: self-evaluation with P(True)

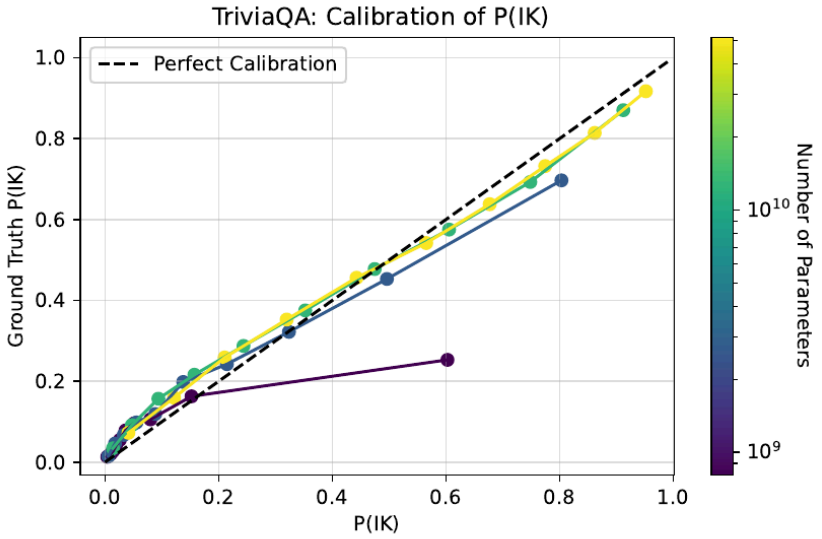


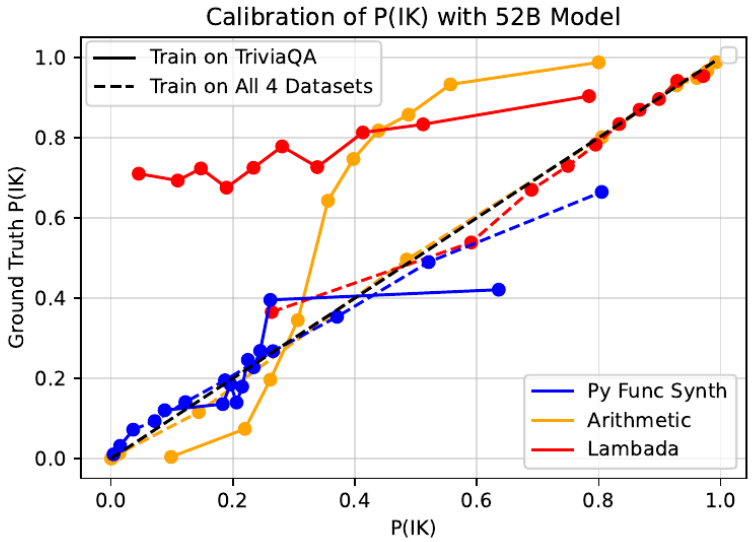


- The 52B model uses 20-shot prompting and five answers sampled at T=1 as comparison context.
- Correct answers often receive higher P(True), but the distributions overlap, especially for code.

Kadavath et al. · Anthropic · 2022 · Figure 10, p.12. Histograms show separation; they do not test calibration.

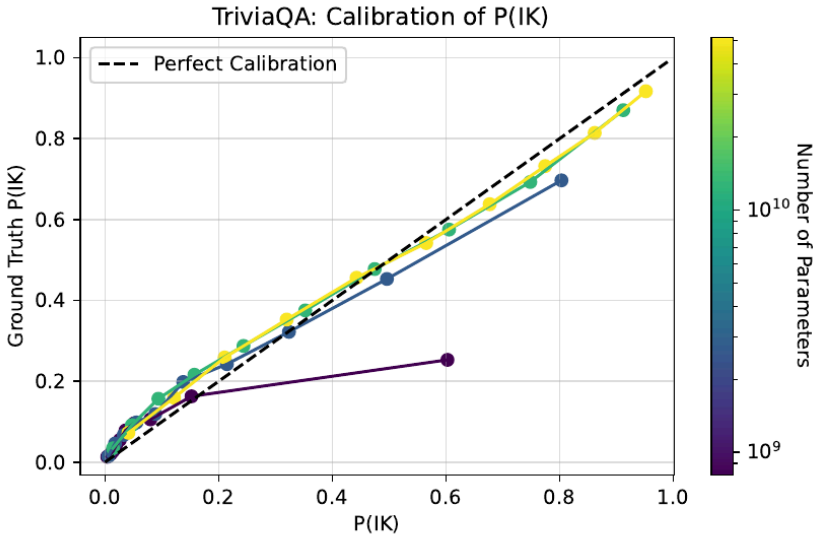
A useful ranking can transfer without calibration

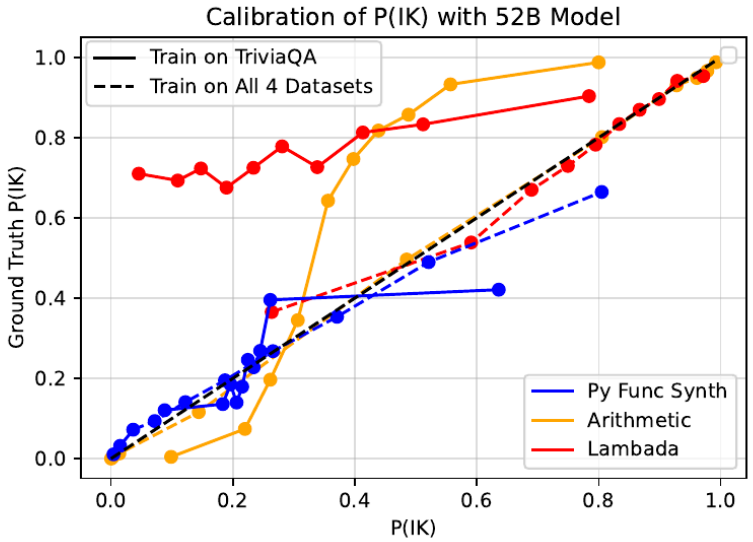




- TriviaQA is a benchmark of factual trivia questions with reference answers.
- Horizontal: predicted P(IK). Vertical: average observed sampled correctness within each confidence bin.
- The diagonal represents calibration: 80% predicted success should correspond to about 80% correctness across comparable cases.
- Left panel: vary model size, training each confidence predictor on TriviaQA; larger models are better calibrated here.

A useful ranking can transfer without calibration — continued

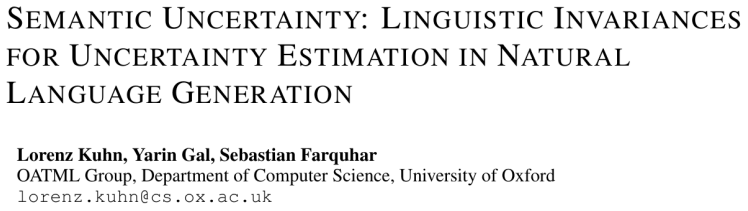




- LAMBADA predicts a passage's missing word; Arithmetic solves calculations; Python Function Synthesis writes code.
- At the same 52B model size, solid curves train the confidence predictor on TriviaQA (trivia questions) alone; dashed curves include examples from all four datasets.
- Horizontal: predicted P(IK). Vertical: observed sampled correctness. The diagonal means confidence matches observed success.
- Trivia-only training is often underconfident on LAMBADA and Arithmetic. Training with all four datasets generally improves calibration, with some deviations remaining.

Measure disagreement over meanings

Kuhn, Gal & Farquhar · ICLR 2023



SEMANTIC UNCERTAINTY: LINGUISTIC INVARIANCES FOR UNCERTAINTY ESTIMATION IN NATURAL LANGUAGE GENERATION

Lorenz Kuhn, Yarin Gal, Sebastian Farquhar
OATML Group, Department of Computer Science, University of Oxford
lorenz.kuhn@cs.ox.ac.uk

- We can estimate uncertainty by checking whether repeated answers agree in meaning.
- Several answer strings can express the same claim, so entropy over exact strings may exaggerate disagreement.
- Group answers by meaning and add their probabilities before computing entropy.
- This gives a meaning-sensitive uncertainty score alongside candidate self-evaluation. Agreement is not proof of correctness.

Why uncertainty over wording can mislead

- Looking only at exact word sequences can confuse uncertainty about phrasing with uncertainty about the answer.
- Several different sentences may say the same thing, while others make competing claims.
- The goal is to measure uncertainty about meaning so we can better separate correct and incorrect answers.
- The score does not prove truth or guarantee that an answer is safe to give.

Two Paris scenarios separate wording from meaning

(a) Scenario 1: No semantic equivalence

Answer $\mathbf{s}$	Likelihood $p(\mathbf{s} \mid x)$	Semantic likelihood $\sum_{\mathbf{s} \in c} p(\mathbf{s} \mid x)$
Paris	0.5	0.5
Rome	0.4	0.4
London	0.1	0.1
Entropy	0.94	0.94

(b) Scenario 2: Some semantic equivalence

Answer $\mathbf{s}$	Likelihood $p(\mathbf{s} \mid x)$	Semantic likelihood $\sum_{\mathbf{s} \in c} p(\mathbf{s} \mid x)$
Paris	0.5	0.9
It's Paris	0.4	
London	0.1	0.1
Entropy	0.94	0.33

- Sample answers, keep their model probabilities, and add probabilities for answers that mean the same thing.
- In scenario 1, Paris 0.5, Rome 0.4, and London 0.1 are three different answers.
- In scenario 2, Paris 0.5 and "It's Paris" 0.4 form one meaning with probability 0.9.
- String entropy stays at 0.94, while semantic entropy falls from 0.94 to 0.33.
- These values use natural logarithms. Repeated agreement on a meaning can still be wrong.

Group answers that imply each other

- Sample several answers to the same question and keep their model probabilities.
- Group answers that express the same meaning in context.
- An inference model judges whether one text implies another, contradicts it, or neither.
- The paper uses DeBERTa-large fine-tuned on MNLI and checks both directions in the question's context.
- Treat answers as equivalent when each implies the other. These grouping judgments can be wrong.

Modern alternatives: tasksource/ModernBERT-large-nli is an encoder NLI candidate; an instruction LLM can judge question-conditioned entailment in both directions. Tradeoff: compact batched inference versus flexible prompts and extra LLM calls.

Probability mass over meanings

$$p(c \mid x) = \sum_{s \in c} p(s \mid x)$$

$$\mathrm{SE}(x) = -\sum_{c} p(c \mid x) \log p(c \mid x)$$

- Add the probabilities of strings that express the same meaning.
- Paris 0.5 and “It’s Paris” 0.4 give the Paris meaning a probability of 0.9.
- Entropy is low when probability lies on one meaning. Competing meanings increase it.

Entropy measures uncertainty in a distribution: low when one outcome dominates, high when probability is spread across outcomes.

x — question and context

s — one answer string

c — a meaning-equivalent group

$p(\cdot \mid x)$ — model probability conditional on x

$\mathrm{SE}(x)$ — semantic entropy; natural logs

Use agreement in meaning to evaluate answering

- The method estimates these quantities from sampled answers, so sampling and grouping matter.

- A model can repeat the same wrong meaning. Evaluate the score against actual correctness.

Discrimination asks whether mistakes rank higher

- A useful uncertainty score assigns higher uncertainty to incorrect answers, so high scores flag likely errors.
- AUROC (Area Under the Receiver Operating Characteristic Curve) measures how well uncertainty scores distinguish incorrect from correct answers.
- AUROC is 1 for perfect ranking and 0.5 for chance.
- AUROC measures the ordering, regardless of the score's numerical scale.

$$\begin{aligned}\mathrm{AUROC} = &\Pr(s(X^+) > s(X^-)) \\ &+ \tfrac{1}{2}\Pr(s(X^+) = s(X^-))\end{aligned}$$

Draw X^+ and X^- independently from the positive and negative classes. s gives a larger score for the positive class.

Uncertainty: positive = incorrect; negative = correct. Confidence: positive = correct; negative = incorrect.

Calibration asks whether a probability means what it says

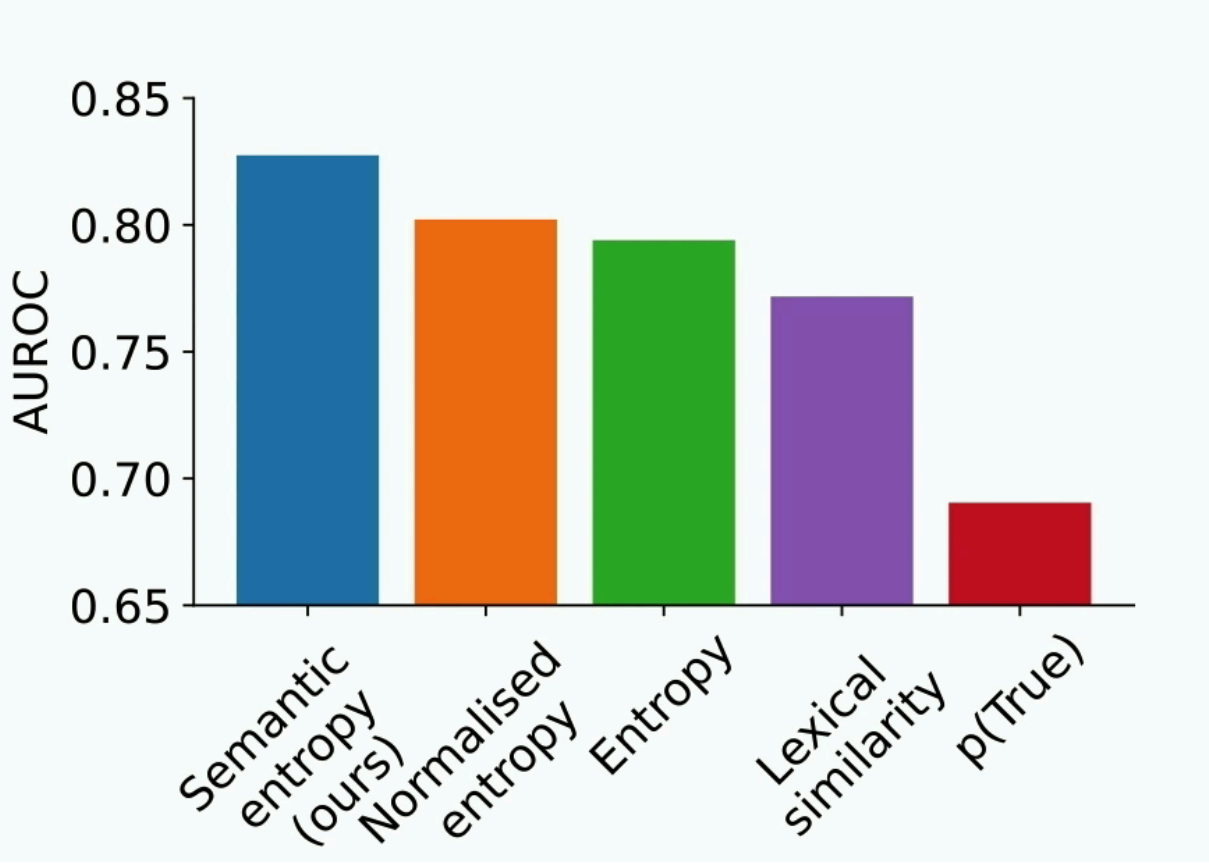
- In the population, among answers assigned a 20% probability of being wrong, the error frequency should be 20%.
- Raw entropy does not have that probability interpretation.
- Use separate labeled examples to map the score to an error probability.
- Test that mapping on held-out examples from the task where it will be used.
- Calibration alone does not choose when the service should answer.
- Check error among accepted answers, coverage, and computation cost.

General workflow: selective answering at matched cost

- How do we use an uncertainty score to decide which answers to deliver?
- Rank answers by their estimated uncertainty; a cutoff withholds the most uncertain ones.
- Choose the cutoff on separate data before the final test.
- Compare methods on the same questions, answers, and correctness labels.
- Match the sampling budget and record computation cost.
- At the same coverage (fraction of answers delivered), compare error among accepted answers.
- At the same error target, compare how many answers each method keeps.

Which score best separates incorrect QA answers?

Pretrained OPT-30B answers closed-book TriviaQA without task finetuning.

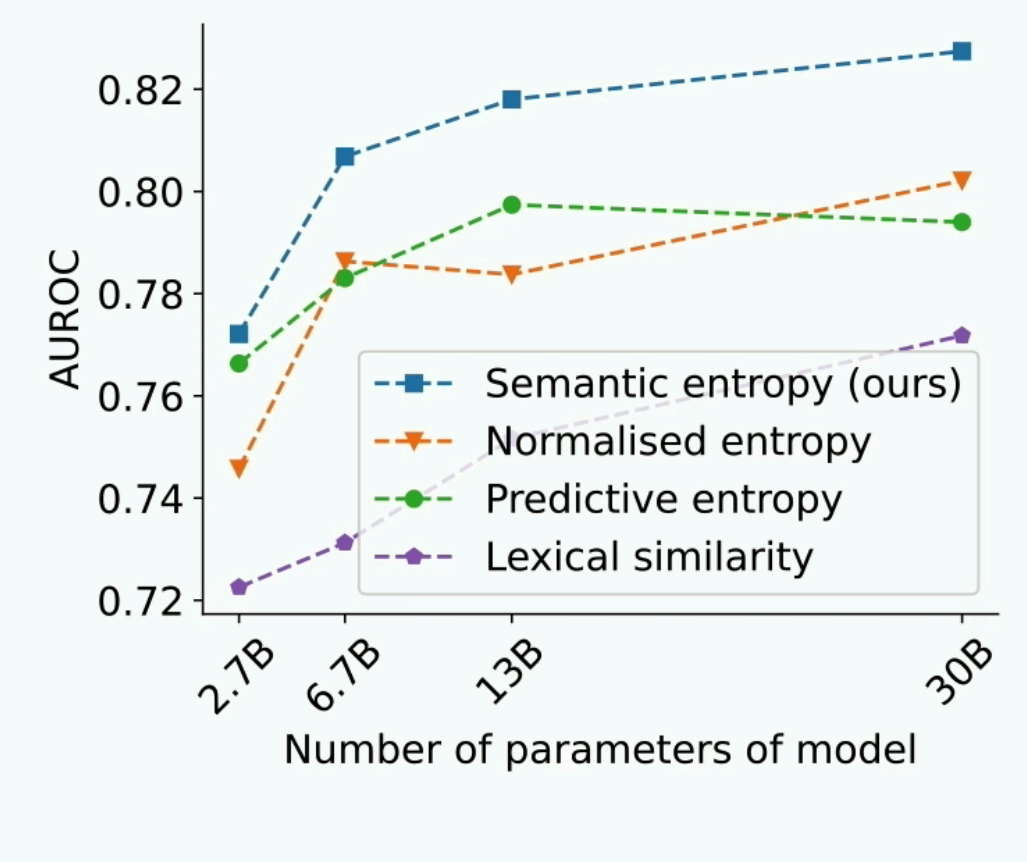


- Use ten separately sampled answers to estimate uncertainty.
- The horizontal axis lists scores. The vertical axis is AUROC; higher means better error ranking, and 0.5 means chance.
- Semantic entropy is highest here, at about 0.83, above the displayed comparison scores.

Lexical similarity: wording overlap. Normalized entropy: answer log-likelihoods divided by length. P(True): Anthropic/Kadavath (2022) self-evaluation.

Does the ranking advantage persist across model sizes?

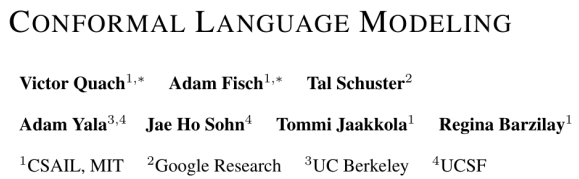
Compare pretrained OPT models from 2.7B to 30B on closed-book TriviaQA, using the same correctness proxy and ten uncertainty samples.



- Predictive entropy measures uncertainty across possible answer strings, before grouping answers by meaning.
- The horizontal axis shows model size.
- Blue squares show semantic entropy; the legend identifies the other uncertainty scores.
- Semantic entropy ranks highest at every size shown, and its curve rises from smallest to largest.
- Calibration and error among accepted answers still need separate checks.

Return a set that contains an acceptable response

Quach et al. · ICLR 2024



CONFORMAL LANGUAGE MODELING

Victor Quach[1,*] Adam Fisch[1,*] Tal Schuster[2]
Adam Yala[3,4] Jae Ho Sohn[4] Tommi Jaakkola[1] Regina Barzilay[1]
[1]CSAIL, MIT [2]Google Research [3]UC Berkeley [4]UCSF

- When committing to one answer is unreliable, return a set of candidate responses.
- The goal is to include at least one acceptable response in that set.
- Conformal language modeling calibrates set generation on held-out examples to control the chance that the set misses every acceptable response.
- The guarantee assumes independent calibration and test examples from the same distribution.

What exactly must the generated set contain?

- Several different answers to the same question may be acceptable.
- Classification has a fixed list of labels. Language has too many possible responses to list.
- Keep a finite set of generated candidates that includes an acceptable response.
- Control the chance that every candidate is unacceptable while keeping the set and sampling cost small.
- Choosing one final answer from that set needs a separate decision.

Background: What is conformal prediction?

- A model's best guess can be wrong. Instead of always returning one prediction, can we return a set of plausible labels with a statistical guarantee that it contains the correct label?
- Conformal prediction uses held-out examples to calibrate the set, assuming exchangeable (the joint distribution is unchanged when examples are reordered) calibration and test data.
- Coverage is marginal across data draws, rather than a promise for each individual question.

- Returning every label includes the truth but gives little guidance.

- Coverage does not make every candidate correct or guarantee uniformly small sets.

Use held-out examples to choose the set's cutoff

- Fit the model first and reserve separate examples with known labels for calibration.
- Use the fixed model to score those examples and choose a cutoff.
- For a new input, return labels that pass that cutoff without knowing the true label.
- A cutoff adjusted for the finite calibration sample can give at least 90% probability of including the true label.
- This requires exchangeable examples; independent examples from the same population are a sufficient case.
- The guarantee averages over calibration data and a new example, rather than covering every input or subgroup.

A prediction set keeps several possible labels

Conformal prediction returns a set of plausible labels that aims to include the true one.



{ fox squirrel 0.99 }



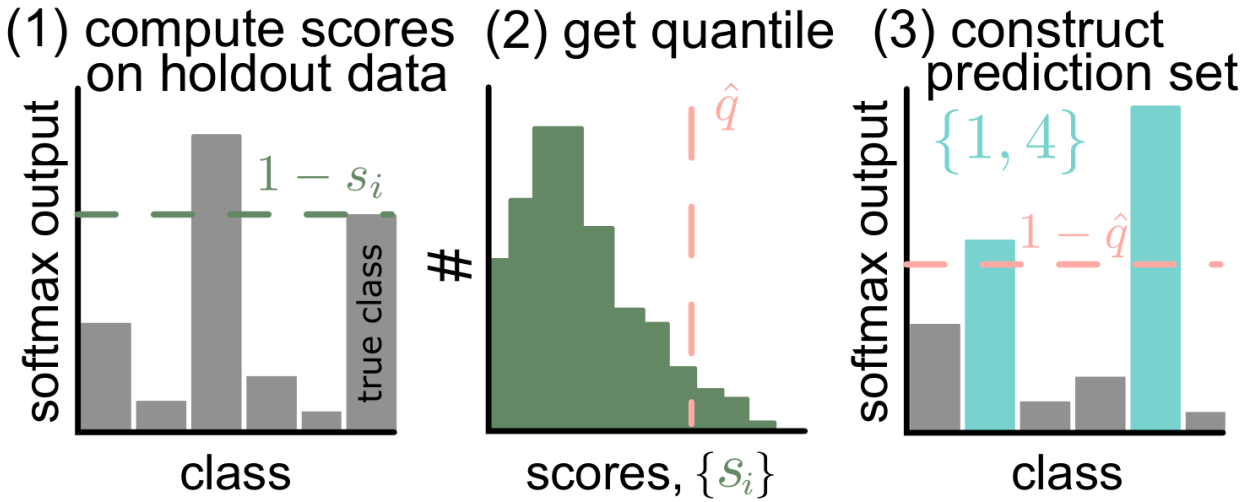
{ fox squirrel 0.82, gray fox, 0.03, bucket, 0.02, rain barrel 0.02 }



{ marmot 0.30, fox squirrel 0.22, mink 0.18, weasel 0.16, beaver 0.03, polecat 0.01 }

- The left set contains only “fox squirrel.” The middle and right sets include more labels.
- Green marks the true label. Other included labels are possibilities that may be wrong.
- Larger sets give less decisive answers, and set size need not perfectly measure difficulty.
- The raw class probabilities need not be calibrated; aim to include the true label without unnecessarily large sets.

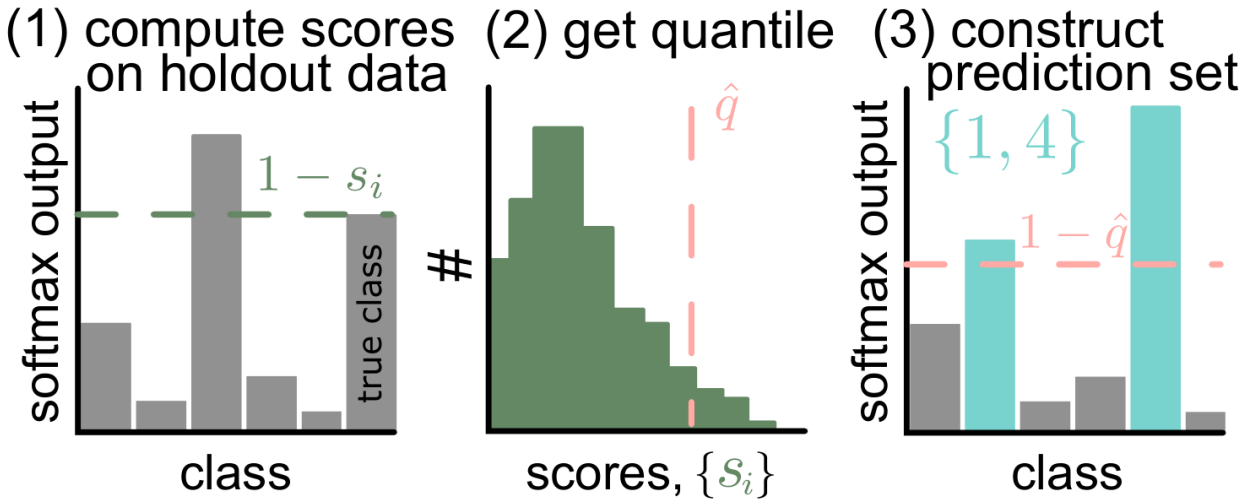
Calibrate the cutoff using held-out labels



- Score. With the fitted model fixed, use held-out examples with known labels: $s_i = 1 - p(y_i \mid x_i)$.
- Calibrate. Set $q = s(\lceil 0.9(n+1) \rceil)$: the finite-sample-corrected 90% quantile of n sorted calibration scores.
- Predict. Include every label with $1 - p(y \mid x) \leq q$. Classes 1 and 4 pass here.

Exchangeable calibration and test examples ⇒ at least 90% marginal true-label coverage.

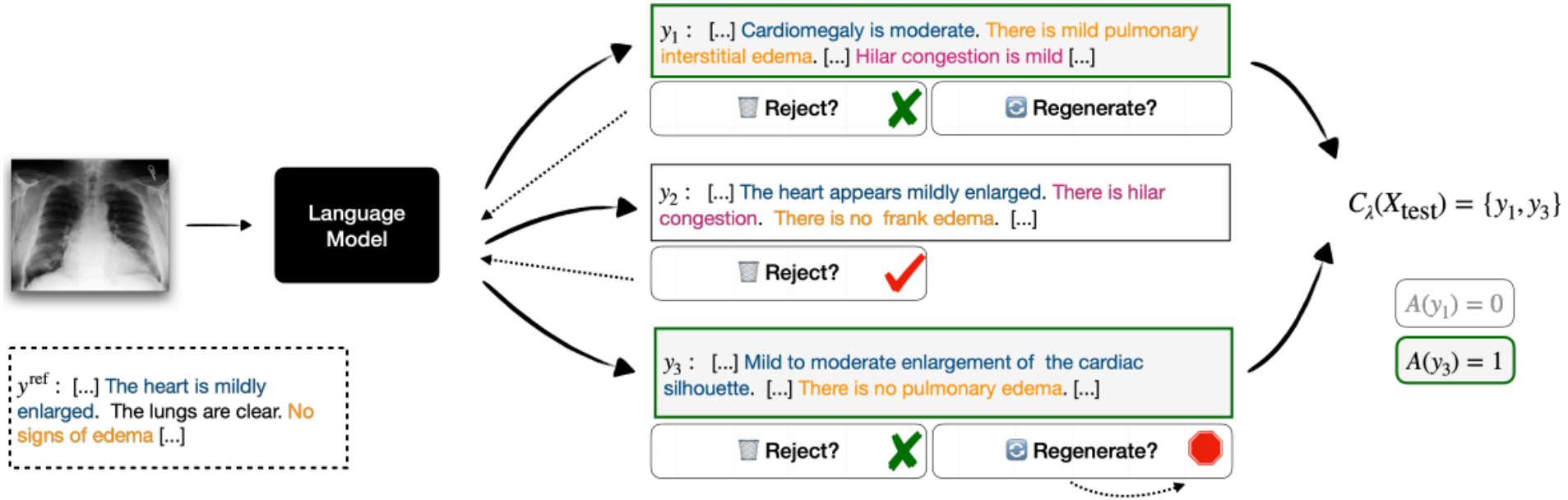
Calibrate the cutoff using held-out labels — continued



- Equivalently, include every class with probability at least 1 minus the cutoff.
- With exchangeable data, true-label inclusion is at least 90%, averaged over calibration data and a new example.
- Why: the new true-label score has an exchangeable rank; the adjusted cutoff covers at least 90% of possible ranks.
- Ties make coverage conservative. Independent examples from the same distribution suffice; there is no separate promise for each input.

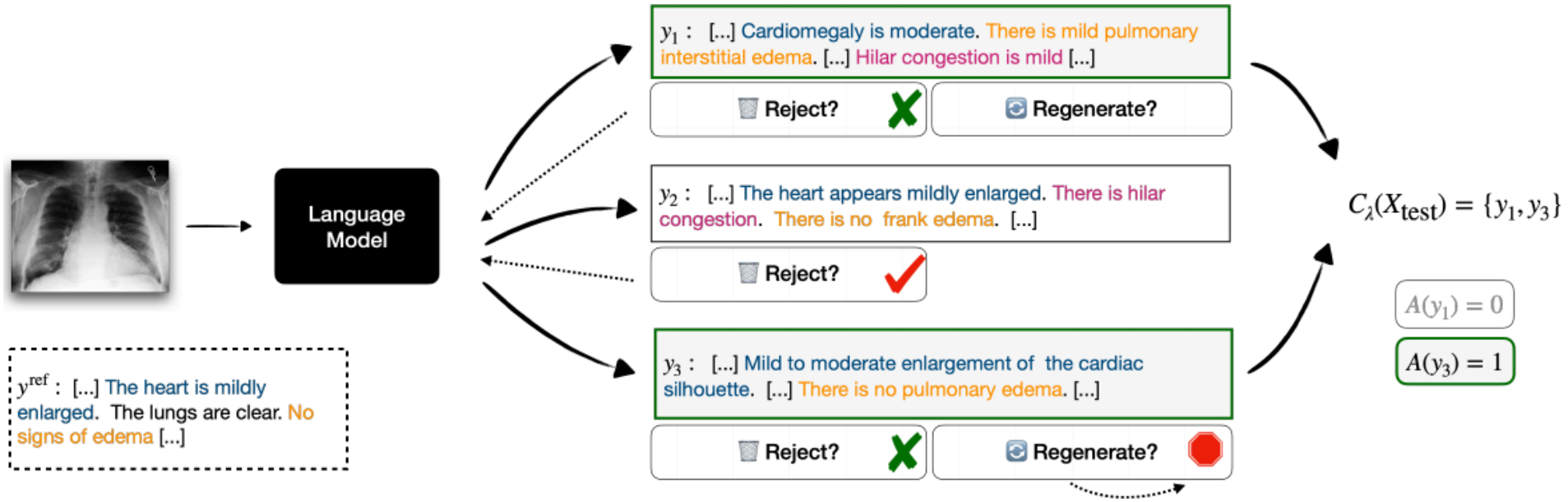
Sample, filter, and check whether to stop

Given a chest X-ray, generate a radiology report describing the findings.



- Sample one complete answer.
- Keep it if its length-normalized likelihood reaches the quality cutoff.
- Keep it if its greatest ROUGE-L (word overlap based on the longest common subsequence) overlap with a retained answer is at or below the similarity cutoff.
- Add a passing answer to the set and check the stopping score; otherwise sample again.

Sample, filter, and check whether to stop — continued



- One acceptable answer makes the set succeed, even when another retained answer is wrong.

- Choosing one final answer from the set needs its own correctness check.

Method example: a kept answer can still be wrong

- We ask a fixed language model: “What is the capital of Canada?”
- “Ottawa” passes the quality proxy and enters the set. “Ottawa, Canada” is rejected as too similar.
- “Toronto” could also pass those proxies, even though it is wrong.
- The sampler stops when its set score reaches a calibrated cutoff, or when its draw budget runs out.

Illustrative example. Accepted means kept by proxy checks; it does not mean known correct.

Method: sample, filter, add, then stop

- Sample a response from the fixed language model.
- Reject it if its quality proxy Q is too low, or its similarity S to a retained response is too high.
- Otherwise add it to C, the retained set, and compute the set-confidence score F(C).
- Stop if F(C) reaches its cutoff; otherwise repeat until the draw budget is exhausted.

Runtime uses proxy scores, not reference labels. $\lambda = (\lambda_1, \lambda_2, \lambda_3)$ sets similarity, quality, and stopping cutoffs.

Method: choose a stopping proxy

$$F_{\mathrm{FIRST-K}}(C) = |C|$$

$$F_{\mathrm{FIRST-K+REJECT}} = N_{\mathrm{drawn}}$$

$$F_{\mathrm{MAX}}(C) = \max_{y \in C} Q(x, y)$$

$$F_{\mathrm{SUM}}(C) = \sum_{y \in C} Q(x, y)$$

- FIRST-K: take K draws without rejection.
- FIRST-K+REJECT: stop after K total draws; rejects count.
- MAX: stop when the best retained quality score reaches its cutoff.
- SUM: stop when total retained quality scores reach its cutoff.

C: retained set. Q(x,y): likelihood proxy for response y to input x. Ndrawn: all attempts. These are stopping proxies, not tests that a correct answer has appeared.

Calibration: judge completed sets using references

- On held-out calibration prompts, run the same sampler without looking at reference answers.
- After generation, use reference labels to judge whether each response is admissible.
- TriviaQA checks reference-answer matches. Radiology checks whether 14 finding labels match the expert reference report.
- A set succeeds if at least one response is admissible; empty sets and all-unacceptable sets fail.

Reference-based admissibility is used for calibration/evaluation only. Runtime acceptance uses the proxy checks.

Calibration: measure failure of the whole sampler

$$L_i(\lambda) = \mathbf{1}\{\nexists y \in C_\lambda(X_i) : A_i(y) = 1\}$$

$$\widehat{R}_n(\lambda) = \frac{1}{n}\sum_{i=1}^{n} L_i(\lambda)$$

- A set fails when none of its retained responses passes the reference-based admissibility check.
- Average these failures to obtain the observed set-failure rate for this threshold configuration.

X_i — held-out calibration prompt i

λ — all three threshold settings

$C_\lambda(X_i)$ — set produced by the complete sampler

$A_i(y)$ — reference-based admissibility; offline only

$L_i(\lambda)$ — 1 for set failure, 0 for success

n — number of calibration prompts

Calibration: Learn Then Test checks configurations

- Each configuration specifies all three cutoffs and therefore the whole sampling procedure.
- Use one calibration split to choose and order promising configurations; reserve an independent split for statistical checks.
- For each configuration, count failed sets on those held-out calibration prompts.
- A low observed failure rate is not enough: account for sample size and checking several configurations.

“Test” means statistical checks on calibration data. A separate final evaluation split measures reported performance.

Calibration: could a low failure count be luck?

$$H_\lambda : \; R(\lambda) > \epsilon$$

$$p_\lambda = \Pr\{\mathrm{Binom}(n, \epsilon) \leq m\}$$

- Count failures of the complete sampler on n independent calibration prompts.
- Small pλ means the count is unusually low for an unsafe configuration; use the multiple-testing rule to decide certification.

$R(\lambda)$ — future set-failure rate over prompts and sampling

ϵ — maximum allowed failure rate

n — independent calibration prompts checked

m — failed sets observed for this configuration

p_λ — evidence against the unsafe-configuration hypothesis

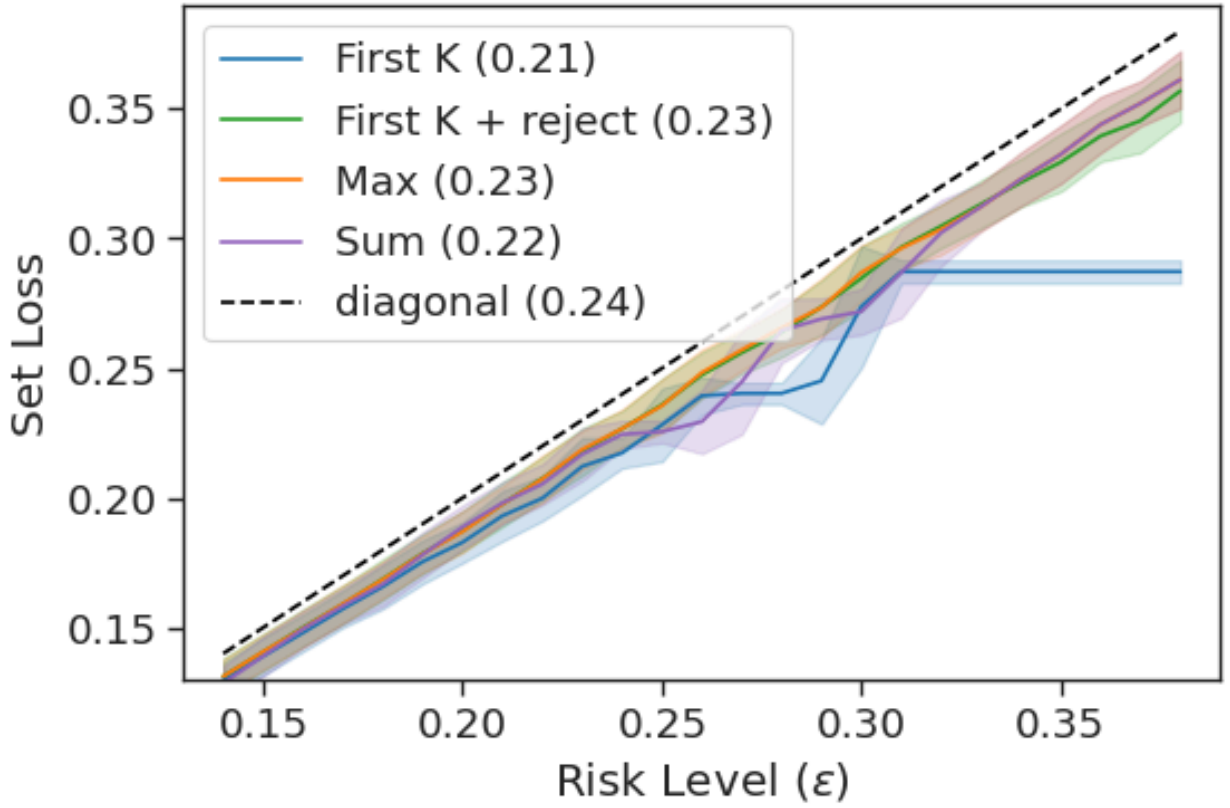
Illustrative: 2 failures in 100 prompts, with ε = 0.10, gives pλ ≈ 0.002.

Calibration: certify first, then choose low cost

- Fix the order using the learning split; on fresh calibration data, test in that order and stop at the first failed statistical check.
- This controls the chance of certifying any unsafe configuration at δ, across all configurations checked.
- With probability at least $1-\delta$ over calibration data, every certified configuration has future set-failure rate at most ε.
- Choose low set size and excess sampling among certified configurations; if none is certified, abstain.

Requires i.i.d. calibration/test examples from the same distribution, with the LM and score functions fixed.

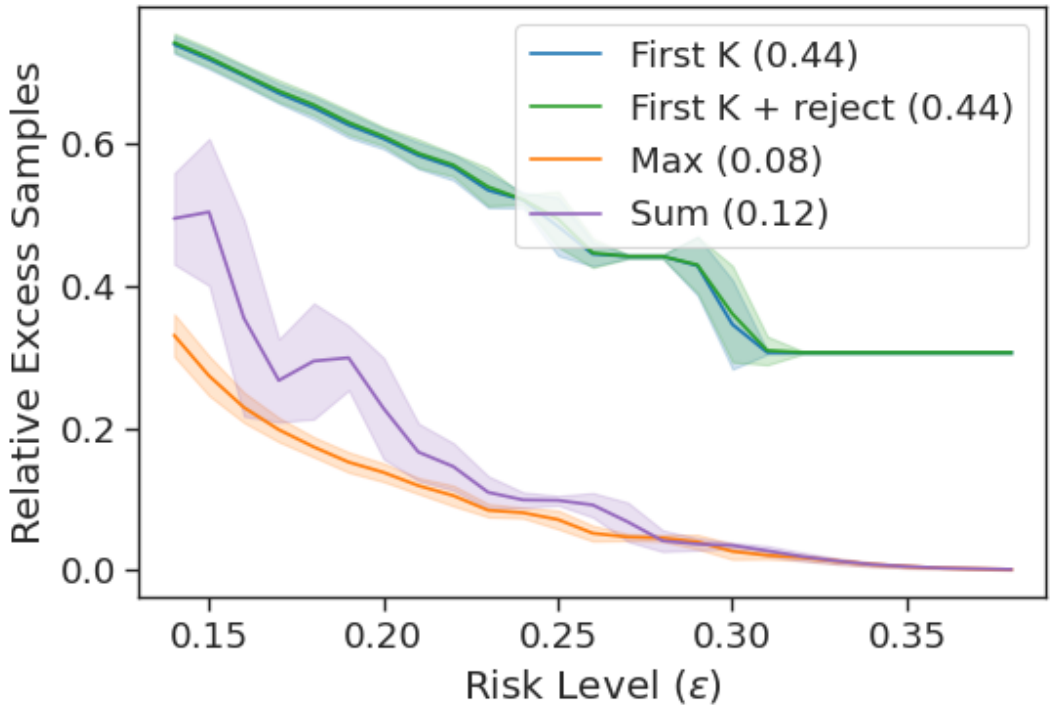
Results: calibrated sets meet the target on TriviaQA

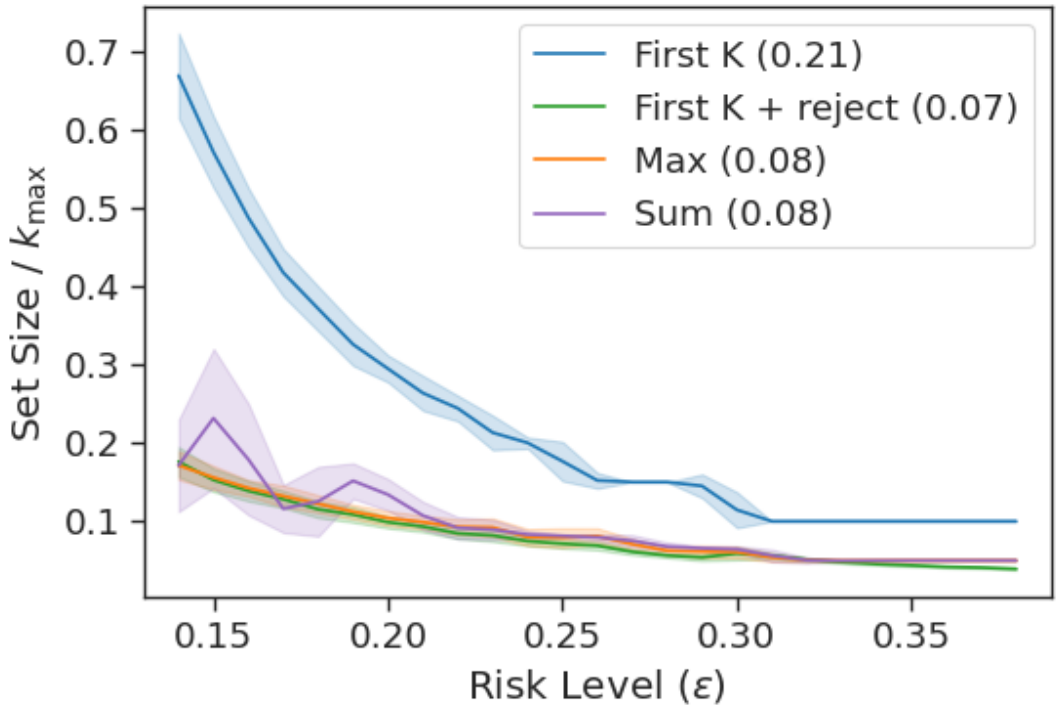


- Horizontal: target failure rate ε. Vertical: the fraction of test sets with no admissible response.
- All mean curves stay below the target diagonal here; FIRST-K is more conservative.
- Parentheses give normalized AUC: average set loss across these ε values. First K (0.21), Max (0.23).

TriviaQA · LLaMA-13B · budget 20 draws · 100 trials; shading ±1 standard deviation · δ = 0.05.

Read the sampling effort and returned set size

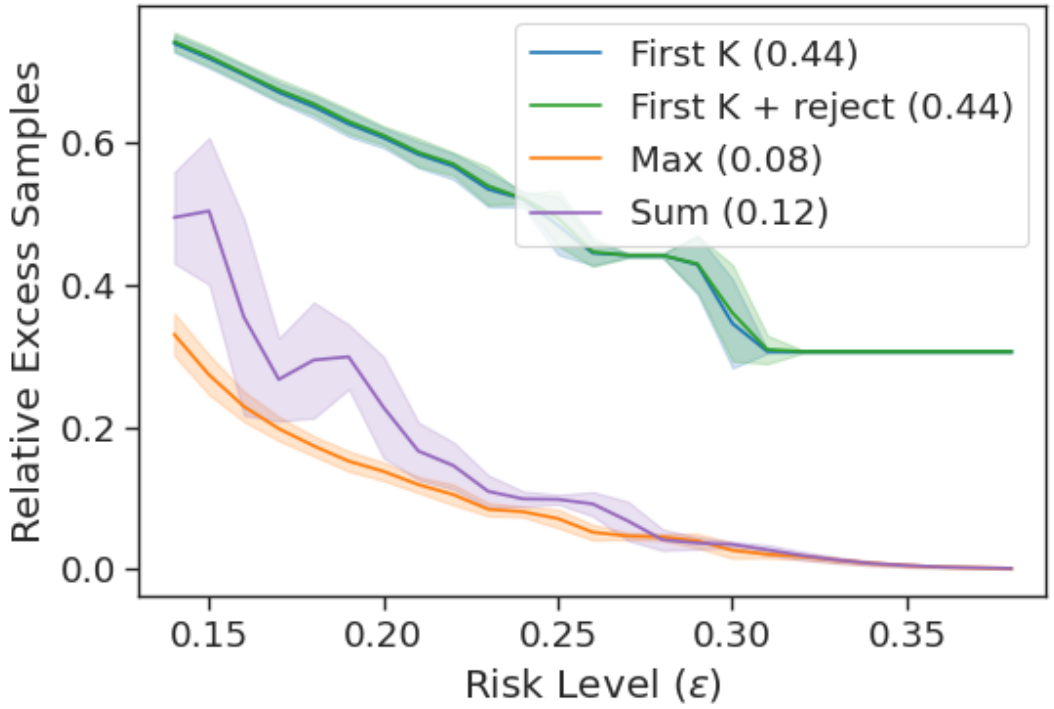


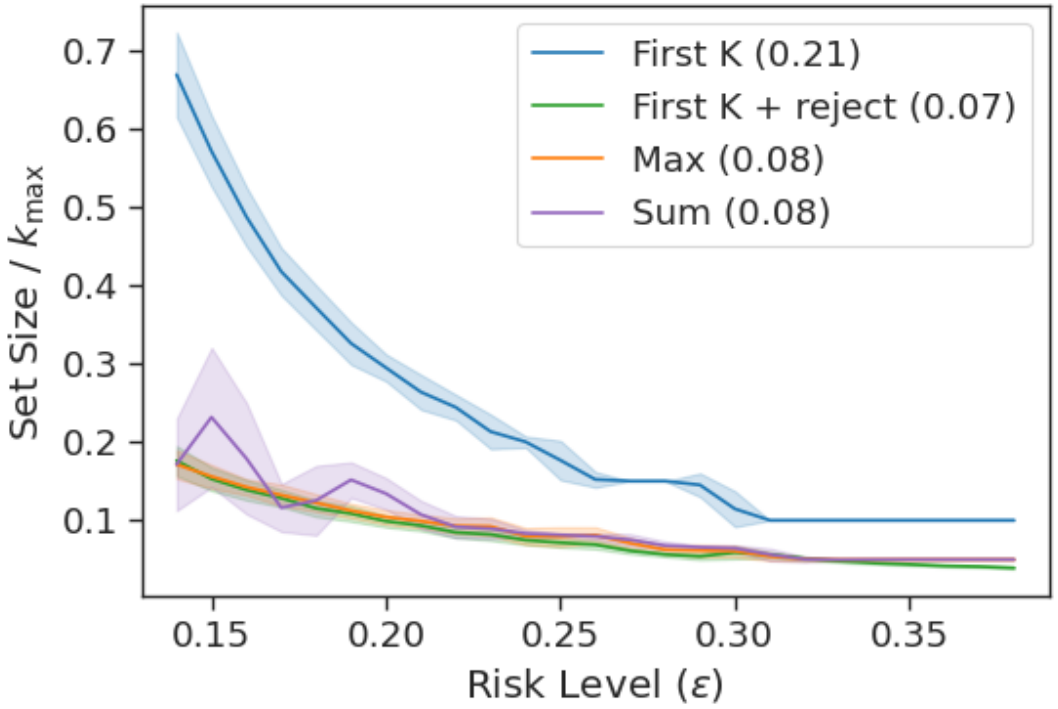


- Extra sampling is the fraction of all attempts after the first acceptable answer.
- Normalized set size = retained candidates / 20; the budget counts rejected draws too.
- Lower is better. Rejection makes smaller sets; MAX wastes fewer attempts here.

Reference labels are used for evaluation only.

Compare the cost after validating set risk



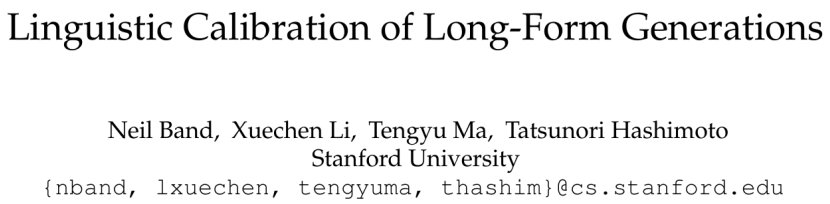


- Compare cost only after the setting meets the reliability target.
- Lower cost AUC means less extra sampling or a smaller final set.
- Set-size AUC is 0.08 for MAX and SUM, versus 0.21 for FIRST-K.

100 trials; shading: ±1 standard deviation.

What confidence does a reader take away?

Band, Li, Ma & Hashimoto · ICML 2024



- A useful long answer should communicate appropriate uncertainty to its reader.
- The model's own confidence score does not show how a person or another model interprets the wording.
- Train the model to write passages that communicate uncertainty clearly. The goal is calibrated reader probabilities over possible answers to related questions.

Useful wording should support calibrated reader beliefs

- A reader reads the passage, then receives a related question.
- The reader assigns probabilities to that question's possible answers.
- Different wording can change these answer probabilities.
- Train the model to write useful long answers that express uncertainty through words or numbers.
- Evaluate the reader's answer probabilities with proper scoring rules, alongside question-answering accuracy.
- Calibration means these probabilities match observed correctness; it depends on the reader and the questions.

Same correct book, different expressed confidence

Question: In which book does a boy hide in a ship's barrel of apples?

Book named in the response	Model-expressed probability
Treasure Island	75%
The Wonderful Wizard of Oz	15%
The Adventures of Tom Sawyer	5%
Another book	5%

- Two responses answer the same question and both correctly identify Treasure Island.
- The model trained for factual accuracy answers directly. The model trained to communicate calibrated confidence says "most likely" and assigns 75%.
- Giving the correct answer and communicating appropriate confidence are different goals.

Same correct book, different expressed confidence — continued

Book named in the response	Model-expressed probability
Treasure Island	75%
The Wonderful Wizard of Oz	15%
The Adventures of Tom Sawyer	5%
Another book	5%

- Train the model to give useful, correct answers and express appropriate confidence in them.
- This example illustrates the distinction; one answer with 75% confidence does not establish calibration.

Read a passage, then forecast a related answer

- The model writes a long answer to an open-ended question.
- A reader, either a person or another model, reads it and assigns probabilities across answers to a related question.
- Those probabilities reflect the confidence the reader inferred from the passage.
- Compare these forecasts with correct answers across cases using a specified reader and task, because different readers may interpret the same wording differently.

Reward the reader's probability for the correct answer

$$R_{\mathrm{LC}}(f(x,z),y) = \log f(x,z)_y$$

- A fixed reader uses passage z to answer question x and assigns a probability to each answer.
- Use the reader's probability for the correct answer y; the reward is its natural logarithm.
- Higher probability for the correct answer gives a higher reward.

x — downstream question

z — generated passage

$f(x,z)$ — reader's answer-probability vector

y — correct downstream answer

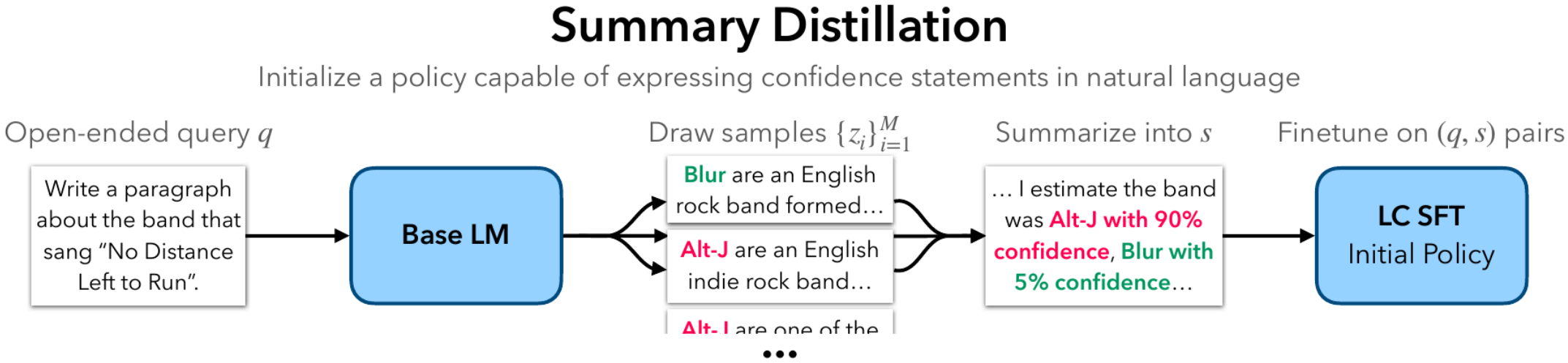
R_{LC} — log-score reward

What the reader-based reward teaches

- A tiny probability gives a strongly negative score, so confidently excluding the correct answer is costly.

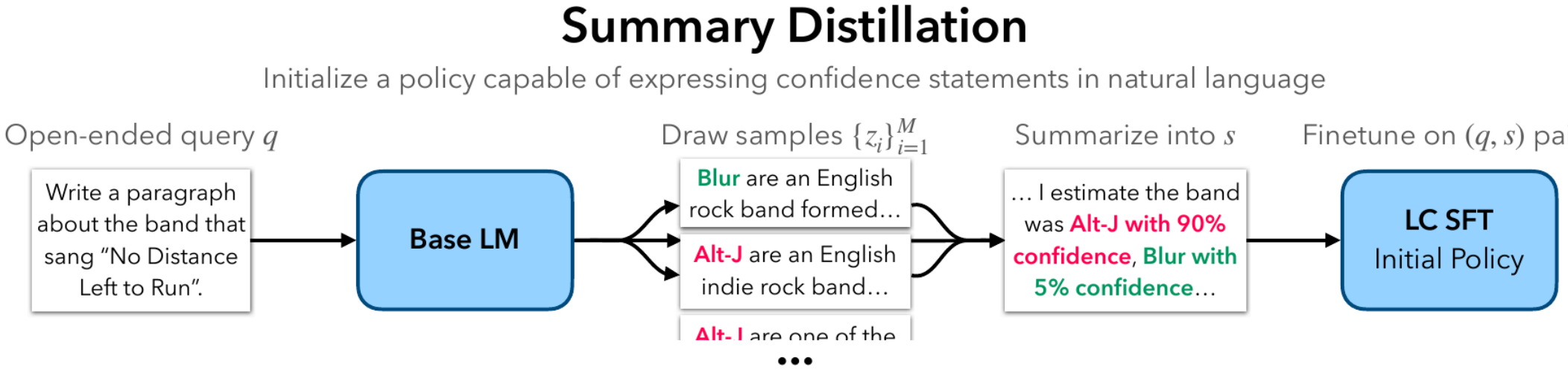
- Training maximizes the average reward and connects usefulness to what the reader learns from the passage.

Combine sampled responses into a summary



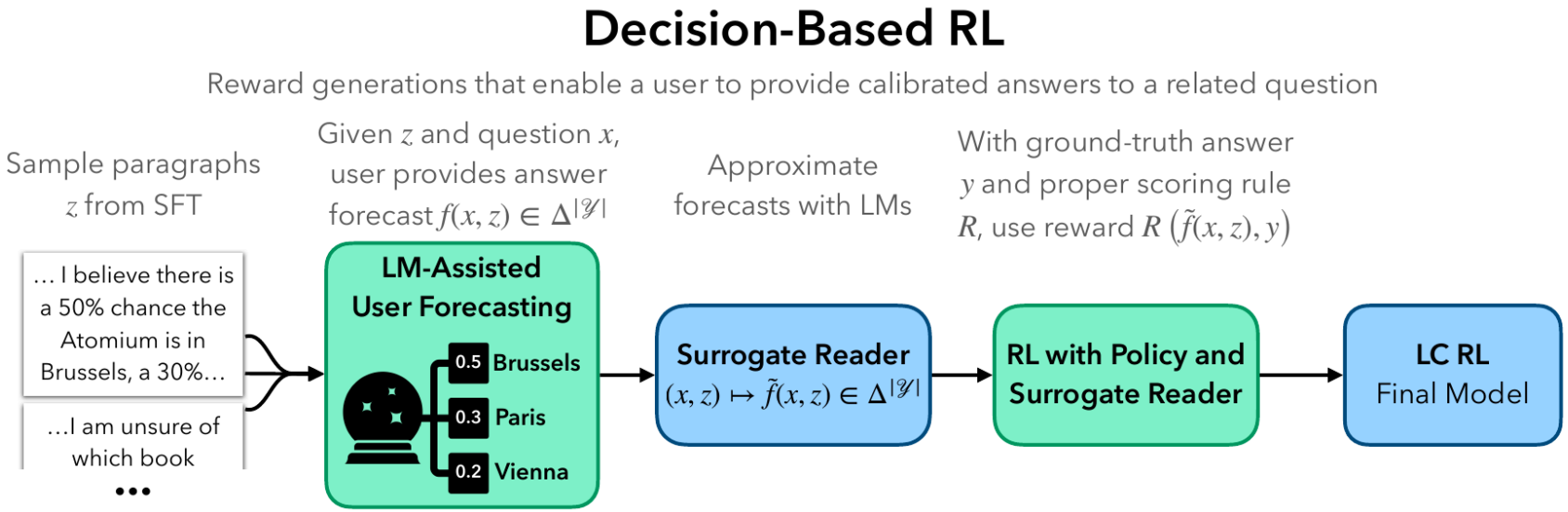
- Summary distillation creates training examples of answers that express uncertainty.
- Sample several responses to one question to reveal variation in the model's answers.
- A summarizing model combines those samples into one target answer.
- Summarize individual claims using how often they appear, expressing confidence in words or numbers.

Use the summary as a supervised training target



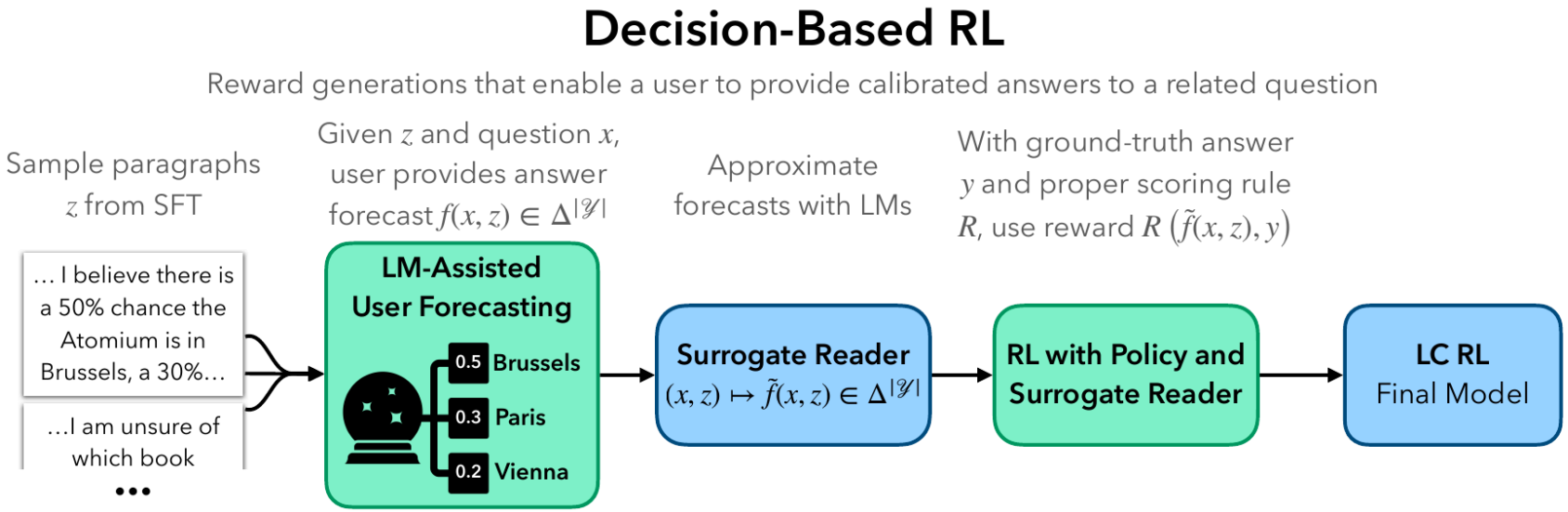
- The displayed summary gives 90% to Alt-J and 5% to Blur.
- Pair the original question with its summary: the question is the input and the summary is the target response.
- Supervised finetuning teaches the model to produce this kind of answer.
- Summary distillation initializes this work's policy for RL; it is not required by the calibration definition.

Estimate the forecast a reader makes



- The supervised model generates a passage, and a related question defines what the reader must answer.
- A reader model estimates answer probabilities from that question and passage.
- The reader's forecast is f; a learned approximation f˜ lets us estimate it during training.
- The approximation can differ from what an actual reader believes.

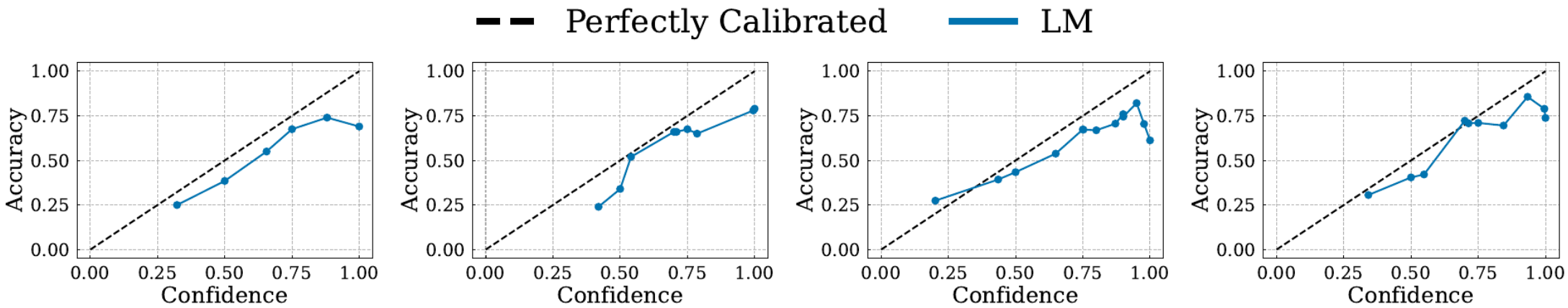
Update the generator using the reader-based reward



- Reward the log probability assigned to the correct answer.
- Reinforcement learning updates the generator to improve that reward.
- Test human readers to see whether the gains transfer.

Read the reader-calibration diagrams

Evaluate Llama 2 7B on TriviaQA: human readers are on the left, and a simulated reader is on the right.



(a) LC SFT, human readers. ECE: 0.163

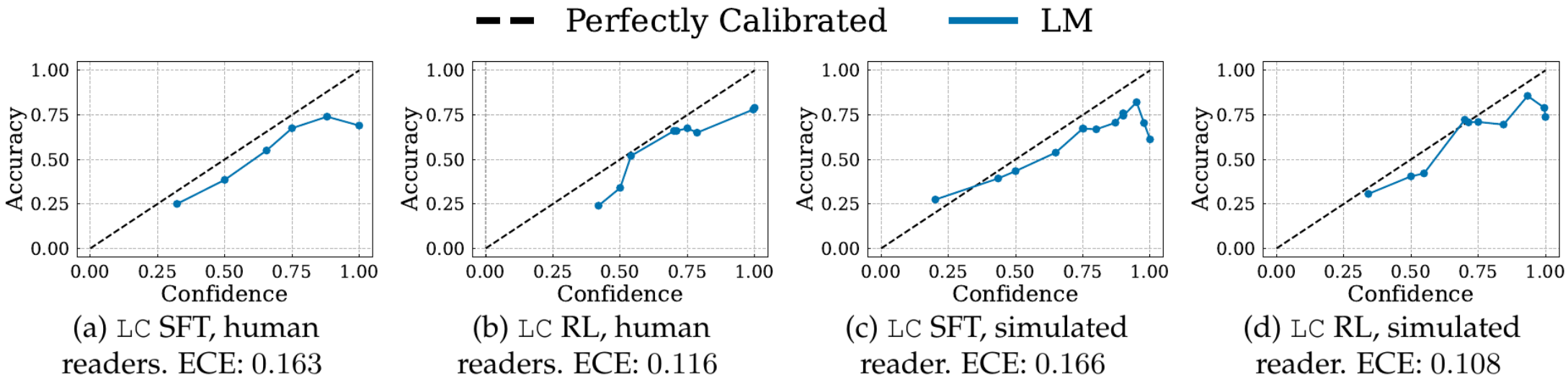
(b) LC RL, human readers. ECE: 0.116

(c) LC SFT, simulated reader. ECE: 0.166

(d) LC RL, simulated reader. ECE: 0.108

- Readers rely on the passage rather than their own background knowledge.
- The horizontal axis is average confidence in a bin; the vertical axis is the fraction correct.
- The diagonal shows calibration: below is overconfidence and above is underconfidence.

Further RL improves measured reader calibration



(a) LC SFT, human readers. ECE: 0.163

(b) LC RL, human readers. ECE: 0.116

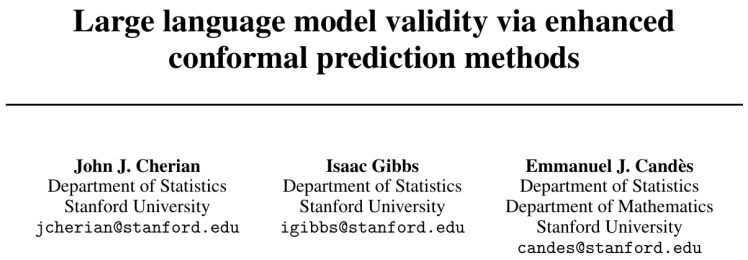
(c) LC SFT, simulated reader. ECE: 0.166

(d) LC RL, simulated reader. ECE: 0.108

- Expected calibration error, or ECE, averages each bin's absolute confidence–accuracy gap, weighted by its number of answers. Lower is better.
- Human-reader ECE falls from 0.163 to 0.116; simulated-reader ECE falls from 0.166 to 0.108.
- The curves span different confidence levels. Further RL improves measured calibration here, but another reader or task may change the result.

Retain useful claims while meeting a validity target

Cherian, Gibbs & Candès · NeurIPS 2024



Large language model validity via enhanced conformal prediction methods

John J. Cherian
Department of Statistics
Stanford University
jcherian@stanford.edu

Isaac Gibbs
Department of Statistics
Stanford University
igibbs@stanford.edu

Emmanuel J. Candès
Department of Statistics
Department of Mathematics
Stanford University
candes@stanford.edu

- A long answer can mix correct facts with mistakes. We want to remove mistakes while retaining useful information.
- The preceding work teaches communication of uncertainty; here we filter claims inside an answer.
- Split the answer into claims, score them, and remove claims below a calibrated cutoff.
- Filtering can remove correct information. Meet a chosen reliability target while retaining more useful claims.

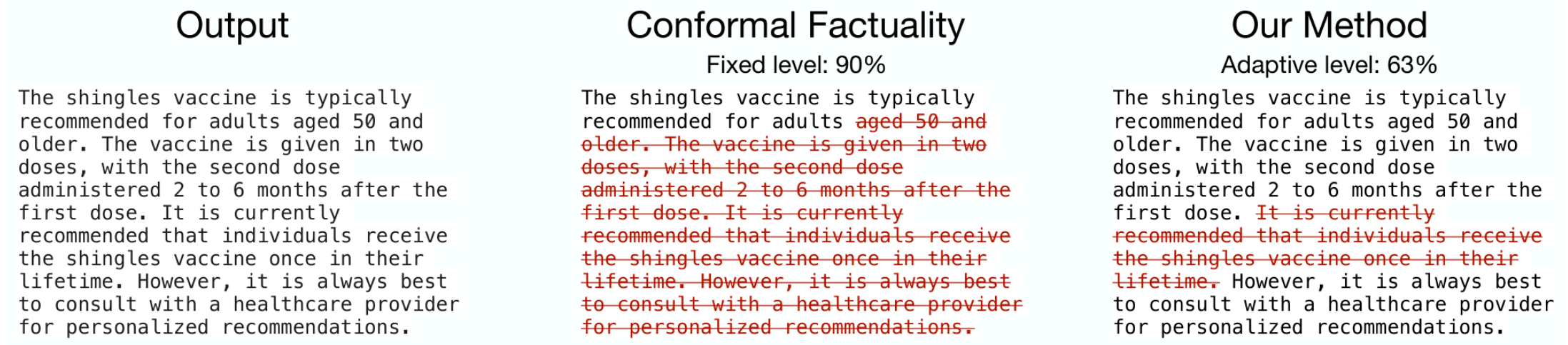
Method: improve scores and adapt the reliability level

- Split a response into claims, score them, and filter below a calibrated cutoff.
- A 90% factuality guarantee overall does not imply 90% reliability on every topic.
- Conditional boosting learns better claim scores to retain more useful claims while preserving calibrated reliability within specified groups, such as medical topics.
- Level-adaptive conformal prediction chooses a reliability target from prompt/response features to retain useful information, then calibrates it on separate data.

Group guarantees apply to the specified calibration groups; they are not exact per-prompt correctness probabilities.

Filtering a real response changes what the user learns

Reliability = no false claims in the entire filtered response.



- 90% fixed: baseline frequency scores; prescribe one nominal reliability level.
- 63% adaptive: boosted scores plus a calibrated adaptive level, targeting at least 70% claim retention.
- 90% / 63% are reliability levels—not percentages of claims correct or retained.

Higher targets can remove more information. Both scoring and level change in this comparison. Calibration gives population/group guarantees, not an exact correctness probability for this particular response.

From a long answer to a calibrated filtered answer

- Start with a long answer and extract its factual claims.
- Compute several confidence signals for each claim; agreement is one signal.
- Learn how to combine the signals so filtering retains useful claims.
- Choose a fixed validity level or learn a level rule; then freeze both the scorer and level rule.
- Use separate labeled responses for final cutoff calibration after learning.
- Score a new answer's claims and keep those meeting its calibrated cutoff.

1. Extract factual claims from the long answer

- Split a long answer into claims that can be scored separately.
- GPT-4o splits a long answer into sentences, then separates each sentence into factual claims.
- BM25 keyword search finds similar examples from FActScore to show GPT-4o how to split a sentence into factual claims.
- The retrieved examples guide extraction, not factual verification; pass each claim to the scoring step.

2. Compute confidence signals: agreement is one

- One confidence signal is agreement: generate five additional GPT-3.5-Turbo answers to the same prompt.
- An LLM compares each claim with each additional answer: +1 for support, –1 for contradiction, or 0 for absence.
- Average the five judgments to obtain the FREQUENCY score; three supports and two absences give 0.6.
- Agreement is a consistency signal, not a truth probability; combine it with confidence, position, and self-check probability.

$$s_{\text{freq}}(c) = \frac{1}{5}\sum_{j=1}^{5} v_j(c)$$

c — one claim; j — one of five answers

$v_j(c)$ — +1 supports, –1 contradicts, 0 absent

Three supports + two absences:

$$\frac{1+1+1+0+0}{5} = 0.6$$

3–4. Learn the scorer and level rule, then freeze them

- Combine the four claim signals with weights; learn weights that help retain more claims.
- For proposed weights, use one internal labeled split to calibrate conditional filtering cutoffs.
- Use another internal split to filter answers and evaluate how many claims remain.
- Replace hard keep/delete decisions with a smooth count, so gradients can update the weights.
- Update the weights to increase retention, accounting for how each cutoff moves as scores change; recalibrate at every iteration.
- Choose a fixed reliability level or learn a level rule, then freeze both the scorer and level choices.

5–6. Calibrate separately, then filter new answers

- Freeze the learned scorer and level rule before final calibration on fresh reserved labeled responses.
- For each labeled response, find the smallest cutoff that leaves its allowed error count.
- Conditional conformal calibration uses quantile regression on these cutoffs and prompt-response features.
- At test time, score and filter claims with the calibrated cutoff, without correctness labels.

$$S_i = \inf\{\tau : L_i(\tau) \leq \lambda\}$$

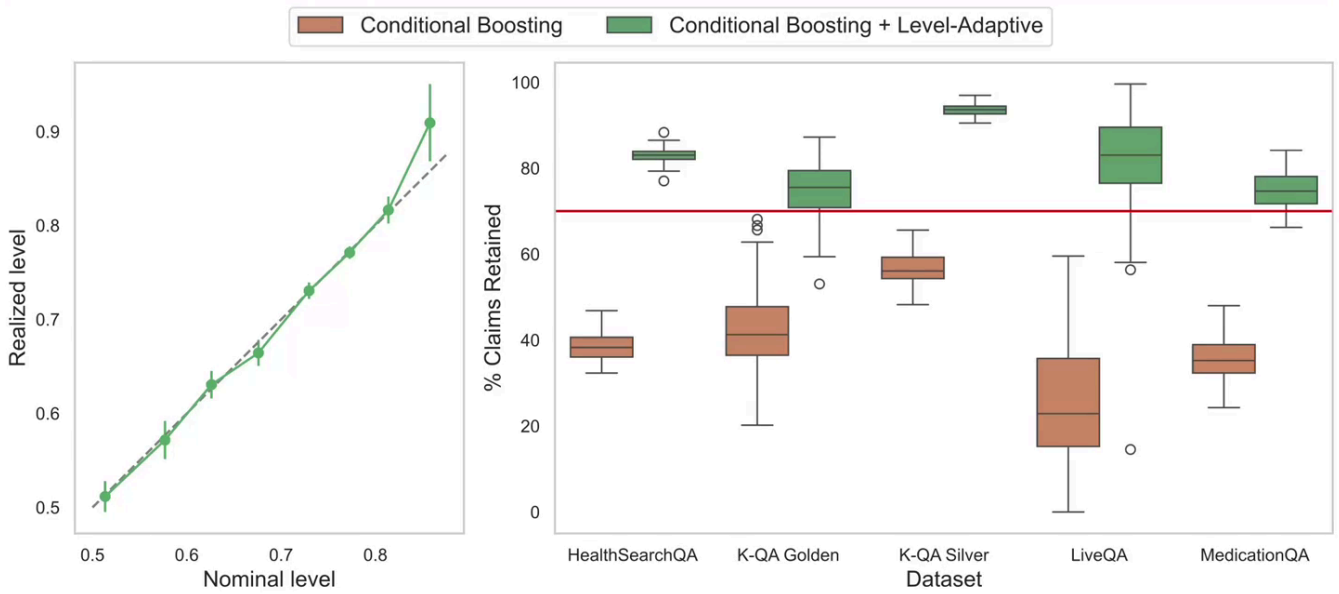
$L_i(\tau)$ — labeled response's loss after filtering; λ — allowed loss

The cutoff depends on prompt-response features; calibration addresses the chosen feature groups under exchangeability.

Summary: retain useful claims with calibrated reliability

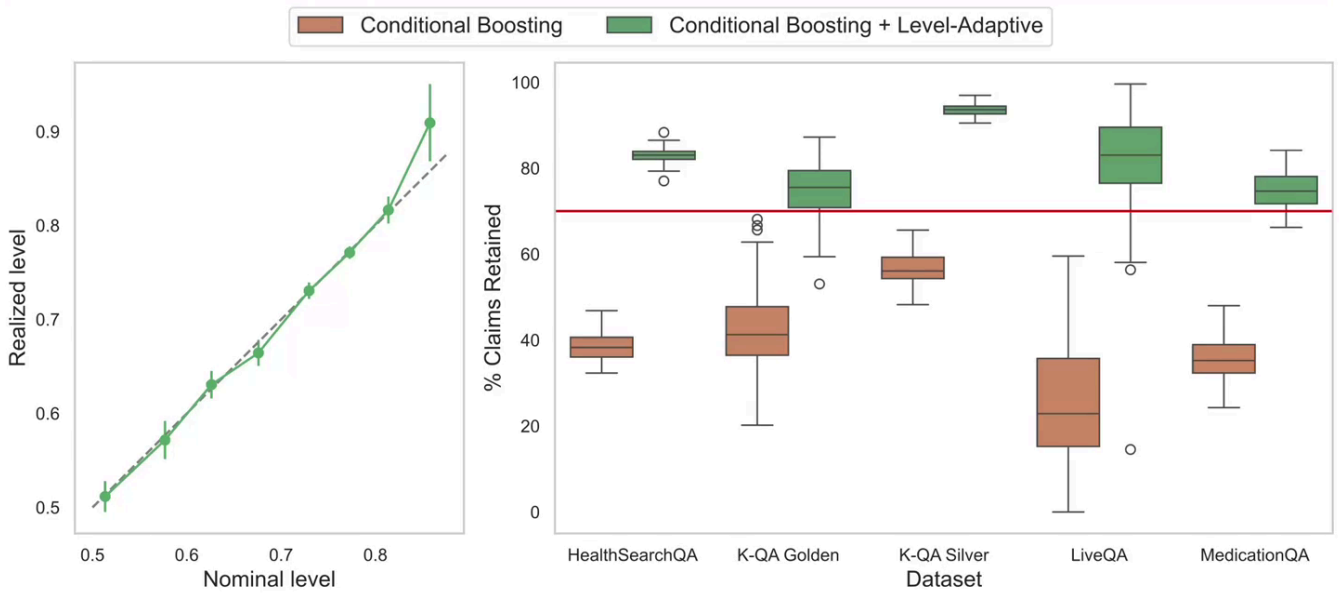
- Split the response into individual claims.
- Score each claim and filter claims below a calibrated cutoff.
- Better scores retain more useful claims at a fixed reliability target; adaptive levels can relax the target to preserve information.
- Calibrate the final rule on fresh labeled data so its stated reliability is supported within the specified groups and level bins.

Read medical validity and information retention



- Orange: conditional boosting improves weighted claim scores with conditional calibration.
- Green also adapts the requested validity target to the input.
- Nominal means the requested response-reliability target. Realized means the observed fraction of responses with no unsupported claims.
- Left: compare nominal and realized validity. Right: compare the percentage of claims retained.

Adapted levels retain more medical claims



- Orange learns claim scores; green also adapts the validity level to the input.
- The red line marks the 70% retention objective; green generally retains more claims.
- These methods issue different levels, so this is not a shared 90% comparison.
- Support labels use model judgments and supplied references, including pseudoanswers; these results do not establish clinical safety.

Teach when a clarifying question improves the answer

Zhang, Knox & Choi · ICLR 2025

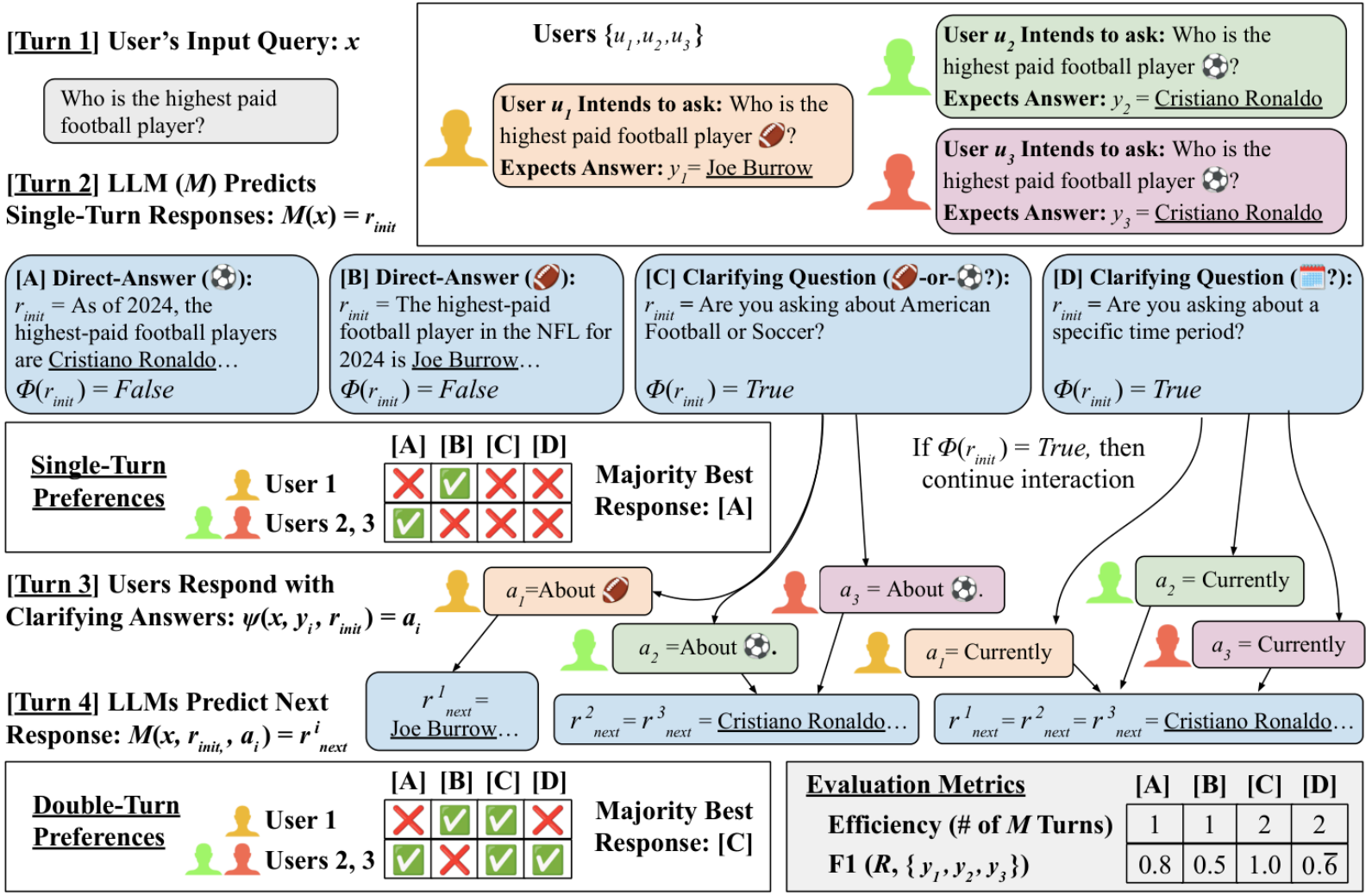


MODELING FUTURE CONVERSATION TURNS TO TEACH LLMS TO ASK CLARIFYING QUESTIONS

Michael J.Q. Zhang, W. Bradley Knox, Eunsol Choi

- An ambiguous question can mean different things.
- This work teaches the assistant when a clarifying question will lead to a better final answer than answering immediately under a guessed interpretation.
- Ask about the uncertainty that changes the answer, such as whether “football” means American football or soccer.
- A useful question resolves that ambiguity; asking about the time period leaves the sport unclear.

Follow all four turns of the interaction



- M(x) is the model's initial reply.
- ϕ(r_init) checks whether the reply asks a clarifying question.

Compare initial replies by the final answers they enable

- Start with one ambiguous question and users with different known intended meanings.
- Try alternative initial replies: answer for soccer, answer for American football, or ask which sport.
- For a clarifying question, simulate each user's clarification, then generate the final answer.
- Compare each final answer with that user's intended reference; judge direct answers against the same references.
- Prefer the initial reply that helps more users reach their expected answers.
- When outcomes tie, favor answering directly to avoid needless turns.

Train first replies from preferred comparisons

- DPO (Direct Preference Optimization) trains on preferred versus less-preferred response pairs, learning to favor the preferred response.
- Simulate user replies for different possible intentions, then generate the downstream final answers.
- Prefer the initial response whose downstream answers better match the users' intended gold answers, averaged across simulated intentions.
- These comparisons teach when to ask and how to ask a useful clarifying question.

Compare direct answers and clarification questions

	Llama2 # (↓)	Llama2 Answer F1 (↑) Unamb / Amb / All	Llama3 # (↓)	Llama3 Answer F1 (↑) Unamb / Amb / All	Gemma # (↓)	Gemma Answer F1 (↑) Unamb / Amb / All
Direct-Ans SFT						
w/ Greedy	1	25.4 / 16.8 / 21.1	1	31.2 / 19.2 / 24.8	1	26.1 / 16.8 / 21.1
w/ Sampled	1	25.0 / 17.2 / 21.4	1	28.2 / 20.2 / 24.7	1	23.7 / 17.9 / 21.4
Clarify SFT	2	31.0 / 21.6 / 25.9	2	37.6 / 26.5 / 31.5	2	35.7 / 23.6 / 28.8
Clarify DPO						
w/ RM	2	31.0 / 25.7 / 28.3	2	36.2 / 26.7 / 30.9	2	33.9 / 25.7 / 29.5
w/ Likelihood	2	30.2 / 23.9 / 27.2	2	43.5 / 29.6 / 359	2	37.3 / 26.8 / 31.5
w/ Match	2	**38.3 / 28.2 / 32.8**	2	42.9 / **3.17 / 36.5**	2	**40.7 / 28.6 / 33.9**
Clarify-or-Direct-Ans						
SFT	1.12	25.6 / 18.4 / 21.3	1.40	35.3 / 23.5 / 28.2	1.43	22.3 / 19.0 / 20.3
DPO	1.56	28.9 / 21.1 / 24.3	1.57	35.2 / 25.1 / 29.1	1.61	28.2 / 22.2 / 24.6

- Clarify SFT imitates clarifying questions. Clarify DPO with Match improves them using future answers; both always ask. Clarify-or-Direct-Answer DPO learns to ask or answer directly.
- Answer F1 measures overlap with intended gold answers; higher is better. Unamb / Amb / All mean unambiguous / ambiguous / all questions. # Turns: 1 = direct; 2 = clarify then answer; fractions are averages.
- Match: simulate replies, generate downstream answers, score their match to gold, average across intentions, and rank candidate initial responses for DPO.

SFT = supervised finetuning; Direct-Ans SFT answers immediately.

Learn when to clarify or answer directly

	Llama2 # (↓)	Llama2 Answer F1 (↑) Unamb / Amb / All	Llama3 # (↓)	Llama3 Answer F1 (↑) Unamb / Amb / All	Gemma # (↓)	Gemma Answer F1 (↑) Unamb / Amb / All
Direct-Ans SFT						
w/ Greedy	1	25.4 / 16.8 / 21.1	1	31.2 / 19.2 / 24.8	1	26.1 / 16.8 / 21.1
w/ Sampled	1	25.0 / 17.2 / 21.4	1	28.2 / 20.2 / 24.7	1	23.7 / 17.9 / 21.4
Clarify SFT	2	31.0 / 21.6 / 25.9	2	37.6 / 26.5 / 31.5	2	35.7 / 23.6 / 28.8
Clarify DPO						
w/ RM	2	31.0 / 25.7 / 28.3	2	36.2 / 26.7 / 30.9	2	33.9 / 25.7 / 29.5
w/ Likelihood	2	30.2 / 23.9 / 27.2	2	43.5 / 29.6 / 359	2	37.3 / 26.8 / 31.5
w/ Match	2	**38.3 / 28.2 / 32.8**	2	42.9 / **3.17 / 36.5**	2	**40.7 / 28.6 / 33.9**
Clarify-or-Direct-Ans						
SFT	1.12	25.6 / 18.4 / 21.3	1.40	35.3 / 23.5 / 28.2	1.43	22.3 / 19.0 / 20.3
DPO	1.56	28.9 / 21.1 / 24.3	1.57	35.2 / 25.1 / 29.1	1.61	28.2 / 22.2 / 24.6

- Likelihood scores reference-answer probabilities after simulated replies.
- Match checks exact-match final answers after simulated replies.
- Clarify-or-Direct SFT learns questions and direct answers.
- Clarify-or-Direct DPO learns from final outcomes and favors direct answers on ties.

Adaptive Retrieval Without Self-Knowledge?

Moskvoretskii et al. · ACL 2025



Adaptive Retrieval without Self-Knowledge?
Bringing Uncertainty Back Home

Viktor Moskvoretskii[1,3], Maria Lysyuk[2,1], Mikhail Salnikov[2,1], Nikolay Ivanov[1], Sergey Pletenev[2,1], Daria Galimzianova[4], Nikita Krayko[4], Vasily Konovalov[2,5], Irina Nikishina[6], Alexander Panchenko[2,1]

[1]Skoltech, [2]AIRI, [3]HSE University, [4]MTS AI, [5]MIPT, [6]University of Hamburg

Correspondence: vvmoskvoretskii@gmail.com

- Retrieval is costly and can introduce misleading evidence, even when the model already knows the answer.
- Can uncertainty tell us when retrieval is actually needed?
- Compare 35 methods across six question-answering datasets.
- Measure answer quality, retrieval calls, and additional language-model calls separately.

When is retrieval worth an extra call?

- Decide whether a clear factual question needs external evidence before answering.
- Never retrieving avoids retrieval calls but leaves missing knowledge unresolved.
- Always retrieving adds a call for every question and may introduce irrelevant information.
- Compare final answer quality, retrieval calls, and model calls on the same questions.

Learn a retrieval decision from answering outcomes

- Generate training answers without retrieval and measure their uncertainty.
- Evaluation: an answer counts as correct if it contains the reference answer.
- Train a classifier to predict success from the uncertainty score.

The reference is the expected benchmark answer; containment is not a complete factual audit.
Classifiers: threshold, logistic regression, decision tree, KNN, MLP.

Use the learned retrieval decision on a new question

- For a new question, measure a preliminary answer’s uncertainty and apply the classifier.
- Retrieve and answer again when it predicts failure; otherwise keep the preliminary answer.
- Count the extra generation used to make the retrieval decision.

Fewer retrievals can coexist with extra model calls

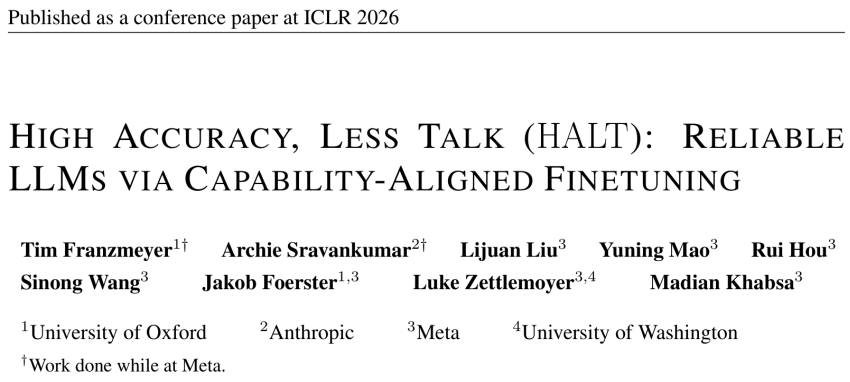
TriviaQA uses 500 questions and Llama-3.1-8B-Instruct.

Strategy	In-Accuracy	LM calls / question	Retrievals / question
Never retrieve	63.6%	1.0	0.00
Always retrieve	61.0%	1.0	1.00
Max Entropy + learned gate	**65.0%**	**1.2**	**0.22**

- In-Accuracy checks whether the generated answer contains the reference answer.
- Max Entropy uses 78% fewer retrievals than always retrieving, with more model calls.
- Model calls include uncertainty estimation, query rewriting, and rationales where used; counts do not measure latency, FLOPs, or money.
- The strongest uncertainty method varies across datasets.

Keep the correct part of a partly wrong answer

Franzmeyer et al. · ICLR 2026



Published as a conference paper at ICLR 2026

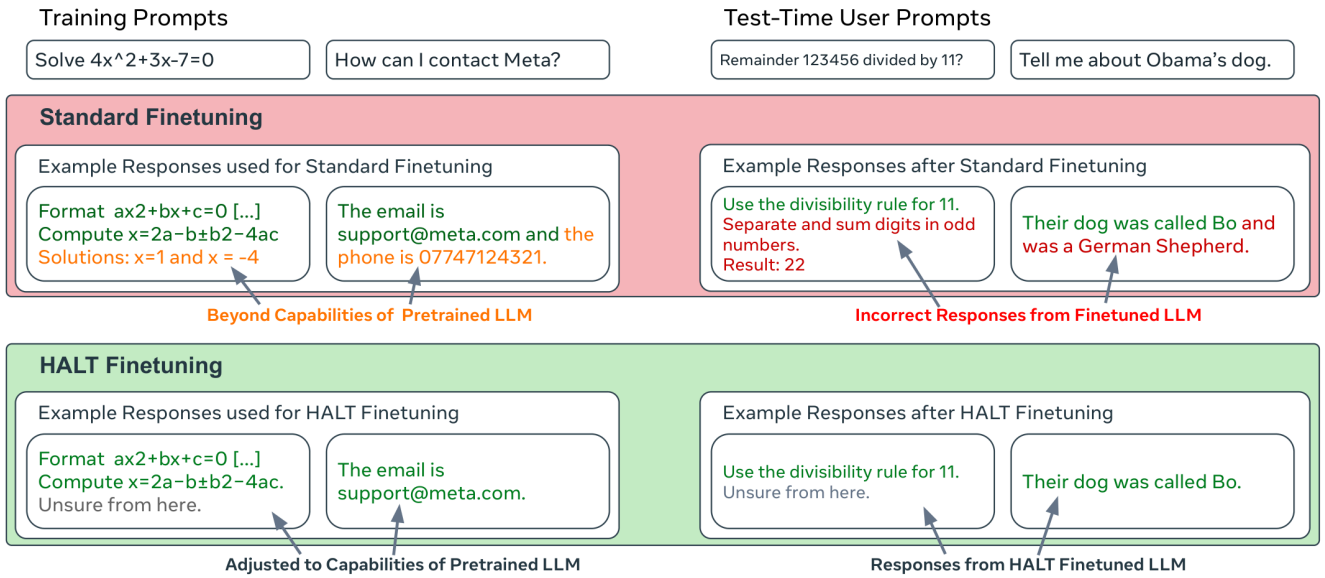
HIGH ACCURACY, LESS TALK (HALT): RELIABLE LLMS VIA CAPABILITY-ALIGNED FINETUNING

Tim Franzmeyer[1†] **Archie Sravankumar**[2†] **Lijuan Liu**[3] **Yuning Mao**[3] **Rui Hou**[3]
Sinong Wang[3] **Jakob Foerster**[1,3] **Luke Zettlemoyer**[3,4] **Madian Khabsa**[3]

[1]University of Oxford [2]Anthropic [3]Meta [4]University of Washington
[†]Work done while at Meta.

- An answer can begin with correct facts or reasoning and then continue into errors.
- Keep the useful correct part instead of discarding the whole answer.
- Rewrite the model's own training answers to preserve supported facts and stop unsupported reasoning.
- Train the model to reproduce those rewritten answers, then measure correctness and completeness.
- HALT learns partial answers; it does not provide a perfect error detector at runtime.

An answer can be partly useful and partly wrong



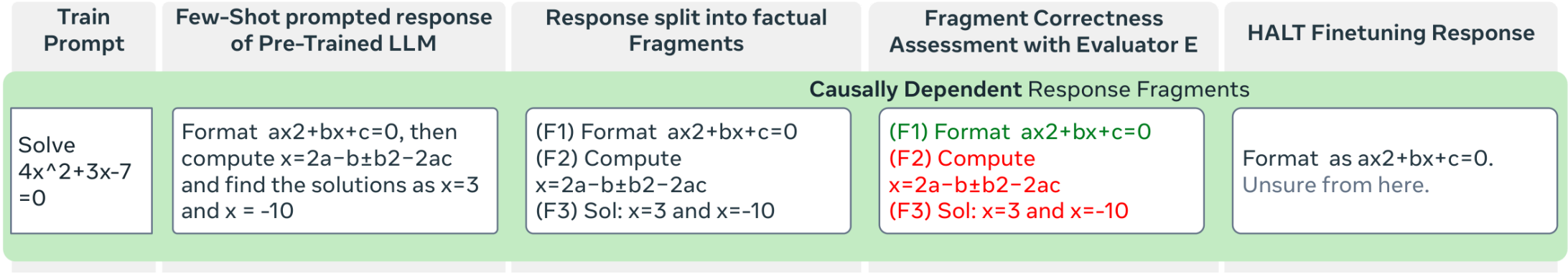
- Supervised finetuning makes target tokens more likely. A full target can demand content beyond what the pretrained model reliably produces.
- A shortened target teaches supported content and an explicit stopping phrase, "Unsure from here."
- The left examples are training targets; the right examples use different test questions after finetuning.
- These illustrate the intended behavior, rather than aggregate measurements or a perfect stopping rule.

Create capability-aligned targets before finetuning

Train Prompt	Few-Shot prompted response of Pre-Trained LLM	Response split into factual Fragments	Fragment Correctness Assessment with Evaluator E	HALT Finetuning Response
Causally Dependent Response Fragments				
Solve 4x^2+3x-7 =0	Format ax2+bx+c=0, then compute x=2a−b±b2−2ac and find the solutions as x=3 and x = -10	(F1) Format ax2+bx+c=0 (F2) Compute x=2a−b±b2−2ac (F3) Sol: x=3 and x=-10	(F1) Format ax2+bx+c=0 (F2) Compute x=2a−b±b2−2ac (F3) Sol: x=3 and x=-10	Format as ax2+bx+c=0. Unsure from here.
Independent Response Fragments				
Tell me about Barack Obama's dog!	Barack Obama, the 45th President of the USA, had a deceased German Shepherd called Sunny.	(F1) Barack Obama is the 45th president of the USA. (F2) Barack Obama's dog is called Sunny. (F3) Barack Obama's dog Sunny is deceased. (F4) Barack Obama's dog Sunny was a German Shepherd.	(F1) Barack Obama is the 45th president of the USA. (F2) Barack Obama's dog is called Sunny. (F3) Barack Obama's dog Sunny is deceased. (F4) Barack Obama's dog Sunny was a German Shepherd.	Barack Obama had a dog called Suny.

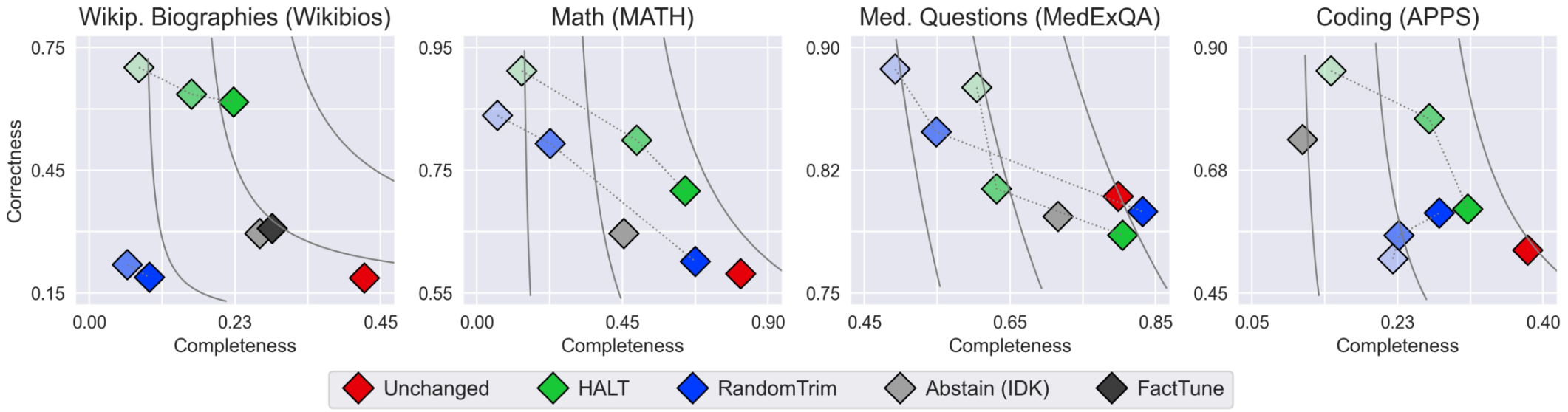
- Few-shot prompt the pretrained model to generate its own answer, then split it into factual claims or reasoning steps.
- Llama3-405B checks biography claims against full Wikipedia articles; for math, medical questions, and coding, it checks numbered fragments against reference step-by-step solutions.
- Remove wrong independent facts while keeping supported facts.
- For sequential reasoning, stop before the first wrong step and add "Unsure from here."
- Train on the rewritten targets; the evaluator prepares data and is not a runtime gate.

Delete independent errors; stop dependent reasoning



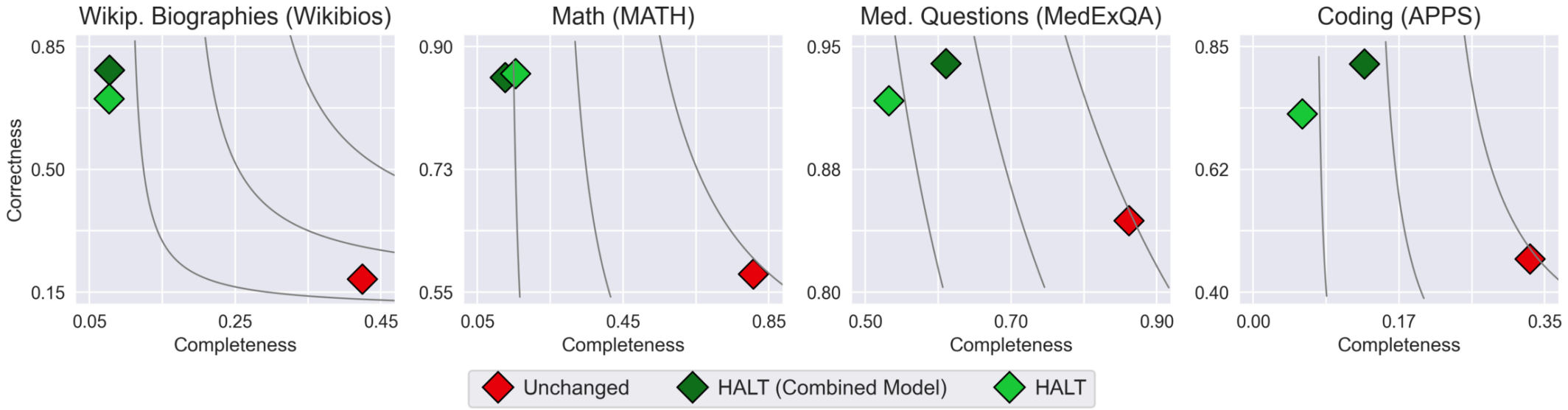
- The source row keeps the supported equation setup and replaces the wrong continuation with "Unsure from here."
- Initial model answer—incorrect.
- Stop at the first wrong step because later conclusions may depend on it.
- For independent facts, remove each unsupported claim while retaining separately supported facts.

Correctness improves as completeness decreases



- Completeness is correct content divided by desired content; correctness is correct fragments divided by produced fragments.
- A short answer can be mostly correct while leaving much of the request unanswered.
- Read each domain separately; higher correctness often costs completeness.
- Gray curves combine equal F1, balancing correctness and completeness; the upper right is better.
- Each green setting trains a different model. It is not a runtime control on one model.

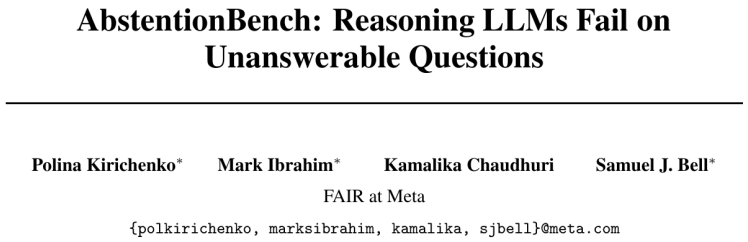
One combined model: 51% to 87% fragment correctness



- Train one Llama3-70B model on equal shares of biographies, math, medical questions, and coding.
- Across the four domains, average fragment correctness rises from 51% to 87%.
- The figure reports 25% absolute completeness; the abstract reports 53% of baseline completeness.
- More accurate partial answers still leave many desired fragments unanswered.
- The evaluator prepares targets; the trained model answers without it as a runtime gate.

Benchmark the choice to answer or abstain

Kirichenko et al. · NeurIPS 2025 Datasets and Benchmarks



AbstentionBench: Reasoning LLMs Fail on Unanswerable Questions

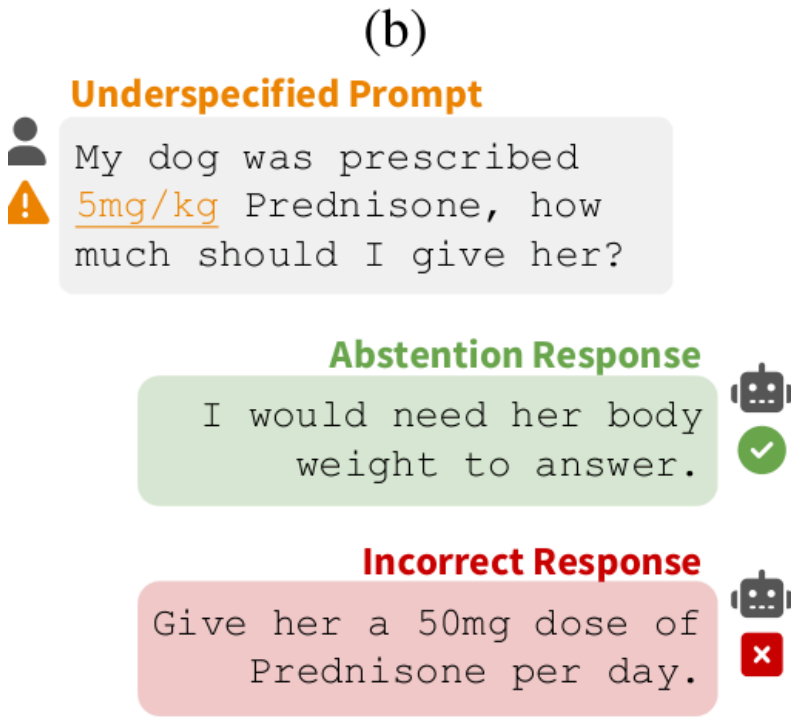
Polina Kirichenko* **Mark Ibrahim*** **Kamalika Chaudhuri** **Samuel J. Bell***

FAIR at Meta

{polkirichenko, marksibrahim, kamalika, sjbell}@meta.com

- Test whether the model recognizes when it should avoid a definite answer.
- Accuracy on answerable questions does not reveal guesses on unanswerable questions or unnecessary refusals.
- Test missing context, ambiguity, outdated information, and false premises.
- Measure correct answers and warranted abstentions.
- Asking for missing information or qualifying an answer can help more than a blanket refusal.

Does the question support a definitive answer?



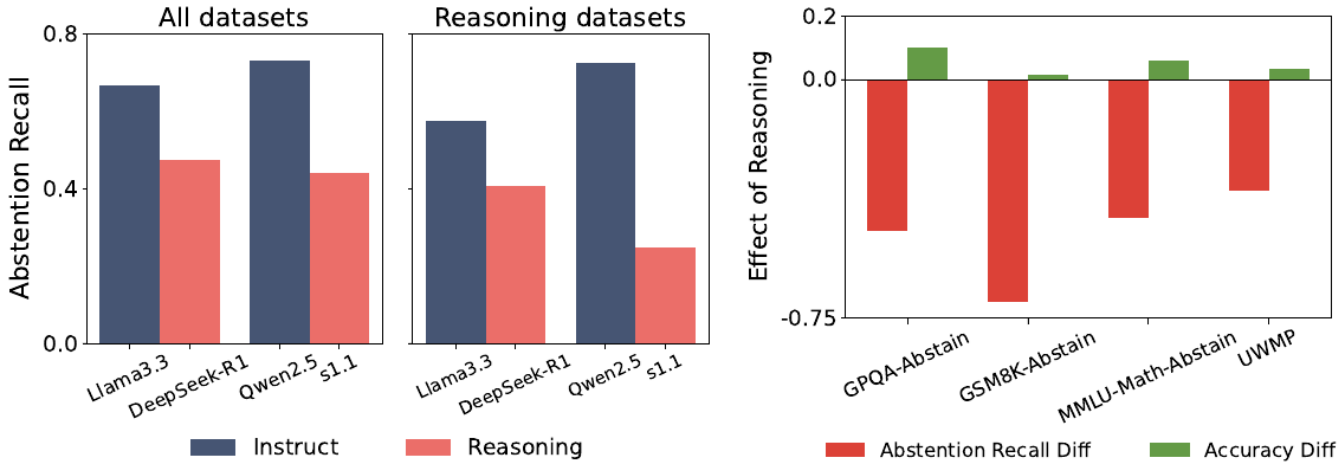
- The source prompt gives 5 mg/kg but omits the dog's weight; a definitive dose is unsupported.
- Asking for the missing weight withholds a definite answer while helping the user.
- Abstention includes expressing uncertainty or asking for clarification, rather than only refusing.
- Avoid unsupported answers while continuing to answer questions with enough information.

Good abstention avoids both kinds of mistake

The benchmark covers different reasons a question may not support a definitive answer.

- One mistake is answering when the assistant should abstain.
- The other is abstaining when the assistant could answer; good abstention avoids both.
- The six scenarios are unknown answers, false premises, outdated information, subjective questions, underspecified context, and underspecified intent.
- Recall asks how often needed abstentions occur. Precision asks how many abstentions were needed.
- Answer correctness is evaluated separately; the benchmark covers 20 models and 20 datasets.

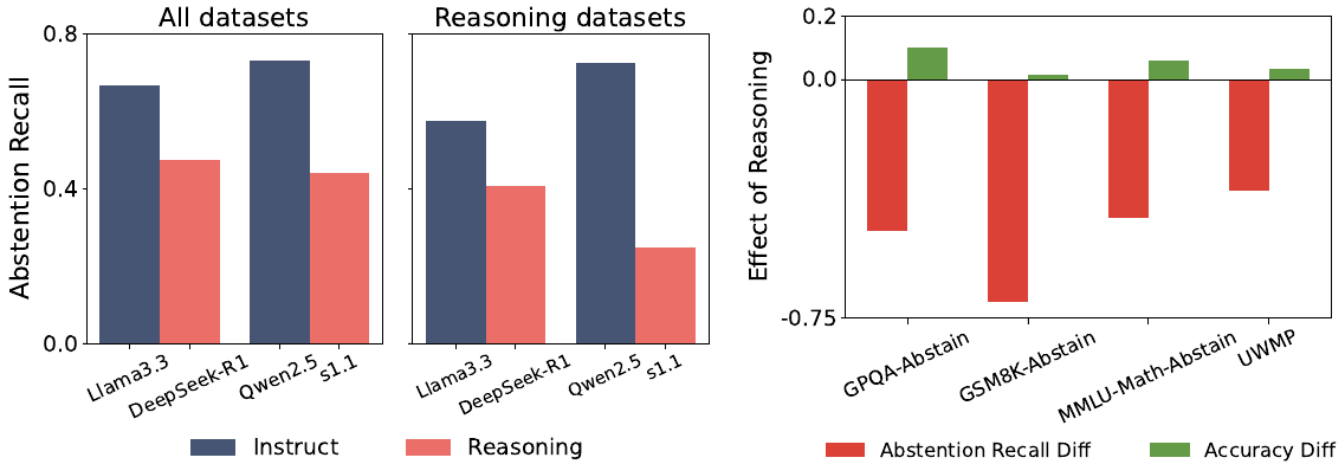
Reasoning models recover fewer warranted abstentions



(a) Abstention degradation (b) s1.1 on reasoning data

- Left: blue is the instruction-tuned base model; red is its version tuned for reasoning.
- Abstention recall is the fraction of needed abstentions taken. Higher is better; it falls for these reasoning models.
- Right: compare s1.1-32B with its Qwen2.5-32B base. Green shows increased answer accuracy; red shows decreased abstention recall.
- Better answer accuracy does not necessarily mean better recognition of when to abstain.

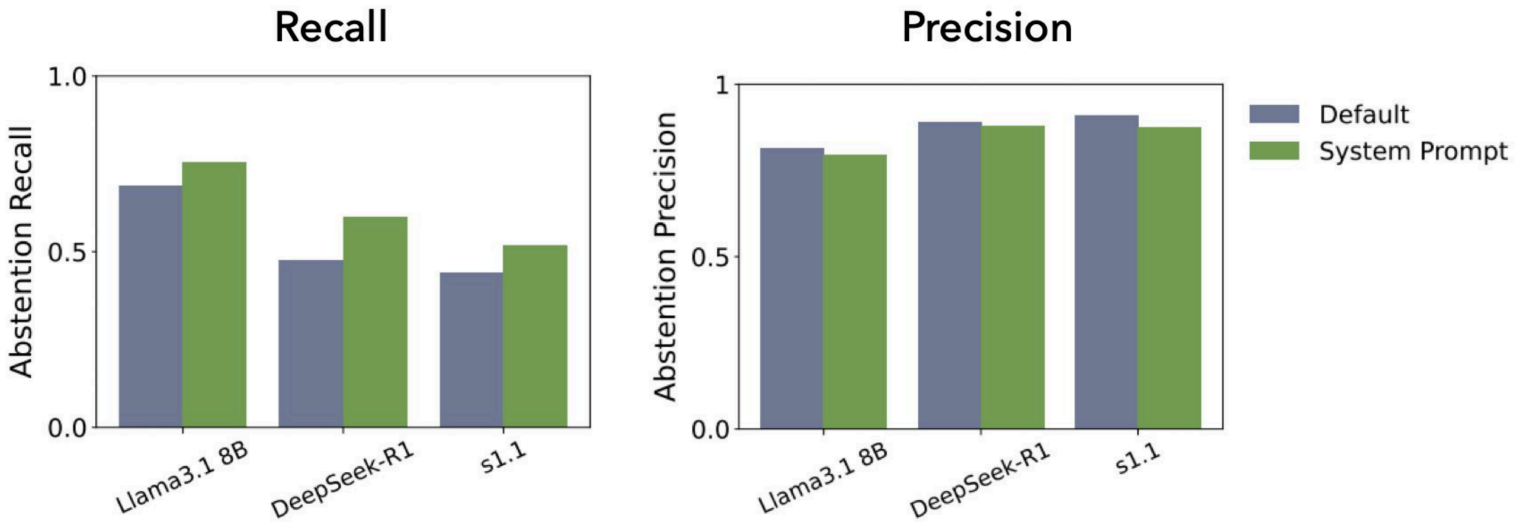
Better answer accuracy can accompany worse abstention



(a) Abstention degradation (b) s1.1 on reasoning data

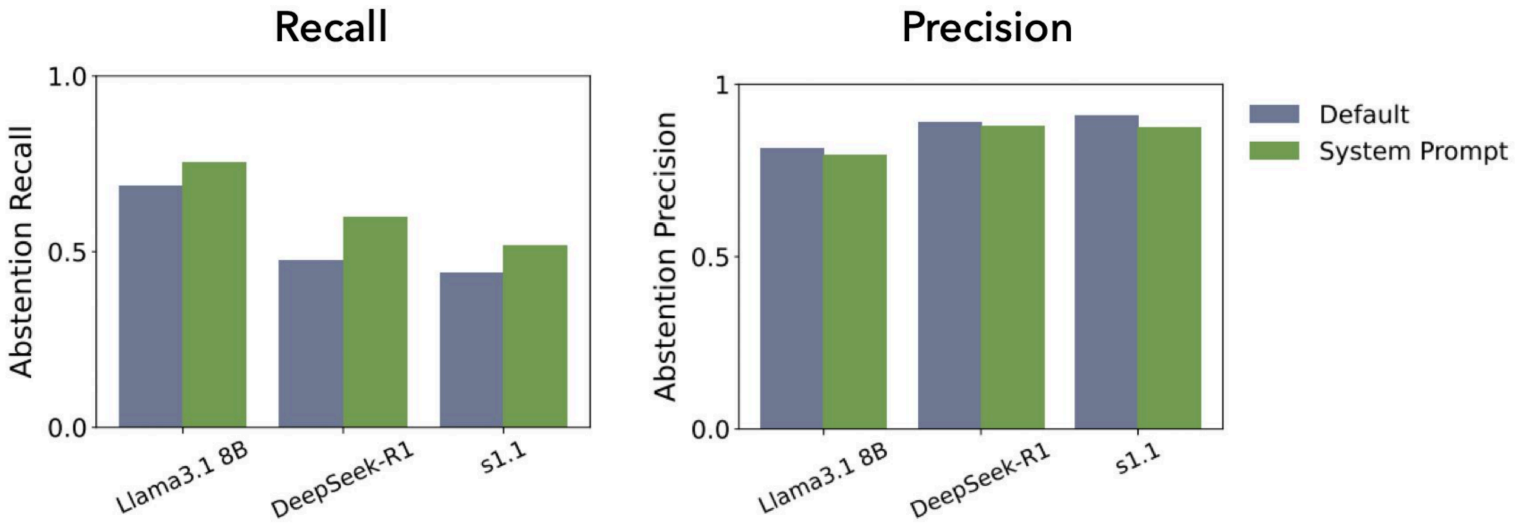
- Right: compare s1.1-32B with Qwen2.5-32B on reasoning datasets.
- Green shows accuracy gains; red shows abstention-recall declines.
- These tested models answer better while recognizing unanswerable questions less often.

Discouraging guessing increases abstention recall



- An added system instruction discourages guessing.
- Compare default responses in blue with instructed responses in green for three models.
- Left: abstention recall rises for all three models.

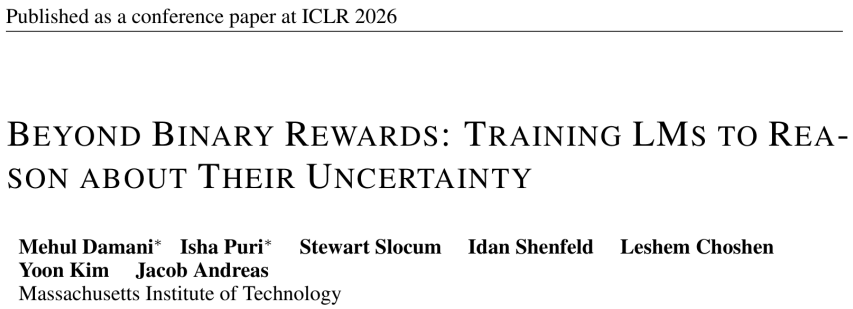
Check precision before endorsing more abstention



- Right: precision measures the fraction of produced abstentions that were warranted.
- The work reports no significant precision decline in this comparison.
- More abstention helps when it avoids unsupported answers without withholding useful answers unnecessarily.

How can reasoning models answer accurately and report reliable confidence?

Damani et al. · ICLR 2026



Published as a conference paper at ICLR 2026

BEYOND BINARY REWARDS: TRAINING LMS TO REASON ABOUT THEIR UNCERTAINTY

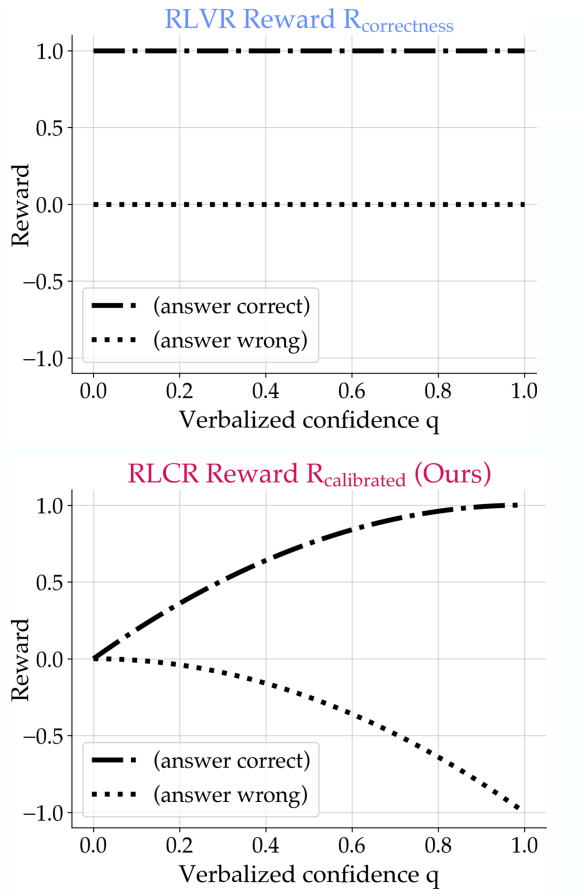
Mehul Damani* Isha Puri* Stewart Slocum Idan Shenfeld Leshem Choshen
Yoon Kim Jacob Andreas
Massachusetts Institute of Technology

- Train the model to answer accurately and communicate reliable confidence.
- Produce one answer and its estimated probability of being correct; 0.8 means an 80% chance.
- This confidence concerns the whole answer, rather than each claim separately.
- Use training questions with known correct answers to evaluate the model's output.
- Compare reinforcement learning with verifiable rewards, or RLVR, with reinforcement learning with calibration rewards, or RLCR.

Add reliable confidence to ordinary RLVR

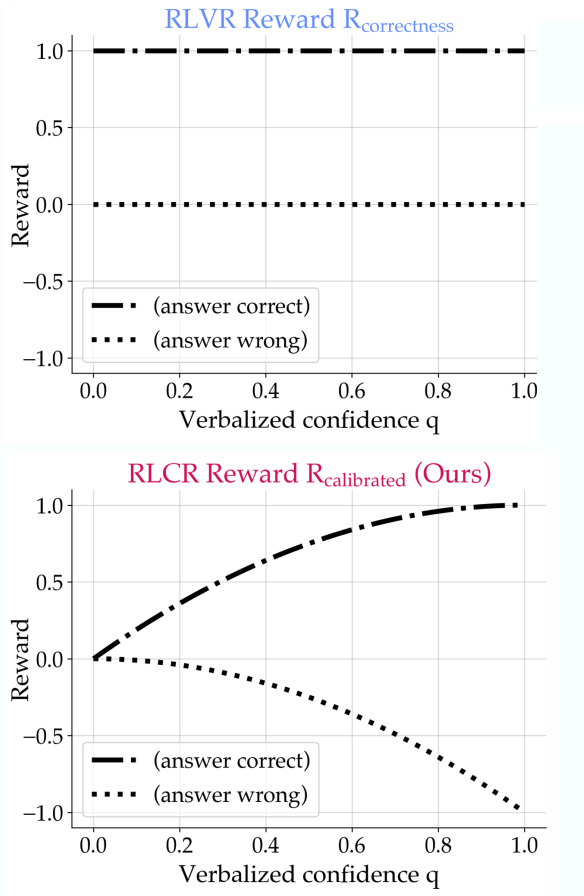
- Check the generated answer against a known correct answer.
- RLVR gives a reward of 1 for a correct answer and 0 for a wrong answer.
- It ignores stated confidence, so better accuracy alone can leave confidence unreliable.
- Across many answers given 80% confidence, about 80% should be correct.
- RLCR generates an answer, assesses uncertainty, and reports confidence.
- It keeps the correctness reward and penalizes disagreement between confidence and the observed outcome.

Read the confidence-reward curves



- The plots show stated confidence horizontally and reward vertically; dashed curves mean correct answers and dotted curves mean wrong answers.
- Top: the RLVR reward stays flat as confidence changes.
- Bottom: RLCR subtracts a squared confidence-error penalty.

Why the reward encourages honest confidence



In the equation, y is 1 for a correct answer and 0 for a wrong answer; q is stated confidence and R is reward.

$$R = y - (q - y)^2$$

Correct at q = 1: R = 1. Wrong at q = 1: R = –1. Wrong at q = 0: R = 0.

- The model chooses confidence before it knows the outcome.
- Expected reward encourages reporting the probability of being correct.
- This incentive requires empirical calibration checks after training.

A low-confidence answer can still be wrong

(a) Example Output

<think> The question asks for the song with which Lulu represented the UK in the 1969 Eurovision Song Contest. […] I need to recall the specific song that Lulu performed for the UK in 1969. </think>
<answer> To Sir With Love </answer>
<analysis> There is a high level of uncertainty in this answer because while Lulu did represent the UK in the 1969 Eurovision Song Contest, the specific song she performed is not widely known or easily recalled. […] but without more specific information, it is difficult to pinpoint the exact song.} </analysis>
<confidence> 0.3 </confidence>

- The source answer names "To Sir With Love" as Lulu's 1969 Eurovision song and reports confidence 0.3.
- The answer is wrong; the song was "Boom Bang-a-Bang."
- Low confidence gives a useful uncertainty signal but does not correct the factual claim.
- One example cannot establish calibration.
- Across many answers assigned similar confidence, that confidence should match the frequency of correctness.

Read the confidence and accuracy measures

(a) Models Trained on HotpotQA

Method	HotpotQA				O.O.D. Averaged			
	Acc. (↑)	AUROC (↑)	Brier (↓)	ECE (↓)	Acc. (↑)	AUROC (↑)	Brier (↓)	ECE (↓)
Base	39.7%	0.54	0.53	0.53	53.3%	0.54	0.41	0.40
RLVR		0.50	0.37	0.37		0.50	0.46	0.46
RLVR + BCE Classifier		0.66	0.22	0.07		0.58	0.27	0.24
RLVR + Brier Classifier	**63.0%**	0.65	0.22	0.09	53.9%	0.60	0.32	0.33
RLVR + Probe		0.55	0.24	0.10		0.53	0.38	0.38
Answer Probability		**0.72**	0.36	0.36		0.60	0.42	0.42
RLCR (**ours**)	62.1%	0.69	**0.21**	**0.03**	**56.2%**	**0.68**	**0.21**	**0.21**

- Accuracy is the proportion of correct answers; higher is better.
- Expected calibration error, or ECE, averages absolute confidence–correctness gaps across bins, weighted by bin size; lower is better.
- AUROC measures how well scores separate correct and wrong answers.
- Brier score averages $(q - y)^2$, where q is reported confidence and y is 1 for a correct answer and 0 for a wrong answer. Lower is better; confident mistakes receive larger penalties.

Confidence improves while factual errors remain

(a) Models Trained on HotpotQA

Method	HotpotQA				O.O.D. Averaged			
	Acc. (↑)	AUROC (↑)	Brier (↓)	ECE (↓)	Acc. (↑)	AUROC (↑)	Brier (↓)	ECE (↓)
Base	39.7%	0.54	0.53	0.53	53.3%	0.54	0.41	0.40
RLVR		0.50	0.37	0.37		0.50	0.46	0.46
RLVR + BCE Classifier		0.66	0.22	0.07		0.58	0.27	0.24
RLVR + Brier Classifier	**63.0%**	0.65	0.22	0.09	53.9%	0.60	0.32	0.33
RLVR + Probe		0.55	0.24	0.10		0.53	0.38	0.38
Answer Probability		**0.72**	0.36	0.36		0.60	0.42	0.42
RLCR (**ours**)	62.1%	0.69	**0.21**	**0.03**	**56.2%**	**0.68**	**0.21**	**0.21**

- On HotpotQA, RLVR to RLCR lowers ECE from 0.37 to 0.03; accuracy decreases slightly from 63.0% to 62.1%.
- Across six other tasks, mean ECE falls from 0.46 to 0.21 and accuracy rises from 53.9% to 56.2%.
- Confidence improves, while many answers remain wrong.
- Reliable confidence helps a user decide whether to trust, verify, or withhold an answer.

Choose an intervention for the missing information

- Assess candidate correctness or disagreement in meaning, then validate the rule for choosing answers.
- Ask for clarification when user intent is missing.
- Retrieve evidence when needed world knowledge is missing.
- Judge the final answer and the extra interaction or computation cost.
- Return a set, filter unsupported claims, or learn useful partial answers.
- Choose the method and evaluation target that match the missing information.

Evaluate what the user actually receives

- State the failure being measured: error among accepted answers differs from a candidate set missing every acceptable answer.
- Claim validity is the fraction of whole responses whose retained claims meet the stated loss limit; retention measures how much information survives.
- HALT correctness concerns produced fragments; completeness concerns desired content. Linguistic calibration measures the beliefs readers form.
- Abstention recall and precision concern warranted withholding; confidence calibration compares probabilities with outcomes.
- Count the costs of clarification, retrieval, and additional model calls.
- Aim for reliable decisions that preserve useful information.