Reliable Reasoning-Based Text Classification via Feature-Based Calibration and Conformal Risk Control

Anonymous ACL submission

Abstract

Large language models (LLMs) are increasingly used for text classification tasks that require semantic reasoning, such as peer review or harmful-comment analysis. For these tasks, accuracy is not enough: a deployable system also needs predictions with meaningful uncertainty and, where possible, formal coverage guarantees. Directly calibrating the LLM is difficult, as closed-source APIs hide token probabilities, and even when they are available the token-level event is a poor proxy for the label-level decision. We propose a two-stage framework that separates semantic reasoning from reliability modeling. The LLM answers a set of label-specific Yes/No sub-questions whose responses are encoded as a feature vector; a standard classifier maps these features to label-wise probabilities; a post-hoc calibrator refines them; and conformal risk control (CRC) converts the calibrated scores into prediction sets with target coverage. Across three reasoning-based benchmarks and two LLM backbones, the framework keeps $|\text{Cov} - 0.9|$ small on every cell, while direct LLM-confidence baselines either undercover or reach high coverage only by returning near-full sets. Among eight calibrators we evaluate, KCal combined with CRC reduces mean $|\text{SS} - 1|$ by 18.0% over the uncalibrated framework and lowers mean expected calibration error from 0.156 to 0.060 at matched coverage. The two components are complementary: CRC supplies the coverage guarantee that calibration alone cannot, and calibration sharpens the scores that CRC turns into informative sets.

1 Introduction

Many text classification tasks hinge on reasoning rather than surface cues. In peer review, online comments, and harmful-language annotation, the target label is rarely stated outright: it must be inferred from dispersed evidence, implied stance, and pragmatic context, and the same passage may support multiple defensible labels. LLMs handle this inference well because they cope with semantic variation that keyword- or embedding-based classifiers miss. But a label or a raw confidence score is not enough for downstream use; we also want predictions whose uncertainty is meaningful and, where possible, backed by a *formal guarantee*.

Directly calibrating an LLM here runs into two obstacles. *First*, the probabilistic interface is unreliable: many commercial APIs do not expose token-level probabilities, and when they do the values shift with prompt phrasing, decoding, and tokenization, so the token-level event is a poor proxy for the label-level event (Guo et al., 2017; Kull et al., 2017). *Second*, scalar probabilities are not the end goal. Calibration tries to estimate $\Pr(Y{=}1 \mid X)$ at every score value, a hard pointwise problem; conformal risk control (CRC) (Vovk et al., 2005; Angelopoulos and Bates, 2023) asks something weaker, at what threshold the score retains the true label often enough, and so admits a distribution-free finite-sample guarantee even when the underlying score is miscalibrated. Calibration sharpens the scalar, CRC turns it into a risk-controlled set, and a reliable system needs both.

We therefore decouple semantic reasoning from reliability modeling into two stages. In the first stage, the LLM acts as a structured feature extractor: for each candidate label it answers a fixed set of Yes/No sub-questions, whose responses are encoded as a symbolic feature vector. In the second stage, a classical classifier, e.g. logistic regression (Bishop and Nasrabadi, 2006), maps this vector to a conditional probability, a post-hoc calibrator refines it, and CRC converts the calibrated score into a prediction set at the target coverage.

The first stage is what makes the second tractable: aggregating the LLM's evidence into a fixed-dimensional feature vector restores the regime classical post-hoc calibrators (Platt, 1999; Zadrozny and Elkan, 2002; Kull et al., 2017) were

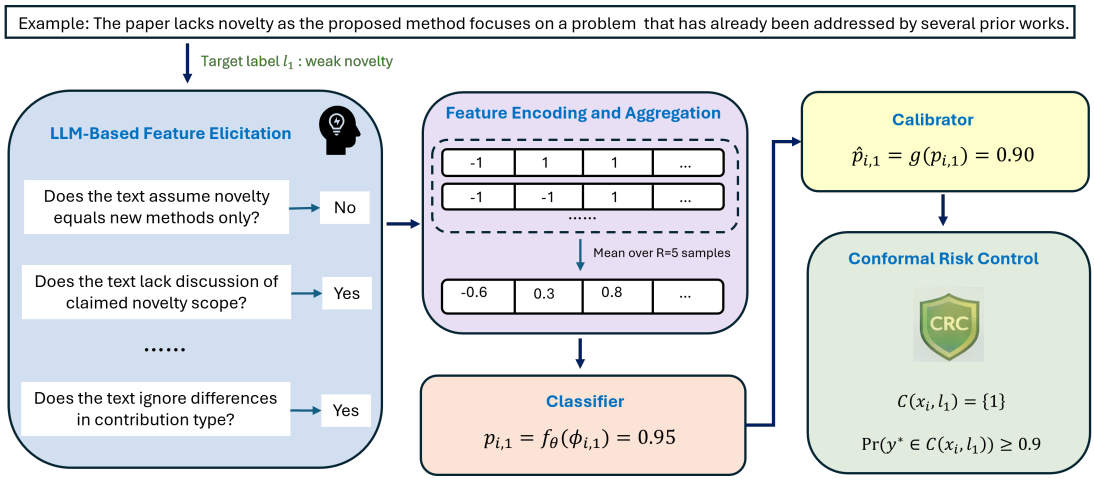

Figure 1: **Overview of the proposed framework**, illustrated on a LazyReview review with target label *weak novelty*. (1) **LLM-based feature elicitation**: the LLM answers K=10 label-specific Yes/No sub-questions that probe semantic evidence for the target label, repeated R=5 times under stochastic decoding. (2) **Feature encoding and aggregation**: Yes/No answers are encoded as ± 1 and aggregated across the R samples into a feature vector $\phi_{i,\ell}$. (3) **Classification and calibration**: a standard classifier maps $\phi_{i,\ell}$ to a label-wise probability $p_{i,\ell}$, which a post-hoc calibrator refines to $\hat{p}_{i,\ell}$. (4) **Conformal risk control**: CRC converts $\hat{p}_{i,\ell}$ into a prediction set $\mathcal{C}(x_i, \ell) \subseteq \{0, 1\}$ with marginal coverage $\Pr\big(y^* \in \mathcal{C}(x_i, \ell)\big) \geq 1 - \alpha$, where α=0.1 in our experiments.

designed for. The calibrated probabilities then serve as inputs to CRC, enabling smaller prediction sets at the same coverage.

In summary, our key contributions are three-fold:

- **A two-stage reframing for reasoning-based classification.** We recast the problem as (i) LLM-driven elicitation onto a fixed, label-conditional feature space, and (ii) a standard reliability layer (classifier, calibrator, CRC) on that space.

- **Calibration-CRC composition.** We show that direct LLM confidence covers inconsistently and calibration alone has no coverage guarantee, while calibration composed with CRC delivers both well-behaved probabilities and risk-controlled sets, with calibration sharpness translating directly into smaller sets at matched coverage.

- **Empirical evidence.** Across two backbones (Llama-3.1-8B, Phi-4-14B) and three datasets, random forest with KCal and CRC achieves the best calibration-efficiency trade-off among eight calibrators evaluated inside the framework, with mean Expected Calibration Error (ECE) 0.06 with tiny deviation from the ideal prediction set size at near-target coverage 0.9, while LLM-confidence baselines either under-cover or return near-full sets.

2 Background

Conformal prediction (Vovk et al., 2005; Angelopoulos and Bates, 2023) is a finite-sample, distribution-free procedure turning model scores into set-valued predictions with controlled error. We use split conformal prediction; conformal risk control (CRC) (Angelopoulos et al., 2024) generalizes coverage to bounded losses and reduces to split CP under the miscoverage loss $\mathbb{1}\{Y \notin \mathcal{C}(X)\}$ used here. We keep the label "CRC" throughout for consistency with the title and figure.

Setup and procedure. For a K-class problem with input X, label Y, and per-label nonconformity scores $s_k(X) \in \mathbb{R}$ (larger = less consistent with label k), split conformal calibrates a threshold on a held-out set $\{(X_i, Y_i)\}_{i=1}^n$ exchangeable with test data: (i) compute $s_i := s_{Y_i}(X_i)$; (ii) set $\hat{q}$ to the $\lceil (n+1)(1-\alpha) \rceil / n$ empirical quantile of $\{s_i\}$; (iii) at test time return $\mathcal{C}(X) = \{k : s_k(X) \leq \hat{q}\}$. Under exchangeability,

$$\mathbb{E}\big[\mathbb{1}\{Y_{\text{test}} \notin \mathcal{C}(X_{\text{test}})\}\big] \leq \alpha, \qquad (1)$$

i.e. $\Pr(Y \in \mathcal{C}(X)) \geq 1 - \alpha$. We use $\alpha = 0.1$ throughout.

Why CRC complements calibration. The guarantee is distribution-free and model-agnostic — the base model enters only through its scores. Calibration shapes the score distribution but gives no set-

level guarantee; CRC supplies the guarantee but does not sharpen scalars. They are complementary.

3 Methodology

Our method is a four-stage pipeline per (text, label) pair: (1) *Probing* — the LLM answers label-specific Yes/No sub-questions across repeated samples; (2) *Aggregation* — the samples are summarized into a feature vector; (3) *Classification* — a standard classifier maps the vector to a label-wise probability; (4) *Calibration + CRC* — a post-hoc calibrator refines the probability, and split CRC yields a prediction set at the target coverage.

Why a two-stage decoupled design. End-to-end fine-tuning needs more labeled data than reasoning-heavy tasks supply and ties reliability to opaque internals. We give the LLM a narrower job — targeted Yes/No questions — and hand probability estimation, calibration, and set construction to standard components on a feature space we control. The framework needs no access to LLM token probabilities (so it works with closed-source APIs); traditional calibrators and CRC keep their small-sample guarantees on a fixed feature space.

Why probing features should be easier to calibrate than LLM-internal scores (informal argument). Post-hoc calibrators rely on (s, Y) having a stable joint distribution on which a small held-out sample can pin down $g(s) \approx \Pr(Y = 1 \mid s)$. Token-level LLM confidences do not cleanly meet this in the reasoning regime: the same label evidence maps to different surface tokens under prompt phrasing, temperature, and decoding, so $\Pr(Y \mid s)$ shifts with nuisance variables outside the calibration set. By contrast, $\phi \in [-1, +1]^K$ is (i) bounded and fixed-dimensional, (ii) label-conditional (one ϕ per (x, ℓ)), and (iii) R-sample-averaged, contracting prompt-conditional noise. Under the working assumption $\Pr(Y_\ell = 1 \mid x) \approx \Pr(Y_\ell = 1 \mid \phi(x, \ell))$, calibration on ϕ runs in its intended regime; on s_{token} it does not. We make no formal sufficiency claim — this is an empirical hypothesis that §4 tests, with §4.4 and §A.5 probing its robustness.

3.1 Problem Setup

We frame the task as label-wise binary prediction. Let $x \in \mathcal{X}$ be an input text and $\mathcal{L} = \{1, \ldots, L\}$ the label set; each x_i carries a multi-label vector $y_i = (y_{i,1}, \ldots, y_{i,L}) \in \{0, 1\}^L$ with $y_{i,\ell} = 1$ if label ℓ applies. We decompose the task into independent binary problems over (x_i, ℓ) pairs (binary relevance) for a uniform interface and per-label reliability evaluation. For each pair we target two reliability forms: **probabilistic reliability** — the predicted score behaves as a meaningful estimate of $\Pr(y_{i,\ell} = 1)$ — and **decision reliability** — the prediction set meets a user-specified risk level while staying as small as possible.

3.2 LLM-based Semantic Probing

A label-level judgment is hard for an LLM to answer reliably as a single probability, but it can be decomposed into narrower checks the LLM handles well. For each label we ask K targeted Yes/No sub-questions probing facets of label-relevant evidence (stance, claim, aspect). Binary answers reduce free-form generation variance and encode directly as numerical features. To capture decoding-induced uncertainty, we repeat each query R times; cross-sample disagreement marks ambiguous evidence the downstream classifier can weigh. In our experiments, we set $R = 5$.

Formally, for input x_i, label ℓ, sub-question $k \in \{1, \ldots, K\}$, and sample $r \in \{1, \ldots, R\}$, the LLM returns $a^{(r)}_{i,\ell,k} \in \{\texttt{yes}, \texttt{no}\}$, encoded as $z^{(r)}_{i,\ell,k} = \pm 1$, giving $\mathbf{z}^{(r)}_{i,\ell} \in \{-1, +1\}^K$.

Sub-question source. Sub-questions are generated once per label by prompting an auxiliary LLM with a fixed template (Gemini by default; see §A.5 for a GPT-5 comparison and §A.8 for the template). This is transferable across tasks without per-label authoring, but the feature space inherits the auxiliary LLM's and the template's blind spots: evidence facets the auxiliary LLM does not surface cannot be recovered downstream. We treat sub-question quality as a first-class design surface (§A.5); tighter coverage and non-redundancy objectives are left to future work.

3.3 Feature Aggregation

The classifier needs a fixed-dimensional input. Our default is the sample mean $\bar{\mathbf{z}}_{i,\ell} = \frac{1}{R} \sum_r \mathbf{z}^{(r)}_{i,\ell} \in [-1, +1]^K$: unambiguous sub-questions saturate near ± 1, contested ones near zero. The mean cannot distinguish a genuinely uncertain sub-question (high variance) from one consistently answered near the middle, so we also consider $\mathbf{h}_{i,\ell} = [\bar{\mathbf{z}}_{i,\ell}; \mathbf{s}_{i,\ell}]$, where $\mathbf{s}_{i,\ell}$ is the per-sub-question standard deviation. We compare both in §4.4 and write $\phi_{i,\ell}$ for the aggregated feature vector throughout.

3.4 Probabilistic Classification

Given a feature matrix with 10 or 20 features, we employ a classical statistical classifier rather than a deep learning model, because logistic regression and random forests have ample capacity without overfitting and produce well-defined scalar probabilities that interact cleanly with post-hoc calibration. The classifier f_θ produces $p_{i,\ell} = f_\theta(\phi_{i,\ell}) \in [0, 1]$, a preliminary estimate of $\Pr(y_{i,\ell} = 1)$. We ablate over logistic regression, random forest, and a small MLP in §4.3.

3.5 Calibration with KCal

Classifier outputs on small datasets are typically miscalibrated, so we add post-hoc calibration on a held-out set. Standard calibrators — Platt scaling (Platt, 1999), Beta calibration (Kull et al., 2017), isotonic regression (Zadrozny and Elkan, 2002) — operate on the scalar score and discard the feature space. Because our LLM-elicited features are structured semantic evidence rather than noise around a probability, our default is KCal (Lin et al., 2023), which performs kernel-based density estimation directly on ϕ:

$$\hat{p}_{i,\ell} = \frac{\sum_j K_h(\phi_{i,\ell}, \phi_j)\, y_j}{\sum_j K_h(\phi_{i,\ell}, \phi_j)}, \tag{2}$$

where K_h is an RBF kernel with bandwidth h from cross-validation on $\{(\phi_j, y_j)\}$. Because KCal reads ϕ directly, it bypasses the classifier output, so when comparing classifier choices we use isotonic regression instead (§4.3).

3.6 Conformal Risk Control

Calibration sharpens the scalar but gives no decision-level guarantee. We add CRC on top to turn $\hat{p}_{i,\ell}$ into a prediction set with marginal coverage $1 - \alpha$. CRC is applied independently per (x_i, ℓ) pair over $\{0, 1\}$ using nonconformity score $s(x_i, \ell, y) = 1 - p_y(x_i, \ell)$ with $p_1(x_i, \ell) = \hat{p}_{i,\ell}$ and $p_0(x_i, \ell) = 1 - \hat{p}_{i,\ell}$. The threshold $\hat{q}_\ell$ is calibrated per label on a held-out set following §2; the coverage guarantee is therefore *marginal per label*, $\Pr(y_{i,\ell} \in \mathcal{C}(x_i, \ell)) \geq 1 - \alpha$, not joint over the multi-label vector. The test-time prediction set is $\mathcal{C}_{\hat{q}_\ell}(x_i, \ell) = \{y \in \{0, 1\} : s(x_i, \ell, y) \leq \hat{q}_\ell\}$. Set size $\in \{0, 1, 2\}$: size 1 commits to one label state, size 2 retains both, size 0 is by construction a miscoverage event and occurs at rate $\leq \alpha$ under exchangeability.

4 Experiments

Our experiments answer two research questions.

- **RQ1:** Does the two-stage approach yield more reliable probabilities and prediction sets than direct calibration of LLMs for reasoning-based text classification?
- **RQ2:** How well does calibration at the second stage affect CRC efficiency under matched risk targets?

We find that the framework consistently keeps coverage close to target while LLM-confidence baselines either undercover or hedge with near-full sets (RQ1), and that calibration tightens CRC sets at matched coverage, with KCal reducing mean deviation from ideal set size by 18.0% over competitive baselines (RQ2).

4.1 Experimental Setup

4.1.1 Datasets

We evaluate on three reasoning-based datasets with different label structures and positivity rates. Full preprocessing, label lists, and positivity counts are in Appendix A.1.

LazyReview (Purkayastha et al., 2025) contains peer reviews with multi-label review-issue tags (weak novelty, unclear methodology, etc.). We keep labels with ≥ 10 positive examples.

D3Code (Davani et al., 2024) is a disagreement-aware benchmark for subjective offensiveness detection. We use the strongly-imbalanced moral subset.

MHS (Sachdeva et al., 2022) comprises social media comments annotated for hate and abusive language. We construct a balanced subset by filtering on annotator-rating reliability and resampling positives per label.

4.1.2 Evaluation Metrics

We report seven macro-averaged metrics in the following three categories.

Predictive discrimination. **AUROC** measures ranking between positives and negatives; **AUPRC** is more informative under imbalance. Higher is better.

Calibration quality. Brier, **NLL**, and **ECE** (Guo et al., 2017) probe squared error to the binary outcome, log-likelihood of the true label, and the gap between predicted confidence and empirical accuracy. Lower is better.

Conformal efficiency. Let coverage (COV) be the fraction of test instances whose prediction set contains the true label, and set size (SS) be mean set cardinality. Since CRC is applied per (x, ℓ) over $\{0, 1\}$, $|\mathcal{C}(x)| \in \{0, 1, 2\}$. To compare methods on a single lower-is-better axis, we report the absolute deviations from their targets: $|\mathbf{COV} - \mathbf{0.9}|$ measures the deviation from the $1 - \alpha = 0.9$ coverage target (smaller means closer to the nominal guarantee, capturing both over- and under-coverage), and $|\mathbf{SS} - \mathbf{1}|$ measures distance from the ideal set size (smaller means more informative sets, capturing both empty sets caused by miscoverage and full sets caused by hedging). Lower is better for both.

4.1.3 Baselines

Calibration variants of our framework. We plug eight post-hoc calibrators into the calibration slot: Temperature Scaling (TS) (Guo et al., 2017), Platt (Platt, 1999), isotonic (Zadrozny and Elkan, 2002), Beta (Kull et al., 2017), spline (Gupta et al., 2021), CCal (Lyu et al., 2025), MCal (Detommaso et al., 2024), and KCal (Lin et al., 2023). All except KCal consume the classifier's scalar output; KCal operates on ϕ (§3.5). **Vanilla** (no calibrator) isolates calibration's contribution.

Direct LLM-confidence baselines. APRICOT (Ulmer et al., 2024) (confidence from generated outputs), ConfTuner (Li et al., 2025) (LLM trained to verbalize confidence), and CoCoA (Vashurin et al., 2025) (confidence $\times$ generation consistency; adapted to binary via repeated Yes/No sampling). All three are run zero-shot per their original design. Our framework instead trains on 80% of each dataset and splits the remaining 20% $50/50$ into calibration and test, so it is evaluated on a strictly smaller test pool.

A note on the supervised-vs-zero-shot gap. The comparison is not symmetric in supervision: we consume labeled training data, the LLM-confidence baselines do not. We use the baselines as proposed (zero-shot); in-domain fine-tuning would change the methods. The question we address is the paradigm-level one — can a supervised lightweight pipeline outperform a zero-shot LLM-internal estimator? — and a same-supervision comparison (few-shot prompting or a fine-tuned LLM with calibrated outputs) is a natural extension we treat as future work.

Conformal baselines. APS (Romano et al., 2020) and RAPS (Angelopoulos et al., 2021) replace our default CRC step. APS/RAPS were proposed for multi-class; we adapt them per label under binary relevance.

Implementation details, hyperparameters, and the calibration-split protocol are in App. A.2.

4.2 Main Performance Comparison

Table **??**, **??** and **??** compare LLM-confidence baselines (top block), the eight calibrators inside our framework (middle and bottom blocks for the two backbones), and the conformal-score variants APS and RAPS, across three datasets.

RQ1: framework vs. direct LLM confidence. CRC-based methods cluster at small $|\text{Cov} - 0.9|$ across all six (backbone, dataset) cells; the three LLM-confidence baselines do not. CoCoA undercovers on D3Code/MHS ($|\text{Cov} - 0.9|$ 0.167/0.059 on Llama, both below the target); ConfTuner overcovers everywhere ($|\text{Cov} - 0.9| \leq 0.100$, always above 0.9) but only by returning near-full sets ($|\text{SS} - 1|$ 0.76–0.97); APRICOT lands in $|\text{Cov} - 0.9| \in [0.002, 0.026]$ with no controlling mechanism behind it. The split is paradigm-level: CRC fixes a threshold on the calibration split; baselines have no equivalent step. A sharper scalar does not by itself produce coverage at a chosen rate.

RQ2: calibration and CRC efficiency. CRC inherits the score shape, so sharper inputs yield smaller sets at the same target. Averaged over six cells, KCal + CRC reaches mean $|\text{SS} - 1| = 0.222$ vs. Vanilla's 0.270 — 18.0% reduction at matched coverage — and mean ECE drops from 0.156 to 0.060. KCal achieves the lowest mean ECE (0.060), Brier (0.089), and $|\text{SS} - 1|$ (0.222) among the eight calibrators while keeping $|\text{Cov} - 0.9|$ in $[0.007, 0.055]$ everywhere. Isotonic and Platt, the closest competitors on probability quality, each collapse on coverage in at least one cell (Isotonic to $|\text{Cov} - 0.9| = 0.204$ on Llama/D3Code; Platt to $|\text{Cov} - 0.9| = 0.139$ on Phi/D3Code), so their lower $|\text{SS} - 1|$ there is not comparable. We adopt RandomForest + KCal + CRC as default.

Discussion. Calibration and CRC are complementary: CRC supplies the coverage guarantee

Table 1: Multi-seed results on LazyReview. Values are mean±std over five seeds, with standard deviations shown in a smaller font. Bold indicates the best value within each block.

Method	AUROC↑	AUPRC↑	Brier↓	NLL↓	ECE↓	$\lvert\mathrm{Cov}-0.9\rvert$ ↓	$\lvert\mathrm{SS}-1\rvert$ ↓
APRICOT	**0.553±0.000**	0.145±0.000	0.160±0.000	**0.509±0.000**	0.231±0.000	**0.026±0.000**	0.500±0.000
CoCoA	0.548±0.000	**0.220±0.000**	**0.124±0.000**	1.532±0.000	**0.120±0.000**	0.032±0.000	**0.065±0.000**
ConfTuner	0.493±0.000	0.124±0.000	0.548±0.000	1.393±0.000	0.656±0.000	0.100±0.000	0.972±0.000
Llama-3.1-8B							
Vanilla	0.535±0.017	0.200±0.013	0.121±0.003	0.438±0.012	0.110±0.006	0.010±0.007	0.171±0.023
Beta	0.507±0.016	0.194±0.016	0.118±0.006	0.587±0.113	0.089±0.016	0.010±0.008	0.157±0.008
Isotonic	0.530±0.015	0.163±0.015	0.117±0.003	0.721±0.124	0.080±0.007	0.037±0.036	0.149±0.074
Platt	0.523±0.026	0.198±0.015	0.108±0.002	0.385±0.009	0.072±0.007	0.014±0.008	0.150±0.043
TS	0.535±0.017	0.200±0.013	0.120±0.001	0.438±0.011	0.109±0.010	0.010±0.007	0.171±0.023
Spline	**0.546±0.061**	0.212±0.052	0.111±0.008	0.531±0.099	0.092±0.022	0.015±0.010	0.124±0.036
CCal	0.515±0.004	0.141±0.010	0.123±0.001	1.277±0.099	0.118±0.003	0.054±0.010	0.541±0.096
MCal	0.523±0.014	0.200±0.015	0.119±0.004	0.445±0.016	0.104±0.012	**0.006±0.004**	0.182±0.036
KCal	0.534±0.049	**0.228±0.044**	**0.103±0.003**	**0.382±0.038**	**0.050±0.011**	0.026±0.021	0.046±0.031
APS	0.535±0.017	0.200±0.013	0.121±0.003	0.438±0.012	0.110±0.006	0.125±0.010	0.004±0.006
RAPS	0.535±0.017	0.200±0.013	0.121±0.003	0.438±0.012	0.110±0.006	0.129±0.013	**0.003±0.005**
Phi-4-14B							
Vanilla	0.612±0.014	0.248±0.007	0.134±0.002	0.432±0.005	0.154±0.005	0.027±0.008	0.255±0.029
Beta	0.469±0.048	0.172±0.012	0.123±0.001	0.947±0.166	0.109±0.004	0.015±0.013	0.291±0.063
Isotonic	0.585±0.020	0.172±0.009	0.109±0.003	0.604±0.097	0.071±0.004	0.042±0.032	0.198±0.076
Platt	**0.617±0.010**	**0.258±0.018**	**0.103±0.002**	**0.358±0.010**	**0.066±0.008**	0.025±0.008	0.177±0.024
TS	0.612±0.014	0.248±0.007	0.132±0.001	0.433±0.008	0.149±0.007	0.027±0.008	0.255±0.029
Spline	0.543±0.045	0.214±0.026	0.113±0.005	0.597±0.096	0.100±0.016	**0.012±0.006**	0.156±0.019
CCal	0.591±0.015	0.220±0.017	0.173±0.013	1.824±0.149	0.184±0.012	0.068±0.014	0.611±0.147
MCal	0.613±0.013	0.248±0.007	0.134±0.002	0.435±0.012	0.155±0.004	0.027±0.008	0.255±0.029
KCal	0.599±0.017	0.240±0.026	0.104±0.002	0.380±0.024	0.069±0.015	0.016±0.009	0.095±0.034
APS	0.612±0.014	0.248±0.007	0.134±0.002	0.432±0.005	0.154±0.005	0.170±0.017	0.001±0.002
RAPS	0.612±0.014	0.248±0.007	0.134±0.002	0.432±0.005	0.154±0.005	0.176±0.017	**0.001±0.001**

calibration alone cannot, and calibration sharpens the scores CRC turns into informative sets. APS and RAPS reach smaller $|\mathrm{SS}-1|$ only by trading away coverage (Llama/LazyReview $|\mathrm{Cov}-0.9|$ 0.120; Phi/MHS $|\mathrm{Cov}-0.9|$ 0.099), confirming that KCal's gain comes from better scores rather than a more aggressive set-construction rule.

4.3 Effect of the Classifier

The probability-estimation slot can be filled by any standard classifier. We compare logistic regression, a small MLP, and random forest, with isotonic regression as the calibrator (KCal short-circuits the classifier, so a classifier-output-consuming calibrator is needed for a non-vacuous comparison (§3.5).

Random forest is the most consistent default. Averaged across the six cells, RF achieves the best mean AUROC (0.629), Brier (0.094), NLL (0.526), and ECE (0.070), losing only mean AUPRC narrowly to logistic regression (0.335 vs. 0.344). RF's advantage is clearest on the imbalanced D3Code, where bagging and bootstrap subsampling handle skewed positivity more gracefully than parametric alternatives. We adopt RF as the default.

Note on the final pipeline. This comparison uses isotonic regression as the calibrator. When KCal — our default — is used, it reads ϕ directly (§3.5), so classifier choice has limited impact on the final pipeline. We still recommend RF: (i) it is the most robust classifier in the no-KCal regime, useful for variants that swap KCal for a classifier-output calibrator; (ii) the classifier scalar feeds the discrimination metrics (AUROC, AUPRC) we report alongside calibration, so a stronger classifier sharpens the diagnostic picture.

4.4 Effect of Feature Aggregation

The simpler mean $\bar{\mathbf{z}}$ is the more reliable default. Across the six cells, $\bar{\mathbf{z}}$ wins outright on 19 of 36 (column, cell) entries vs. 13 for $[\bar{\mathbf{z}}; \mathbf{s}]$, and the advantage concentrates on metrics the framework targets — $\bar{\mathbf{z}}$ wins on Brier, NLL, or ECE in five of six cells, including the largest single ECE gap on Llama-MHS (0.092 vs. 0.104). $[\bar{\mathbf{z}}; \mathbf{s}]$ produces real

	D3Code						LazyReview						MHS					
	AUROC↑	**AUPRC↑**	**Brier↓**	**NLL↓**	**ECE↓**	**|SS−1|↓**	**AUROC↑**	**AUPRC↑**	**Brier↓**	**NLL↓**	**ECE↓**	**|SS−1|↓**	**AUROC↑**	**AUPRC↑**	**Brier↓**	**NLL↓**	**ECE↓**	**|SS−1|↓**
Llama-3.1-8B																		
LogisticRegression	0.707	**0.120**	**0.034**	0.198	0.023	0.112	0.509	0.162	0.119	0.562	0.084	0.266	0.666	**0.747**	0.153	1.071	0.164	**0.504**
MLP	0.674	0.095	0.035	0.185	**0.017**	**0.104**	**0.561**	0.167	**0.111**	**0.487**	0.075	0.345	0.661	0.735	0.133	0.876	**0.129**	0.546
RandomForest	**0.734**	0.113	**0.034**	**0.161**	**0.017**	0.286	0.549	**0.170**	0.119	0.606	**0.072**	**0.113**	**0.669**	0.746	**0.124**	**0.664**	0.130	0.562
Phi-4-14B																		
LogisticRegression	0.679	**0.118**	**0.035**	**0.169**	0.027	0.037	**0.602**	**0.194**	0.114	0.680	0.102	0.137	0.559	**0.720**	0.152	1.151	0.132	**0.561**
MLP	0.675	0.078	**0.035**	0.191	0.016	0.035	0.534	0.168	**0.111**	**0.491**	0.080	**0.032**	0.544	0.717	0.142	0.969	0.116	0.658
RandomForest	**0.701**	0.086	**0.035**	0.209	**0.014**	**0.000**	0.556	0.179	**0.111**	0.736	**0.070**	0.090	**0.566**	0.717	**0.141**	**0.781**	**0.115**	0.631

Table 2: Effect of the classifier within our framework, with isotonic regression as the calibrator. Best per column within each backbone in **bold**; $|\mathrm{SS}-1|$ is bolded as smallest (closest to the ideal singleton). Values rounded to three decimal places; tied displayed values are bolded together.

		D3Code						LazyReview						MHS					
Features	**Backbone**	**AUROC↑**	**AUPRC↑**	**Brier↓**	**NLL↓**	**ECE↓**	**|SS−1|↓**	**AUROC↑**	**AUPRC↑**	**Brier↓**	**NLL↓**	**ECE↓**	**|SS−1|↓**	**AUROC↑**	**AUPRC↑**	**Brier↓**	**NLL↓**	**ECE↓**	**|SS−1|↓**
$\bar{\mathbf{z}}$ (10D)	Llama	**0.618**	**0.151**	**0.035**	0.157	0.011	**0.043**	0.499	**0.202**	**0.103**	0.434	**0.049**	0.090	0.566	0.738	**0.123**	**0.528**	**0.092**	0.490
	Phi	0.697	**0.202**	**0.035**	**0.156**	0.015	0.131	**0.604**	**0.237**	**0.106**	**0.403**	0.085	0.051	**0.667**	**0.766**	**0.129**	**0.493**	0.109	0.526
$[\bar{\mathbf{z}}; \mathbf{s}]$ (20D)	Llama	0.612	0.121	**0.035**	**0.156**	**0.009**	0.061	**0.505**	0.197	**0.103**	**0.407**	0.051	**0.072**	**0.636**	**0.766**	0.125	0.536	0.104	**0.464**
	Phi	**0.731**	0.180	**0.035**	0.157	**0.014**	**0.053**	0.589	0.227	0.107	0.419	**0.084**	**0.021**	0.665	0.763	**0.129**	0.494	**0.099**	**0.492**

Table 3: Effect of feature aggregation. Within each backbone, the better value per column between mean only ($\bar{\mathbf{z}}$, 10-D) and mean concatenated with standard deviation ($[\bar{\mathbf{z}}; \mathbf{s}]$, 20-D) is in **bold**. $|\mathrm{SS}-1|$ is bolded as smallest (closest to the ideal singleton).

AUROC gains on the two hardest cells (Llama-MHS +7.0pp, Phi-D3Code +3.4pp), but those ranking gains come with degraded calibration on the same cells. Combined with doubled dimensionality on small datasets, we keep $\bar{\mathbf{z}}$ as default and treat $[\bar{\mathbf{z}}; \mathbf{s}]$ as an optional extension.

Reconciling with the consistency analysis. §4.5 shows that per-sub-question dispersion carries label-informative signal on Phi-4-14B. The disappointment of $[\bar{\mathbf{z}}; \mathbf{s}]$ on calibration is not that variance is uninformative, but that naive concatenation does not exploit it well at the available sample size. A more principled aggregation — dispersion as an attention/gating signal rather than raw features — is a natural next step.

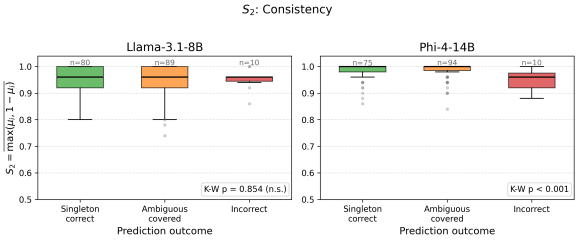

Figure 2: Sample consistency S (Eq. 3) versus CRC prediction outcome on MHS, pooled over nine binary labels. Each box shows the distribution of S within an outcome group on Llama-3.1-8B (left) and Phi-4-14B (right); n above each box is the group size. Phi shows a significant drop in consistency on Incorrect predictions (Kruskal–Wallis $p < 0.001$; Cohen's $d{=}{-}1.09$, large effect); Llama is uniformly high across groups ($p = 0.85$, n.s.).

4.5 Sample Consistency Analysis

LLM sampling variance carries information beyond the mean: when an instance's sub-question answers flip across samples, the disagreement plausibly signals unstable evidence. We test this by checking whether sample-level consistency tracks CRC outcomes, restricting to MHS where the 9-label structure pools sub-questions across labels for more reliable distributional estimates.

For each MHS test instance x we draw $N{=}10$ samples per sub-question $k \in \{1, \ldots, K\}$ and let $\mu_k(x) \in [0, 1]$ be the empirical yes frequency. The instance-level consistency score is

$$S(x) \;=\; \frac{1}{K}\sum_{k=1}^{K} \max\big(\mu_k(x),\; 1{-}\mu_k(x)\big) \;\in\; [0.5,\, 1], \tag{3}$$

i.e. the average dominant-answer frequency: $S(x){\approx}1$ means the LLM consistently picks one side, $S(x){\approx}0.5$ means it flips. We stratify by the relation between $\mathcal{C}(x)$ and true label y^*: *Singleton-correct* ($\mathcal{C}(x){=}\{y^*\}$), *Ambiguous* ($\mathcal{C}(x){=}\{0, 1\}$), *Incorrect* ($y^* \notin \mathcal{C}(x)$).

On Phi-4-14B, Incorrect predictions have significantly lower S than Singleton-correct ones (means 0.95 vs. 0.98, Kruskal–Wallis* $p < 0.001$, Cohen's $d{=}{-}1.09$): sampling spread tracks miscoverage. On Llama-3.1-8B the test is not significant ($p{=}0.85$), but Llama saturates near $S{=}0.96$ uniformly with only $n{=}10$ Incorrect instances, making the absence of signal hard to interpret. Phi

*We use Kruskal–Wallis because $S(x)$ is bounded and non-normal (Fig. 2); it assumes neither normality nor equal variance.

reveals its uncertainty through sampling spread; Llama does not. This is direct evidence that per-sub-question dispersion carries information beyond the mean, motivating $[\bar{\mathbf{z}}; \mathbf{s}]$ for backbones whose sampling actually varies. Three worked examples spanning Singleton-correct, Ambiguous, and Incorrect outcomes are given in App. A.7; they illustrate that miscoverage here looks more like borderline sub-questions tipping the aggregated ϕ" than probing in chaos.

5 Related Work

5.1 Probability Calibration

Classical post-hoc calibration methods include Platt scaling (Platt, 1999), isotonic regression (Zadrozny and Elkan, 2002), temperature scaling (Guo et al., 2017), and beta calibration (Kull et al., 2017), with extensions to spline (Gupta et al., 2021), kernel-based (Lin et al., 2023), and multi-calibration (Detommaso et al., 2024) formulations. These methods assume the base score is meaningfully aligned with the target prediction event. That breaks for LLMs: token probabilities and verbalized confidence are sensitive to prompting and decoding, and the token-level event is a poor proxy for the label-level event. We sidestep this by calibrating a classifier trained on structured LLM-elicited features.

5.2 Reliability Estimation for LLMs

A separate line estimates LLM reliability from the model's own behavior: APRICOT (Ulmer et al., 2024), ConfTuner (Li et al., 2025), CoCoA (Vashurin et al., 2025), CCal (Lyu et al., 2025), and broader work on whether models recognize what they do not know (Kadavath et al., 2022). All treat the LLM as the final probabilistic predictor. We use it only as a semantic feature extractor and place probability estimation, calibration, and set construction in components on those features, more reliable when the internal representations and model parameters of LLMs are unavailable or noisy.

5.3 Conformal Prediction and Risk Control

Conformal prediction provides distribution-free, finite-sample coverage guarantees (Vovk et al., 2005; Angelopoulos and Bates, 2023). APS (Romano et al., 2020) and RAPS (Angelopoulos et al., 2021) construct adaptive sets; conformal risk control (Bates et al., 2021; Angelopoulos et al., 2024) generalizes coverage to bounded losses; BRConformal (Örnbratt and Hallberg Szabadváry, 2025) applies label-wise conformal prediction under binary relevance. Xi et al. (2024) report that calibration can sometimes reduce conformal efficiency, raising the question of whether it helps when CRC is the goal. We answer this empirically for reasoning-based classification: calibration consistently improves CRC efficiency in our framework, with KCal reducing set size at matched coverage.

6 Conclusion

We present a framework for reasoning-based text classification that recasts the problem in two stages: an LLM-driven evidence-elicitation step mapping each $(\text{text}, \text{label})$ pair to a fixed feature vector of label-conditional Yes/No probing means, and a statistical-reliability layer (classifier + calibration + split CP) on that space. The framing matters more than any individual component: by replacing opaque, prompt-conditional LLM confidences with an observable, bounded, label-conditional feature space, classical calibrators and conformal procedures regain their working assumptions and their finite-sample guarantees, even under closed-source LLMs and small calibration sets.

Across three datasets and two backbones, the framework holds coverage near target on every cell while LLM-confidence baselines either undercover or hedge with near-full sets. Within the framework, KCal reduces mean set-size deviation by 18.0% and lowers mean ECE from 0.156 to 0.060, while CRC supplies the coverage guarantee. This is evidence for the effectiveness of the two-stage approach, not for any individual calibrator.

Limitations

The framework's effectiveness depends on sub-question quality: poor or incomplete questions leave a representation gap the classifier cannot recover. Sub-questions are generated once per label by prompting an auxiliary LLM (Gemini by default, §A.5) with a fixed template (§A.8) reused across labels and datasets, so the per-task human cost is listing the label set and inspecting outputs for obvious failures, not authoring questions. A self-supervised step in which the LLM iteratively proposes and refines its own sub-questions against held-out feedback is a natural extension.

Additionally, CRC guarantees validity but not efficiency: weakly aligned probabilities can still

yield large sets at the desired risk level.

Ethics Statement

We use three publicly available datasets and do not collect new personal data; the original creators report that identifying or private information has been removed or anonymized. D3Code and MHS contain abusive, hateful, or violent language reflecting harmful social biases. We use these datasets only for inference-time evaluation, not for training or fine-tuning. All data and models are appropriately licensed, and we adhered to ethical guidelines protecting individual privacy.

References

Anastasios N. Angelopoulos and Stephen Bates. 2023. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591.

Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. 2024. Conformal risk control. In *International Conference on Learning Representations*.

Anastasios N. Angelopoulos, Stephen Bates, Jitendra Malik, and Michael I. Jordan. 2021. Uncertainty sets for image classifiers using conformal prediction. In *International Conference on Learning Representations*.

Stephen Bates, Anastasios N. Angelopoulos, Lihua Lei, Jitendra Malik, and Michael I. Jordan. 2021. Distribution-free, risk-controlling prediction sets. *Journal of the ACM*, 68(6).

Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*, volume 4. Springer.

Aida Mostafazadeh Davani, Mark Díaz, Dylan Baker, and Vinodkumar Prabhakaran. 2024. D3CODE: Disentangling disagreements in data across cultures on offensiveness detection and evaluation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Gianluca Detommaso, Martin Andres Bertran, Riccardo Fogliato, and Aaron Roth. 2024. Multicalibration for confidence scoring in llms. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10624–10641.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, ICML, pages 1321–1330. PMLR.

Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. 2021. Calibration of neural networks using splines. In *International Conference on Learning Representations*.

Saurav Kadavath and 1 others. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Meelis Kull, Telmo Silva Filho, and Peter Flach. 2017. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, AISTATS, pages 623–631. PMLR.

Yibo Li, Miao Xiong, Jiaying Wu, and Bryan Hooi. 2025. Conftuner: Training large language models to express their confidence verbally. In *Advances in Neural Information Processing Systems*.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Taking a step back with KCal: Multi-class kernel-based calibration for deep neural networks. In *International Conference on Learning Representations*.

Qing Lyu, Kumar Shridhar, Chaitanya Malaviya, Li Zhang, Yanai Elazar, Niket Tandon, Marianna Apidianaki, Mrinmaya Sachan, and Chris Callison-Burch. 2025. Calibrating large language models with sample consistency. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 19260–19268.

John Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Alexander Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press.

Sukannya Purkayastha, Zhuang Li, Anne Lauscher, Lizhen Qu, and Iryna Gurevych. 2025. Lazyreview: A dataset for uncovering lazy thinking in nlp peer reviews. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3280–3308, Vienna, Austria. Association for Computational Linguistics.

Yaniv Romano, Matteo Sesia, and Emmanuel Candès. 2020. Classification with valid and adaptive coverage. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Pratik Sachdeva, Renata Barreto, Geoff Bacon, Alexander Sahn, Claudia von Vacano, and Chris Kennedy. 2022. The measuring hate speech corpus: Leveraging rasch measurement theory for data perspectivism. In *Proceedings of the 1st Workshop on Perspectivist Approaches to NLP @LREC2022*, pages 83–94, Marseille, France. European Language Resources Association.

Dennis Ulmer, Martin Gubri, Hwaran Lee, Sangdoo Yun, and Seong Joon Oh. 2024. Calibrating large

language models using their generations only. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Roman Vashurin and 1 others. 2025. Cocoa: A minimum bayes risk framework bridging confidence and consistency for uncertainty quantification in llms. In *Advances in Neural Information Processing Systems*.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. 2005. *Algorithmic Learning in a Random World*. Springer.

Huajun Xi, Jianguo Huang, Kangdao Liu, Lei Feng, and Hongxin Wei. 2024. Does confidence calibration improve conformal prediction? *arXiv preprint arXiv:2402.04344*.

Bianca Zadrozny and Charles Elkan. 2002. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699. ACM.

Viktor Örnbratt and Johan Hallberg Szabadváry. 2025. Conformal llm multi-label text classification with binary relevance approach. In *Proceedings of the Fourteenth Symposium on Conformal and Probabilistic Prediction with Applications*, volume 266 of *Proceedings of Machine Learning Research*, pages 214–229. PMLR.

A Appendix

A.1 Dataset Analysis

A.1.1 lazyreview

The LazyReview dataset contains 500 review cases in total. Table 4 reports the number of positive instances for each label. The labels *None* and *Not Enough Info* indicate the absence of issues. Excluding these two labels, only eight labels contain more than 10 positive samples. Therefore, our experiments mainly focus on these eight labels, while the remaining labels are excluded from evaluation due to the extremely small number of positive instances. It is also worth noting that the dataset exhibits a severe class imbalance, with substantially more negative samples than positive ones, which makes the classification task particularly challenging.

Label	Count
None	129
Not Enough Info	94
The authors could also do [extra experiment X]	83
The authors should have done [X] instead	31
The paper has language errors / Writing Style	30
The results are not novel	29
The results are not surprising	24
The paper is missing the comparison to the [latest X]	22
The results contradict what I would expect	18
The topic is too niche / Narrow topics	16
The approach is tested only on [not English], so unclear if it will generalize to other languages	7
This method is too simple	6
The results are negative	4
This has no precedent in existing literature	3
The results do not surpass the latest SOTA	3
The paper doesn't use [my preferred methodology], e.g., deep learning	1

Table 4: Label counting of LazyReview dataset.

A.1.2 D3code

We use the **moral** subset of the D3code dataset, as it provides annotator rating information that is well suited for studying subjective label prediction under annotator disagreement. This subset contains 489 cases in total. Table 5 reports the number of samples and positive instances for each label. To define positive instances, we adopt a strict criterion based on majority voting: a case is considered positive if the majority of annotators assign a rating greater than zero. Under this definition, the dataset contains only 89 positive samples. Consequently, the subset is highly imbalanced, with substantially more negative than positive instances, making the prediction task particularly challenging.

A.1.3 MHS

The MHS dataset is derived from a large corpus where each comment is annotated by multiple raters. While the large number of annotations enables the study of subjective judgments and annotator disagreement, it also introduces substantial variability in annotation quality. To improve label reliability, we first aggregate annotations at the comment level and use the Rasch standard error provided in the dataset as a measure of annotation uncertainty. We restrict our evaluation to the 20% most reliable comments (i.e., those with the lowest standard error), resulting in a subset of 9,003 comments.

Label	#Samples	#Positive
authority	100	14
care	99	23
fairness	98	23
purity	98	14
loyalty	94	15
Total	**489**	**89**

Table 5: Label distribution of the *D3code-moral* subset.

For each label, we compute the mean score across annotators and convert it into a binary label. Since most labels follow a five-point rating scale, we treat a comment as positive if the average score is at least 2.0. For the *hatespeech* label, which follows a three-point scale, we use a threshold of 1.0. This procedure produces binary labels while preserving the consensus information captured by the averaged ratings.

To construct a benchmark that allows meaningful evaluation across labels, we further control the number of positive instances per label. Some labels in MHS contain substantially more positive examples than others, which could otherwise dominate the training and evaluation process. Therefore, for labels with more than 100 positive instances, we randomly sample 100 positives; for labels with fewer positives, we retain all available samples. This strategy ensures that each label has a comparable number of positive instances while still preserving the diversity of the original corpus.

A.2 Baseline Process

Data splits. For each dataset, we use an 80/20 train/test split stratified by label (random seed 42). The 20% test fold is further split 50/50 into a calibration set (n_{cal}) and an evaluation test set (n_{test}), yielding: LazyReview n_{cal}=54, n_{test}=54; D3Code n_{cal}=129, n_{test}=131; MHS n_{cal}=87, n_{test}=87 (per label). For MHS (multi-label), we evaluate each binary label independently and report macro-averages across the nine labels. For zero-shot baselines (APRICOT, CoCoA, ConfTuner), we use the

Label	#Positive
sentiment	95
respect	90
insult	82
humiliate	76
status	96
dehumanize	46
violence	6
genocide	3
attack_defend	88
hatespeech	39

Table 6: Label distribution of the MHS benchmark used in our experiments.

authors' recommended prompts and configurations on the same LLM backbones (Llama-3.1-8B, Phi-4-14B) for fair comparison; their LLM calls follow the original prompting scheme and do not use sub-question features.

ConfTuner trains LLMs to explicitly express their prediction confidence in natural language. During inference, the model outputs both an answer and a confidence value (e.g., "Confidence: 85%"). We use the publicly available trained *ConfTuner-LLaMA* model for inference and extract this confidence value and normalize it to the range $[0, 1]$.

Since ConfTuner directly produces a confidence estimate rather than a calibrated probability, we treat the predicted confidence as a proxy probability for CRC. Specifically, the nonconformity score is defined as

$$s(x) = 1 - \text{confidence}(x).$$

The threshold is then estimated on the calibration set using these scores, and prediction sets are constructed accordingly.

APRICOT calibrates LLM predictions using only the model's generated outputs without requiring access to internal logits. Following the original method, we first obtain answers from *Llama-3.1-8B* together with token-level probabilities for candidate responses. From these probabilities we derive a base confidence score corresponding to the probability of the predicted answer.

Given the question, the predicted answer, and its base confidence, APRICOT applies a lightweight calibration model trained on generation features to produce a calibrated probability estimate. Specifically, we use the publicly released APRICOT classifier `apricot_clustering_trivia_qa_deberta-v3-base_for_vicuna-7b-v1.5` to map the triplet $(\text{question}, \text{answer}, \text{confidence})$ to a calibrated probability. These calibrated probabilities are then used to compute probability-based evaluation metrics.

For CRC, we treat the APRICOT-calibrated probability as the model confidence and apply standard split conformal calibration on the calibration set to construct prediction sets.

CoCoA estimates prediction reliability by combining model confidence and response consistency across multiple stochastic generations. Following the original method, we perform 10 stochastic samples using *Llama-3.1-8B*. For each generated response, we extract confidence based on the normalized token log-probabilities.

Let c_i denote the confidence of the i-th generation and y_i its predicted label. The consistency score is defined as the proportion of the most frequent answer among the sampled responses. The final CoCoA score is computed as

$$\text{score}(x) = \text{confidence}(x) \times \text{consistency}(x),$$

Since CoCoA is originally designed for uncertainty quantification rather than probability calibration, it produces a score instead of a calibrated probability $p(y \mid x)$, which makes it unsuitable for direct use in standard calibration metrics. To enable comparison with other methods, we convert its outputs into probabilities by computing the confidence-weighted proportion of *Yes* responses. This derived probability is then used to evaluate probability-based metrics.

For CRC, we directly use the CoCoA score as the conformity measure. The corresponding nonconformity score is defined as

$$s'(x) = 1 - \text{score}(x),$$

which follows the standard formulation where larger scores indicate less conformity.

A.3 Main Performance Results on D3Code

A.4 Main Performance Results on MHS

A.5 Effect of the Feature-Generation Model

The choice of sub-question generator measurably affects downstream reliability. On MHS, Gemini-generated sub-questions outperform GPT-5–generated ones on 9 of the 12 (backbone, metric) cells. The advantage is largest on ranking and likelihood — AUROC improves by $+5.4$ pp on Llama and $+5.0$ pp on Phi, NLL drops by 20–25% on

Table 7: Multi-seed results on D3Code. Values are mean±std over five seeds, with standard deviations shown in a smaller font. Bold indicates the best value within each block.

Method	AUROC↑	AUPRC↑	Brier↓	NLL↓	ECE↓	|Cov−0.9| ↓	|SS−1| ↓
APRICOT	**0.622±0.000**	**0.106±0.000**	0.170±0.000	0.530±0.000	0.367±0.000	**0.022±0.000**	**0.092±0.000**
CoCoA	0.462±0.000	0.081±0.000	0.176±0.000	0.592±0.000	0.310±0.000	0.167±0.000	0.100±0.000
ConfTuner	0.484±0.000	0.039±0.000	0.469±0.000	1.256±0.000	0.625±0.000	0.094±0.000	0.900±0.000
Llama-3.1-8B							
Vanilla	**0.723±0.043**	0.140±0.019	0.058±0.007	0.217±0.020	0.105±0.018	0.026±0.009	0.034±0.022
Beta	0.673±0.051	**0.141±0.049**	0.035±0.000	0.164±0.011	0.021±0.003	0.031±0.021	0.084±0.036
Isotonic	0.676±0.040	0.090±0.015	**0.035±0.001**	0.189±0.026	**0.016±0.004**	0.176±0.074	0.254±0.080
Platt	**0.723±0.043**	0.140±0.019	0.036±0.000	**0.153±0.008**	0.029±0.006	0.033±0.019	0.069±0.044
TS	**0.723±0.043**	0.140±0.019	0.051±0.007	0.194±0.029	0.074±0.022	0.036±0.017	0.044±0.042
Spline	0.604±0.088	0.126±0.040	0.040±0.002	0.291±0.028	0.028±0.011	0.022±0.008	0.059±0.027
CCal	0.568±0.040	0.121±0.054	0.041±0.002	0.445±0.030	0.048±0.004	0.039±0.010	**0.018±0.009**
MCal	0.708±0.026	0.138±0.025	0.065±0.022	0.341±0.216	0.096±0.033	**0.015±0.006**	0.059±0.053
KCal	0.608±0.060	0.131±0.018	0.036±0.000	0.183±0.037	0.016±0.003	0.019±0.010	0.064±0.025
APS	**0.723±0.043**	0.140±0.019	0.058±0.007	0.217±0.020	0.105±0.018	0.055±0.007	0.041±0.023
RAPS	**0.723±0.043**	0.140±0.019	0.058±0.007	0.217±0.020	0.105±0.018	0.054±0.006	0.031±0.011
Phi-4-14B							
Vanilla	**0.719±0.032**	**0.180±0.023**	0.090±0.005	0.307±0.011	0.179±0.006	0.029±0.006	0.044±0.062
Beta	0.530±0.063	0.096±0.036	0.036±0.000	0.226±0.014	0.030±0.002	0.054±0.045	0.105±0.064
Isotonic	0.652±0.044	0.079±0.011	**0.035±0.001**	0.210±0.036	**0.014±0.002**	0.131±0.138	0.147±0.192
Platt	0.688±0.072	0.172±0.036	0.036±0.001	**0.159±0.004**	0.033±0.004	0.050±0.051	0.094±0.075
TS	**0.719±0.032**	**0.180±0.023**	0.077±0.008	0.271±0.021	0.137±0.021	0.037±0.013	0.062±0.057
Spline	0.634±0.058	0.159±0.042	0.042±0.006	0.334±0.090	0.036±0.013	0.037±0.035	0.088±0.053
CCal	0.689±0.022	0.160±0.024	0.099±0.029	1.002±0.397	0.112±0.030	0.047±0.009	0.334±0.112
MCal	0.686±0.032	0.143±0.028	0.079±0.012	0.334±0.048	0.133±0.034	**0.018±0.009**	0.029±0.038
KCal	0.698±0.074	0.149±0.033	0.036±0.000	0.167±0.027	0.017±0.002	0.029±0.024	0.085±0.040
APS	**0.719±0.032**	**0.180±0.023**	0.090±0.005	0.307±0.011	0.179±0.006	0.036±0.046	0.016±0.015
RAPS	**0.719±0.032**	**0.180±0.023**	0.090±0.005	0.307±0.011	0.179±0.006	0.034±0.047	**0.016±0.014**

both backbones — and Gemini also yields smaller prediction sets at the same coverage target (mean $|\mathrm{SS}-1|$ 0.298 vs. 0.508). GPT-5 retains an edge on ECE in both backbones, indicating that its features produce probabilities that are slightly better aligned in magnitude even though the underlying ranking is weaker. The split is consistent with the framework's overall premise: the LLM-elicited features carry most of the discriminative signal, so the generator that produces sharper Yes/No probes drives most of the downstream reliability. We treat Gemini as our default generator and report a single-dataset comparison here for compute reasons; extending the comparison to LazyReview and D3Code is a natural next step.

A.6 Effect of Sampling Temperature

We conduct this ablation on MHS only, since the multi-temperature feature pools were generated for MHS as a representative high-label-density benchmark; extending to LazyReview and D3Code would require regenerating LLM sampling runs at additional cost. We expect the qualitative finding (moderate temperatures preferred) to generalize.

Moderate temperatures work best, and the optimum is backbone-specific. Phi prefers T=0.5, which takes 5 of the 6 metrics including AUROC (0.687 vs. 0.650 at T=0.1) and ECE (0.083 vs. 0.110). Llama spreads its wins between T=0.3 (AUROC, AUPRC, $|\mathrm{SS}-1|$) and T=0.5 (Brier, NLL), with T=0.1 taking ECE only. The two extremes underperform on both backbones: T=0.1 saturates the sub-question answers and erases the cross-sample variance that the framework relies on (§4.5), while T=1.0 adds noise the classifier cannot fully discount. Our default T=0.7 sits in the moderate regime and is competitive across metrics on both backbones — on Phi it matches T=0.5 on Brier and trails the best T=0.5 result by at most 2.0 pp AUROC and 2.6 pp ECE. The ablation suggests T=0.5 may yield small additional gains, but the differences are within the range that does not alter the calibrator ranking that underlies our default configuration. We therefore treat T as a tunable hy-

Table 8: Multi-seed results on MHS. Values are mean±std over five seeds, with standard deviations shown in a smaller font. Bold indicates the best value within each block.

Method	AUROC↑	AUPRC↑	Brier↓	NLL↓	ECE↓	$\lvert\mathrm{Cov}-0.9\rvert$ ↓	$\lvert\mathrm{SS}-1\rvert$ ↓
APRICOT	0.656±0.000	0.692±0.000	0.295±0.000	0.786±0.000	0.390±0.000	**0.002±0.000**	0.663±0.000
CoCoA	**0.692±0.000**	**0.829±0.000**	0.204±0.000	0.669±0.000	0.290±0.000	0.059±0.000	**0.087±0.000**
ConfTuner	0.476±0.000	0.641±0.000	0.259±0.000	1.154±0.000	0.300±0.000	0.086±0.000	0.762±0.000
Llama-3.1-8B							
Vanilla	**0.660±0.019**	0.794±0.031	0.141±0.008	0.454±0.020	0.198±0.012	0.039±0.024	0.474±0.054
Beta	0.470±0.022	0.703±0.020	0.312±0.083	3.883±1.049	0.311±0.084	0.089±0.012	0.915±0.050
Isotonic	0.593±0.057	0.744±0.021	0.130±0.004	0.956±0.238	0.132±0.006	**0.024±0.022**	0.425±0.082
Platt	0.630±0.037	0.769±0.024	**0.125±0.013**	**0.436±0.044**	0.139±0.020	0.024±0.022	0.421±0.102
TS	**0.660±0.019**	0.794±0.031	0.138±0.004	0.458±0.011	0.183±0.008	0.039±0.024	0.474±0.054
Spline	0.617±0.078	**0.800±0.058**	0.145±0.010	0.784±0.068	0.165±0.008	0.037±0.018	0.347±0.141
CCal	0.637±0.032	0.776±0.019	0.158±0.024	1.652±0.281	0.181±0.029	0.070±0.009	0.737±0.066
MCal	**0.660±0.019**	0.794±0.031	0.141±0.008	0.454±0.020	0.198±0.012	0.039±0.024	0.474±0.054
KCal	0.602±0.061	0.767±0.027	0.126±0.008	0.520±0.079	**0.114±0.014**	0.025±0.024	0.275±0.130
APS	**0.660±0.019**	0.794±0.031	0.141±0.008	0.454±0.020	0.198±0.012	0.116±0.025	0.015±0.017
RAPS	**0.660±0.019**	0.794±0.031	0.141±0.008	0.454±0.020	0.198±0.012	0.120±0.030	**0.014±0.017**
Phi-4-14B							
Vanilla	0.624±0.021	**0.750±0.011**	0.159±0.009	0.495±0.021	0.226±0.019	0.027±0.021	0.533±0.082
Beta	0.481±0.028	0.697±0.012	0.293±0.063	3.520±0.932	0.291±0.064	0.088±0.005	0.880±0.045
Isotonic	0.588±0.039	0.721±0.007	0.134±0.006	0.808±0.159	**0.106±0.011**	0.019±0.020	0.510±0.095
Platt	0.620±0.060	0.744±0.012	**0.130±0.005**	**0.436±0.051**	0.108±0.006	0.024±0.023	0.491±0.118
TS	0.624±0.021	**0.750±0.011**	0.148±0.010	0.474±0.040	0.173±0.018	0.027±0.021	0.533±0.082
Spline	0.545±0.022	0.716±0.011	0.167±0.023	1.127±0.291	0.183±0.023	**0.017±0.015**	0.443±0.126
CCal	0.621±0.031	0.732±0.018	0.231±0.033	2.951±0.397	0.241±0.036	0.067±0.014	0.771±0.083
MCal	0.624±0.021	**0.750±0.011**	0.159±0.009	0.495±0.021	0.226±0.019	0.027±0.021	0.533±0.082
KCal	**0.641±0.050**	0.748±0.021	0.135±0.005	0.679±0.164	0.109±0.008	0.020±0.016	0.385±0.082
APS	0.624±0.021	**0.750±0.011**	0.159±0.009	0.495±0.021	0.226±0.019	0.134±0.037	**0.028±0.024**
RAPS	0.624±0.021	**0.750±0.011**	0.159±0.009	0.495±0.021	0.226±0.019	0.141±0.046	**0.028±0.024**

Generator	Backbone	AUROC↑	AUPRC↑	Brier↓	NLL↓	ECE↓	$\lvert\mathrm{SS}-1\rvert$↓
Gemini	Llama	**0.619**	**0.753**	**0.122**	**0.423**	0.121	**0.308**
	Phi	**0.717**	0.764	**0.117**	**0.372**	0.110	**0.287**
GPT-5	Llama	0.566	0.738	0.123	0.528	**0.092**	0.490
	Phi	0.667	**0.766**	0.129	0.493	**0.109**	0.526

Table 9: Effect of the LLM used to generate label-specific sub-questions, evaluated on MHS. Within each backbone, the better value per column is in **bold**; $\lvert\mathrm{SS}-1\rvert$ is bolded as smallest (closest to the ideal singleton).

Backbone	Temp.	AUROC↑	AUPRC↑	Brier↓	NLL↓	ECE↓	$\lvert\mathrm{SS}-1\rvert$↓
Llama	0.1	0.563	0.737	0.123	0.528	**0.087**	0.480
	0.3	**0.656**	**0.750**	0.137	0.505	0.128	**0.471**
	0.5	0.586	0.744	**0.121**	**0.389**	0.101	0.516
	0.7	0.566	0.738	0.123	0.528	0.091	0.490
	1.0	0.543	0.731	0.136	0.552	0.102	0.508
Phi	0.1	0.650	0.754	0.137	0.512	0.110	0.528
	0.3	0.643	0.757	0.138	0.605	0.118	0.525
	0.5	**0.687**	**0.796**	**0.129**	**0.474**	**0.083**	0.505
	0.7	0.667	0.766	**0.129**	0.493	0.109	0.525
	1.0	0.668	0.756	0.132	0.482	0.117	**0.501**

Table 10: Effect of LLM sampling temperature on framework reliability, evaluated on MHS with the default RandomForest + KCal + CRC configuration. Within each backbone, the best value per column is in **bold**; $\lvert\mathrm{SS}-1\rvert$ is bolded as smallest (closest to the ideal singleton). Tied displayed values are bolded together.

perparameter of the elicitation step and recommend the moderate range $T \in [0.3, 0.7]$ in deployment.

A.7 Qualitative Case Studies

[Singleton-correct, id=4723] The text is a benign, context-poor question about regional origin. Probing returns 'No' to 8 of 10 hate-speech sub-questions and 'Yes' only to k=2 ('target people based on group identity') and k=3 ('demean a category of people') – consistently across all 5 runs ($\sigma_k = 0$ for every k, so $S(x) = 1$). KCal places the example deep in the negative neighborhood ($\hat{p} \approx 10^{-6}$) and CRC returns the singleton 0, matching the gold label.

[Ambiguous, id=18051] A religious commentary that explicitly distances itself from 'extreme Christianity' while quoting condemnations. Probing strongly fires on most sub-questions, but k=4 ('portray a group as inferior') returns Yes in 4 of 5 runs and k=10 ('attack people for who they are') returns No in 4 of 5 runs – yielding $\mu_{10} = -0.60$ and the only large sigma values ($\sigma_4 = \sigma_{10} = 0.80$)

Table 11: Three qualitative cases on MHS for label `hatespeech` under the default configuration (Phi-4-14B backbone, RandomForest + KCal + CRC, α=0.1, K=10 sub-questions, R=5 samples). $S(x) = \frac{1}{K}\sum_k \max(p_k, 1-p_k)$ (Eq. 4), with $p_k = (\mu_k + 1)/2$ the across-run Yes-frequency of sub-question k. $S(x) \in [0.5, 1]$; $S(x)$=1 when every sub-question gives the same answer in all R runs.

Category	Text (truncated)	$S(x)$	$\hat{p}$	$C(x)$	y^*
Singleton-correct	`[4723]` They're inner Mongolian, perhaps?	1.00	0.000	$\{0\}$	0
Ambiguous	`[18051]` An extreme Christian would not be using the word to take authority he does not have. W...	0.96	0.503	$\{0, 1\}$	0
Incorrect	`[28446]` U bitches be annoying and it ain't me I'm not bitches	0.94	0.379	$\{0\}$	1

Table 12: Probing evidence for the singleton-correct case (`id=4723`). Y_k = count of Yes answers over R=5 runs; σ_k = across-run std of the $\{+1, -1\}$ answer.

k	Sub-question (abridged)	$Y_k/5$	σ_k
1	hostility	0	0.00
2	group identity	5	0.00
3	demean group	5	0.00
4	group inferior	0	0.00
5	contempt	0	0.00
6	neg. generalization	0	0.00
7	prejudice	0	0.00
8	animosity	0	0.00
9	undesirable	0	0.00
10	attack identity	0	0.00

Table 13: Probing evidence for the ambiguous case (`id=18051`). Y_k = count of Yes answers over R=5 runs; σ_k = across-run std of the $\{+1, -1\}$ answer.

k	Sub-question (abridged)	$Y_k/5$	σ_k
1	hostility	5	0.00
2	group identity	5	0.00
3	demean group	5	0.00
4	group inferior	4	0.80
5	contempt	5	0.00
6	neg. generalization	5	0.00
7	prejudice	0	0.00
8	animosity	5	0.00
9	undesirable	5	0.00
10	attack identity	1	0.80

in the vector. The classifier lands at p_hat = 0.503 (right at the boundary), and CRC therefore retains both 0 and 1. Coverage is preserved without committing to a label, which is exactly the desired behavior on borderline cases.

[Incorrect, id=28446] A short utterance containing the slur "bitches" but framed as in-group complaining — the speaker explicitly excludes herself ("I'm not bitches"). Probing strongly fires on sub-questions about hostility and contempt ($k = 1..9$ mostly Yes), but k=10 ("attack people for who they are") correctly registers that this is not a group identity attack: 3 of 5 runs return No. KCal reads the resulting phi as borderline negative ($\hat{p} = 0.379$), and CRC's threshold places this just inside $C = 0$. The annotator-assigned label $y* = 1$ reflects a lexical-level judgment on the slur itself;

Table 14: Probing evidence for the incorrect case (`id=28446`). Y_k = count of Yes answers over R=5 runs; σ_k = across-run std of the $\{+1, -1\}$ answer.

k	Sub-question (abridged)	$Y_k/5$	σ_k
1	hostility	5	0.00
2	group identity	5	0.00
3	demean group	5	0.00
4	group inferior	5	0.00
5	contempt	5	0.00
6	neg. generalization	5	0.00
7	prejudice	4	0.80
8	animosity	5	0.00
9	undesirable	5	0.00
10	attack identity	2	0.98

the framework, reading semantic structure rather than lexicon, disagrees. This is a useful diagnostic of where probing-based pipelines fail: not when sub-question answers are chaotic, but when LLM-level semantic reasoning genuinely diverges from annotator-level lexical conventions.

A.8 Feature Question Prompt Template

You are given:

- A text: text

- An issue label: issue

- Desired number of questions: 10

Your goal: Extract abstract, high level features that determine whether each text has the specified issue. Express each feature as an objective Yes/No question.

Rules you MUST follow:

1) Use ONLY the provided text. Do NOT rely on external knowledge or assumptions.

2) Work at the ABSTRACT level (features that could generalize across wording), not surface phrasing.

3) Every question must be answerable by inspecting a single text in isolation.

4) Questions must be neutral (no leading language), atomic (one idea per question), and non-redundant.

5) Avoid double negatives, multi-part questions, subjective terms ("clearly", "obviously"), or jargon unless it appears in the text.

6) Prefer beginnings like "Does…", "Is…", "Are…", "Has…", "Do…", "Can…".

7) Keep each question concise (ideally 18 words).

8) Produce EXACTLY N questions. If N is not given, produce 10.

9) OUTPUT FORMAT: Return ONLY a Python list of strings, with double quotes, no code fences, no comments, no trailing commas.

10) Do NOT include any analysis, notes, or explanations in the output — only the list.

Step-by-step procedure:

A) Parse inventory:

- Scan the text to note common patterns about claims, evidence, specificity, comparisons, citations, quantification, assumptions, scope/coverage, and consistency.

- Identify cues that would indicate presence/absence of the issue issue.

B) Abstract feature mining:

- Convert recurring patterns into abstract properties that can reflect the similarity between the text you are given

- Ensure each property directly supports diagnosing issue.

C) Draft discriminative Yes/No tests:

- For each property, write a Yes/No question that a reviewer could answer from a single text.

- Ensure questions collectively cover: evidence/citations, specificity/locating, correctness/faithfulness, scope, quantification, reproducibility/actionability, internal consistency, and fairness/balance (as applicable).

D) Prune & refine:

- Remove overlaps; keep the most general, text-checkable forms.

- Rewrite to be atomic, neutral, and concise.

- Ensure each question distinguishes "issue present" vs "issue absent".

E) Final checks (must pass all):

- [ ] Exactly N questions.

- [ ] All are Yes/No answerable from a single text.

- [ ] No duplicates or near-duplicates.

- [ ] No restating issue verbatim; focus on testable properties.

- [ ] Python list of strings ONLY, no extra text.

Now produce the final output.