ASSAY: Reconciling Stability and Validity in Reference-Free Scientific LLM Judges

Anonymous ACL submission

Abstract

LLM judges have emerged as a key approach to evaluating open-ended scientific outputs that resist automated verification, yet their faithfulness in reference-free settings remains underexplored, particularly for domains such as novel materials synthesis where no ground-truth reference exists. Such judges produce false positives that accept scientifically invalid recipes and false negatives that penalize semantically equivalent ones in unfamiliar notation. We introduce **ASSAY**, a probe suite built on domain-grounded augmentations of synthesis recipes, and reveal that these two failure modes form a systematic Stability–Validity trade-off across thirteen judge models. To resolve this trade-off, we train **SynthJudge-8B** via DPO on contrastive pairs derived from the same augmentations, improving judgment accuracy from 51.4% to 85.1% while maintaining agreement with domain experts. We release our probe suite and trained model to support faithful automated evaluation in scientific domains beyond materials synthesis.

1 Introduction

Large language models (LLMs) are increasingly deployed as research collaborators in the physical sciences–proposing synthesis recipes, planning autonomous experiments, and generating candidate hypotheses across materials science, chemistry, and biology (Boiko et al., 2023; M. Bran et al., 2024; Noh et al., 2026)–yet deploying them as trustworthy research tools requires evaluation protocols that reliably distinguish scientifically valid outputs from superficially plausible but invalid ones (Mirza et al., 2025; Castro Nascimento and Pimentel, 2023; Alkaissi and McFarlane, 2023; Huang et al., 2025). Because such outputs are open-ended text that lacks the rigid structure required for automated verification, surface-level metrics such as BLEU and ROUGE (Papineni et al., 2002; Lin, 2004) cannot assess them, and expert human review does not scale.

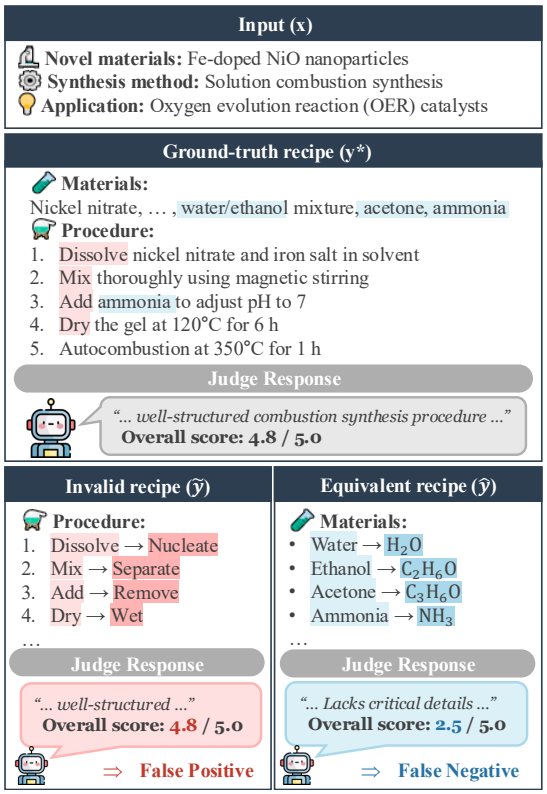

Figure 1: **Two failure modes of LLM judges in reference-free scientific evaluation.** An invalid recipe ($\tilde{y}$) with reversed procedural verbs receives the same high score as the ground-truth (**false positive**), while an equivalent recipe ($\hat{y}$) with chemical formulas substituted for names receives a much lower score (**false negative**).

LLM-as-a-Judge (Laskar et al., 2025; D'Souza et al., 2025; Croxford et al., 2025) has emerged as a scalable alternative that addresses this gap in scientific domain, leveraging a language model to assign rubric-based scores or pairwise preferences to candidate outputs. However, existing systems derive scores from comparison against a ground-truth reference (Laskar et al., 2025; Kim et al., 2025) to compensate for the judge model's limited domain knowledge. This requirement does not extend

to settings where a ground-truth output is unavailable, such as novel materials synthesis (Szymanski et al., 2023), for which reference-free evaluation is the only available alternative. Moreover, whether LLM judges yield faithful evaluations in the absence of a reference has not been systematically examined. This motivates the central question of our work: *Are LLM-as-a-Judge systems capable of faithful reference-free evaluation in scientific domains?* We address this question in the context of materials synthesis recipe evaluation, a recently established setting in which LLM-generated synthesis procedures are assessed prior to physical laboratory trials (Kim et al., 2025).

An ideal judge would ground its assessment in scientific principles such as thermodynamic feasibility, reagent compatibility, and procedure coherence, yet current LLMs do not reliably possess such domain expertise (Zaki et al., 2024; Mirza et al., 2024; Guo et al., 2023). We argue that judges instead approximate scientific quality through surface cues such as structural plausibility and notational convention. As illustrated in Figure 1, this reliance produces two failure modes. First, judges default to surface plausibility as a proxy for scientific validity, yielding **false positives** that reward structurally coherent but scientifically invalid recipes (Wu and Aji, 2025; Zeng et al., 2023; Zheng et al., 2023). Second, judges internalize biases toward canonical notation, yielding **false negatives** that penalize semantically equivalent recipes on the basis of surface form alone (Ganeeva et al., 2025; Wu and Aji, 2025).

To diagnose these failures, we introduce **ASSAY** (Automated Scientific Synthesis Assessment of Your Judges), a probe suite that quantifies both failure modes through domain-grounded augmentations of ground-truth recipes along two complementary modes. *Error Injection* introduces scientifically invalid modifications to probe *Validity*, the degree to which judges penalize invalid recipes. *Equivalence Rewriting* substitutes semantically equivalent representations to probe *Stability*, the degree to which judges preserve scores under such rewriting.

Experiments across thirteen judge models reveal a consistent Stability–Validity trade-off in which judges that correctly penalize invalid recipes also penalize semantically equivalent ones, and vice versa. To mitigate this trade-off, we exploit the fact that augmented recipes carry intended labels that naturally define contrastive pairs for preference learning (Ye et al., 2024b; Trivedi et al., 2024). When trained on these pairs via Direct Preference Optimization (Rafailov et al., 2023), **SynthJudge-8B**, a reference-free judge initialized from Qwen3-8B, attains 85.1% judgment accuracy compared to 51.4% for the base model, surpassing all thirteen baselines.

Our contributions are as follows:

- **Diagnostic Approach.** We introduce **ASSAY**, which exposes false positives from insufficient domain knowledge and false negatives from surface-form bias through two complementary augmentation modes applied to ground-truth synthesis recipes.

- **Empirical Findings.** Evaluation of thirteen judge models reveals a consistent Stability–Validity trade-off, which we further characterize across swap types, training stages, and synthesis domains.

- **Trade-off Mitigation.** **SynthJudge-8B**, trained via DPO on augmented contrastive pairs, improves judgment accuracy from 51.4% to 85.1%, surpassing all baselines while maintaining expert agreement.

2 Related Work

2.1 Diagnosing Failures in LLM-as-a-Judge

Failures of LLM-as-a-Judge have been studied along several distinct dimensions, ranging from surface-feature biases to hallucinated rationales and knowledge limitations. Most prior work has focused on surface-feature biases, including position, verbosity, and self-enhancement through which judges respond to formatting cues rather than content quality (Zheng et al., 2023; Wu and Aji, 2025; Chen et al., 2024b), together forming a broad taxonomy of judgment biases (Ye et al., 2024a). Beyond such biases, judges produce hallucinated rationales decoupled from the scores they assign (Turpin et al., 2023; Stephan et al., 2025; Marioriyad et al., 2026), and lack the domain expertise to apply nuanced criteria in specialized fields (Szymanski et al., 2023; Moon et al., 2025; Belmadani et al., 2026). Yet these analyses diagnose judge failures by comparing judge outputs against an external reference, leaving failures that emerge when no such reference is available largely unexamined.

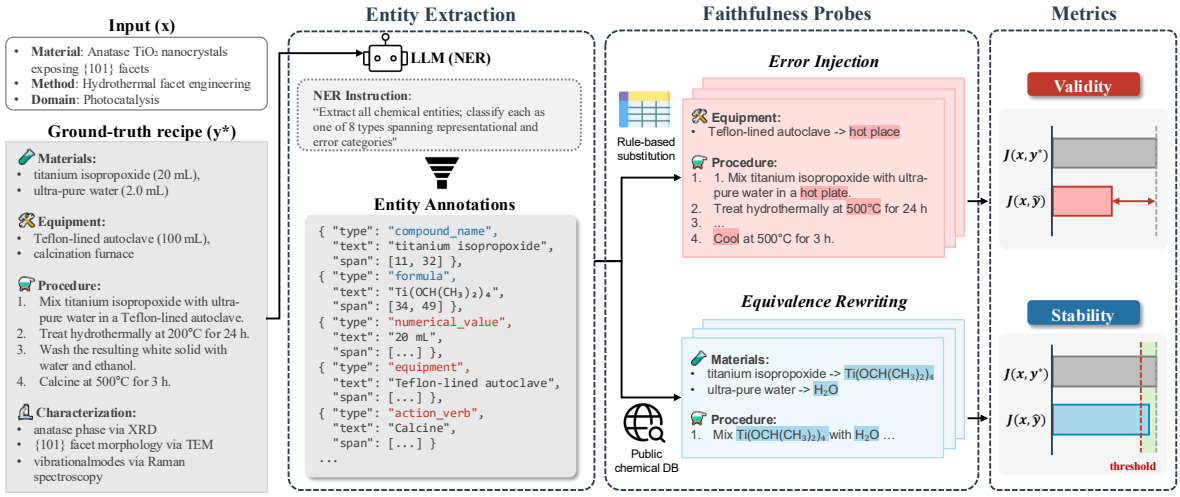

Figure 2: Overview of **ASSAY**. Given a ground-truth recipe y^* and synthesis target x, entity extraction identifies chemical entities, which are then rewritten via one of eight swap types (Table 1). *Error Injection* (pink) produces scientifically invalid recipes $\tilde{y}$ to probe Validity, while *Equivalence Rewriting* (blue) produces semantically equivalent recipes $\hat{y}$ to probe Stability.

2.2 Reference-free Evaluation

Reference-free evaluation has emerged as a distinct paradigm for settings in which canonical answers are unavailable, extended to diverse domains under different calibration regimes. When the task is open-ended, judges are calibrated by aggregating human preferences over intrinsic qualities such as coherence and engagement (Chiang and Lee, 2023; Zheng et al., 2023; Liu et al., 2023), which has also been extended to vision-language tasks (Chen et al., 2024a; Lee et al., 2024). When the task admits objective verification, judges are calibrated against domain-specific signals, including executable test cases in code generation (Tong and Zhang, 2024; Zhuo, 2024), reference translations in machine translation (Rei et al., 2022), and curated answer keys in mathematical reasoning (Xia et al., 2025). Materials synthesis recipe evaluation is amenable to neither path, as a recipe's scientific validity is determined only by physical execution, motivating a diagnostic approach that can establish judge faithfulness without any pre-execution signal.

2.3 Representational Robustness in Scientific Language Models

Scientific language models exhibit sensitivity to meaning-preserving transformations of domain-specific inputs. As predictors, chemical language models such as ChemBERTa-2 (Ahmad et al., 2022) and MolT5 (Edwards et al., 2022) are sensitive to surface-level variations of SMILES strings (Ganeeva et al., 2025; Li et al., 2025), and analogous fragility has been reported for biomedical (Moradi and Samwald, 2022) and protein language models (Carbone et al., 2022). As translators, they exhibit limited consistency across notational conventions and limited accuracy in conversion (Yan et al., 2025; Runcie et al., 2025). Yet these analyses have focused on the predictive and translational roles of scientific language models, leaving their evaluative role largely unexamined.

3 Diagnosing Judge Faithfulness

We present **ASSAY**, a probe suite for auditing the faithfulness of reference-free scientific judges. We define the evaluation task in §3.1, describe the faithfulness probes in §3.2, and formalize the resulting metrics in §3.3.

3.1 Task Setup

To examine whether reference-free judges evaluate scientific content or surface form, we adopt the materials synthesis recipe evaluation task of Kim et al. (2025), in which a judge model J scores a candidate recipe y on a 1–5 rubric given only a synthesis target x, producing a scalar overall score $s = J(x, y)$. The ground-truth recipe y^* is unavailable to the judge, as novel materials synthesis lacks established synthesis routes (Szymanski et al., 2023). We evaluate thirteen judge models on the 169 high-impact test recipes from AlchemyBench (Kim et al., 2025), spanning three cate-

Table 1: The eight swap types for faithfulness probes. *Error Injection* produces scientifically invalid recipes $\tilde{y}$; *Equivalence Rewriting* produces semantically equivalent recipes $\hat{y}$.

Mode	Swap Type	Definition	Example
Error Injection ($\tilde{y}$)	`Element`	Replace element with same-group element	Li → Na
	`Numerical`	Replace quantity with infeasible value	850 °C → 1700 °C
	`Equipment`	Replace equipment with incompatible alternative	furnace → hot plate
	`Action`	Replace verb with its antonym	heat → cool
Equivalence Rewriting ($\hat{y}$)	`Formula`	Convert compound name to molecular formula	ethanol → C_2H_6O
	`Name`	Convert molecular formula to common name	C_2H_6O → ethanol
	`IUPAC`	Convert compound name or formula to IUPAC systematic name	acetone → propan-2-one
	`Cross-lingual`	Translate full recipe text to another language	"Dissolve…" → "dissoudre…"

gories: general-purpose *Instruct* models, *Reasoning*-augmented models, and evaluation-*Specialized* models. The full evaluation rubric, prompt templates, and model details are provided in Appendix A.

3.2 Faithfulness Probes

To systematically diagnose the two failure modes—false positives that accept invalid recipes and false negatives that penalize equivalent ones—we construct augmentation variants of y^* along two complementary modes, as illustrated in Figure 2. Both modes first extract the set of chemical entities from y^* via LLM-based NER (Appendix B.1), then rewrite each according to predefined rules.

Error Injection replaces identified entities with scientifically invalid alternatives—for example, substituting an element with one from the same periodic group or reversing a procedural verb to its antonym. We define the resulting error injected recipe $\tilde{y}$ as

$$\tilde{y} = \mathcal{T}_{\text{err}}(y^*)\,. \tag{1}$$

Equivalence Rewriting converts identified entities into semantically equivalent representations—for example, converting a compound name to its molecular formula or translating the recipe into another language. We define the resulting equivalent rewritten recipe $\hat{y}$ as

$$\hat{y} = \mathcal{T}_{\text{eq}}(y^*)\,. \tag{2}$$

Each augmentation mode operates through four swap types defined in Table 1, and implementation details are provided in Appendix B.2. Three materials science experts independently annotated 595 augmented entities to verify whether each output matched its intended label. Augmentations that a majority of annotators judged as failing were excluded, and all subsequent experiments use this filtered subset (Appendix B.3).

3.3 Metrics

Both transformations are parameterized by a rate $\rho \in (0, 1]$ controlling the fraction of entities rewritten, yielding $\tilde{y}_\rho = \mathcal{T}_{\text{err}}(y^*, \rho)$ and $\hat{y}_\rho = \mathcal{T}_{\text{eq}}(y^*, \rho)$.

Validity measures whether judges detect scientifically invalid recipes. At each ρ, we identify an *Error Injection* output $\tilde{y}_\rho$ as detected when the judge assigns it a substantially lower score than y^*:

$$d(\tilde{y}_\rho) = \mathbb{1}\big[J(x, y^*) - J(x, \tilde{y}_\rho) > \tau\big], \quad \tau = 0.5\,. \tag{3}$$

The detection rate at each ρ is defined as $V(\rho) = \frac{1}{N}\sum_{i=1}^{N} d(\tilde{y}_{i,\rho})$. VALIDITY aggregates $V(\rho)$ into a single score via trapezoidal AUC over $\rho \in \{0.01, 0.02, 0.05, 0.10, 0.15\}$, normalized to $[0, 1]$.

Stability measures whether judges preserve scores under semantically equivalent rewriting. At each ρ, we identify an *Equivalence Rewriting* output $\hat{y}_\rho$ as stable when the judge's score remains within τ of the ground-truth score:

$$d(\hat{y}_\rho) = \mathbb{1}\big[|J(x, y^*) - J(x, \hat{y}_\rho)| \le \tau\big], \quad \tau = 0.5\,. \tag{4}$$

STABILITY is defined analogously by aggregating stability rate $S(\rho) = \frac{1}{N}\sum_{i=1}^{N} d(\hat{y}_{i,\rho})$.

Accuracy captures overall faithfulness as the AUC of the combined correctness rate, the fraction of all $\tilde{y}_\rho$ and $\hat{y}_\rho$ recipes that receive a correct binary decision at each ρ.

4 How Faithful Are Reference-Free Scientific Judges?

We address this question through four analyses. We first establish that current judges exhibit a system-

Category	Model	Validity	Stability	Accuracy
Instruct	Llama-3.1-8B	.203	.772	.455
	Llama-3.3-70B	.257	**.870**	.529
	Qwen3-8B	.310	.763	.514
	Qwen3-32B	.364	.780	.552
	GPT-4o	.439	.757	.586
	Claude-Sonnet-4.5	.587	.699	.641
	Average	.360	.773	.546
Reasoning	Qwen3-8B-Think	.554	.501	.536
	Qwen3-32B-Think	.578	.523	.559
	o4-mini	.713	.434	.596
	GPT-5-mini	.603	.712	.658
	Gemini-3.1-Pro	**.714**	.778	**.749**
	Average	.632	.590	.619
Specialized	JudgeLRM-7B	.469	.423	.453
	Prometheus-7B-v2	.362	.329	.347
	Average	.415	.376	.400

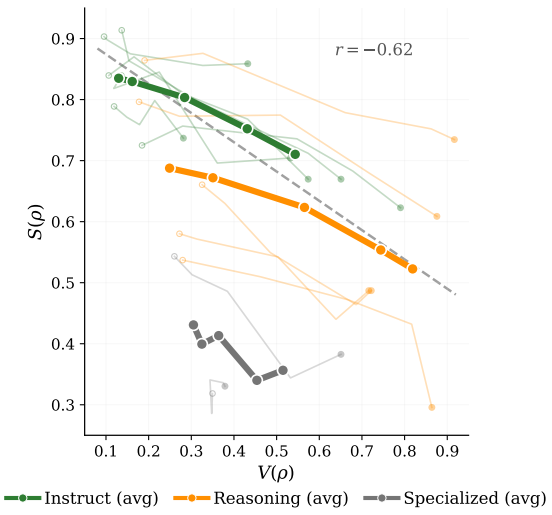

Figure 3: **Left:** AUC summary for thirteen reference-free judges. **Bold** and underline mark the best and second-best per column. **Right:** $S(\rho)$ vs. $V(\rho)$ trajectories as ρ increases from 0.01 to 0.15. Thick lines trace category averages; thin lines show individual models ($\circ = \rho$=0.01, $\bullet = \rho$=0.15). The dashed line shows a linear fit over *Instruct* and *Reasoning* models ($r = -0.62$, $p = 0.043$).

atic Stability–Validity trade-off (§4.1), then identify which swap types are most challenging (§4.2). We next examine how training stages shift the balance between the two metrics (§4.3), and finally assess whether failure patterns vary across synthesis domains (§4.4).

4.1 The Stability–Validity Trade-off (RQ1)

Current reference-free judges exhibit a systematic trade-off in which higher error sensitivity entails lower representational robustness. Figure 3 (left) reports Validity, Stability, and Accuracy for all thirteen judges, yielding a significant negative correlation across the eleven *Instruct* and *Reasoning* models ($r = -0.62$, $p = 0.043$; $r = -0.80$ excluding Gemini-3.1-Pro). Among all thirteen judges, Gemini-3.1-Pro achieves the best balance of the two metrics. At the category level, *Reasoning* models achieve the highest average Validity (0.63) but the lowest Stability (0.59), while *Instruct* models show the opposite (Validity 0.36, Stability 0.77). *Specialized* models trained for general-domain evaluation perform worst on both metrics (Validity 0.42, Stability 0.38), confirming that evaluation ability on general benchmarks does not transfer to scientific faithfulness.

The category-average trajectories in Figure 3 (right) reveal how each category responds to increasing augmentation magnitude. On Validity, *Reasoning* models show the steepest rise ($\Delta = +0.57$), confirming that they are not only more accurate but also more sensitive to injected errors. *Specialized* models, by contrast, show the flattest trajectory ($\Delta = +0.21$), indicating that their low Validity reflects not just weaker performance but a fundamental insensitivity to error signals. On Stability, all three categories decline by a similarly modest margin (*Instruct* $\Delta = -0.12$, *Reasoning* $\Delta = -0.17$, *Specialized* $\Delta = -0.07$), suggesting that false negatives are driven less by the magnitude of rewriting than by whether the judge recognizes the rewritten entities as semantically equivalent.

4.2 Which Swap Types Are Most Challenging? (RQ2)

To identify which swap types drive the failures observed in RQ1, we apply each swap type independently by replacing all identified entities of that type, isolating the effect of each type on judge behavior (Table 2).

On *Error Injection*, `Action` reversal is consistently the least detectable type across all model categories. Reversing procedural verbs preserves the structural form of a synthesis step while rendering it scientifically incoherent, a distinction that current judges rarely penalize. On *Equivalence Rewriting*, `Name` conversion is the most stable type across all categories, while `IUPAC` and `Formula` conversions are the least stable. Notably, IUPAC systematic names are penalized despite following

Table 2: Per-type detection rate and stability rate averaged by category at full augmentation per type. Subscripts denote standard deviation across models. Darker shading indicates higher values within each mode.

	Error Injection Type			
	Elem.	Num.	Equip.	Action
Instruct	$.562_{.05}$	$.574_{.05}$	$.570_{.06}$	$.550_{.03}$
Reas.	$.666_{.06}$	$.697_{.09}$	$.668_{.09}$	$.598_{.06}$
All	$.561_{.13}$	$.586_{.13}$	$.570_{.13}$	$.532_{.11}$

	Equivalence Rewriting Type			
	Formula	Name	IUPAC	Cross-l.
Instruct	$.795_{.05}$	$.809_{.05}$	$.756_{.07}$	$.805_{.08}$
Reas.	$.573_{.10}$	$.671_{.11}$	$.598_{.09}$	$.643_{.12}$
All	$.640_{.18}$	$.693_{.16}$	$.632_{.16}$	$.676_{.18}$

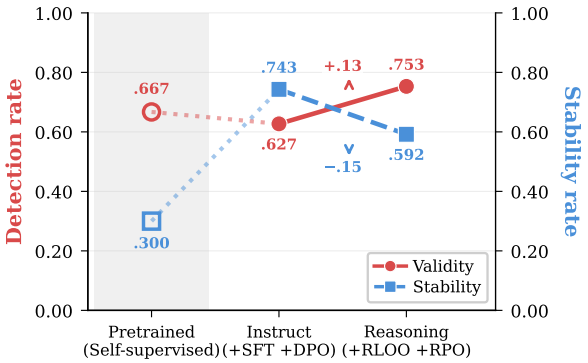

Figure 4: Detection rate and stability rate at $\rho = 0.15$ across training stages of the LLaMA-3.1-8B family. Instruction tuning prioritizes Stability over Validity; reasoning tuning reverses this balance. The Pretrained stage produces outputs conforming to the evaluation format for fewer than 8% of recipes.

a compositional syntax with explicit positional and functional-group encoding (e.g., *propan-2-one* for acetone), while common names such as *water* and *ethanol* are readily accepted. This suggests that structural regularity alone does not ensure representational tolerance, and that judge stability is shaped more by exposure to specific notational conventions during pretraining than by the formal structure of the notation itself.

4.3 How Does Training Shift Faithfulness? (RQ3)

To trace how the Stability–Validity balance evolves during training, we measure both metrics at $\rho = 0.15$ across three stages of the LLaMA-3.1-8B family (Grattafiori et al., 2024), which are Pretrained (self-supervised), Instruct (+SFT and DPO), and Reasoning (+RLOO and RPO via Nemotron-Nano-8B-v1; Bercovich et al. 2025).

As shown in Figure 4, each training stage shifts

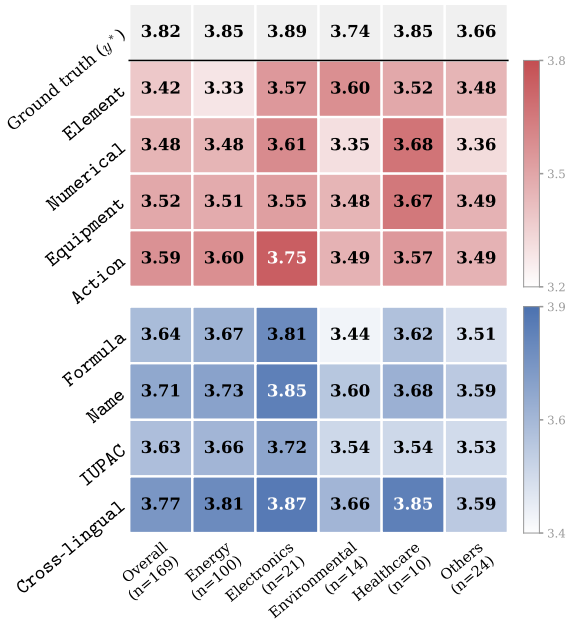

Figure 5: Mean overall score by swap type and synthesis domain, averaged across all judge models. The ground-truth row (grey) provides the baseline score for each domain. For *Error Injection* (red), darker cells indicate higher scores, reflecting undetected errors. For *Equivalence Rewriting* (blue), darker cells indicate higher scores, reflecting greater representational stability.

the balance between the two faithfulness requirements in a way that reflects its learning objective. Instruction tuning yields the highest Stability (74.3%) but moderate Validity (62.7%), suggesting that this stage prioritizes tolerance toward varied input formats over strict content verification. Reasoning tuning reverses this balance. Validity rises to 75.3% while Stability drops to 59.2%, indicating that the model becomes more stringent overall, penalizing not only scientifically invalid content but also notational variants it has not learned to recognize as equivalent.

This pattern mirrors the *Instruct–Reasoning* divergence observed across different model families in RQ1, suggesting that the trade-off is shaped by the training regime rather than model architecture. The Pretrained stage achieves 66.7% Validity but only 30.0% Stability; however, fewer than 8% of its outputs conform to the prescribed evaluation format, limiting the reliability of these estimates.

4.4 Do Failure Patterns Vary Across Synthesis Domains? (RQ4)

To examine whether failure patterns generalize across synthesis domains, we partition the test set into five application domains following the Alche-

myBench (Kim et al., 2025) labels and report mean overall scores per swap type in Figure 5. Energy and Electronics recipes, which constitute the majority of the test set, follow the aggregate pattern from RQ2. `Action` remains the hardest error type to detect, and `IUPAC` the least stable equivalence type. The remaining three domains deviate in ways that reflect their distinct chemical characteristics.

On *Error Injection*, the most challenging swap type varies by domain. In Environmental recipes, it shifts from `Action` to `Element`, where same-group substitutions produce formally charge-compatible formulas (e.g., K $\rightarrow$ Li in a eutectic carbonate) that judges accept as valid despite differences in precursor solubility, phase stability, and thermodynamic feasibility (Szymanski et al., 2023). In Healthcare recipes, both `Numerical` and `Equipment` swaps remain nearest to the ground-truth baseline since biomedical synthesis relies on precise parameter ranges and functionally specialized instruments whose roles are not interchangeable (e.g., replacing an ultrasonic bath with a water bath removes the cavitation essential to nanoparticle dispersion).

On *Equivalence Rewriting*, `Formula` conversion drops sharply in Environmental recipes, partly because polymeric and composite materials prevalent in this domain lose structural information when reduced to monomer-level formulas (Lin et al., 2019) (e.g., HDPE $\rightarrow$ C_2H_4). These variations indicate that the most challenging swap type is domain-dependent, motivating domain-specific calibration as a direction for future work.

5 Mitigating the Stability–Validity Trade-off

The augmentations introduced in §3.2 carry intended labels that naturally define contrastive preference pairs at zero annotation cost. We exploit this signal to train **SynthJudge-8B**, a reference-free judge that resolves the Stability–Validity trade-off.

5.1 Training Setup

Dataset Construction. We construct contrastive pairs by comparing the outputs of a panel of four judge models $\{M_k\}_{k=1}^{K}$ on each augmented recipe. Each model produces a response r_k consisting of a step-by-step rationale followed by an overall score $J_k(\cdot)$. Preference order is determined by the score, while the DPO (Rafailov et al., 2023) objective optimizes over the full response.

To establish a per-sample quality baseline, we define an anchor score as the median over all judges and *Equivalence Rewriting* rates,

$$a(x) = \mathrm{median}\big(\{J_k(x, \hat{y}_\rho)\}_{k,\rho}\big). \tag{5}$$

Because each $\hat{y}_\rho$ is semantically equivalent to y^*, aggregating across four models and five rates yields a debiased estimate of sample quality that is more robust than any single model's score on y^*.

Given two judges M_i and M_j that produce different scores on the same augmented recipe, we select the preferred response based on the augmentation mode. For *Error Injection*, the response assigning the lower score is preferred, as it better detects the injected error:

$$(r_w, r_l) = \begin{cases} (r_i, r_j) & \text{if } J_i(x, \tilde{y}) < J_j(x, \tilde{y}), \\ (r_j, r_i) & \text{otherwise,} \end{cases} \tag{6}$$

For *Equivalence Rewriting*, the direction reverses—the response assigning the higher score is preferred, as it is more robust to surface-form variation. We retain pairs whose score margin exceeds $\delta_{\min} = 0.5$ and exclude responses whose scores deviate substantially from the anchor. Up to five pairs are selected per recipe per augmentation setting, yielding 102K pairs comprising 92K training and 10K validation pairs. Full filtering criteria and augmentation configurations are provided in Appendix D.

Training Objective. Following the contrastive judge training paradigm of Ye et al. (2024b), we initialize **SynthJudge-8B** from Qwen3-8B (Yang et al., 2025) and optimize the Direct Preference Optimization objective (Rafailov et al., 2023):

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}\Big[\log \sigma\big(\beta \log \tfrac{\pi_\theta(r_w|x)}{\pi_{\text{ref}}(r_w|x)} - \beta \log \tfrac{\pi_\theta(r_l|x)}{\pi_{\text{ref}}(r_l|x)}\big)\Big], \tag{7}$$

where π_θ is the policy being trained, π_{ref} is the frozen Qwen3-8B reference, and β controls the deviation from the reference.

5.2 Results

We first examine whether supplying domain knowledge at inference time can resolve the faithfulness failures without additional training. We supplement the Qwen3-8B base judge with a chemistry dictionary that maps entity names to canonical forms (ChemDict), a retrieval module that provides similar training-set recipes as in-context references (RAG), and their combination (CD+RAG). As

Table 3: Mitigation results. Knowledge augmentation (KA) variants are applied to Qwen3-8B at inference time. **SynthJudge-8B** is trained via DPO on 92K contrastive pairs derived from the augmented recipes. Deltas are relative to the *Instruct* model.

Model	Validity	Stability	Accuracy
Qwen3-8B (base)	.310	.763	.514
+ ChemDict	.357 (+.047)	.697 (–.066)	.527 (+.013)
+ RAG	.305 (–.005)	.738 (–.025)	.521 (+.007)
+ CD+RAG	.312 (+.002)	.734 (–.029)	.523 (+.009)
SynthJudge-8B	**.836** (+.526)	**.866** (+.103)	**.851** (+.337)

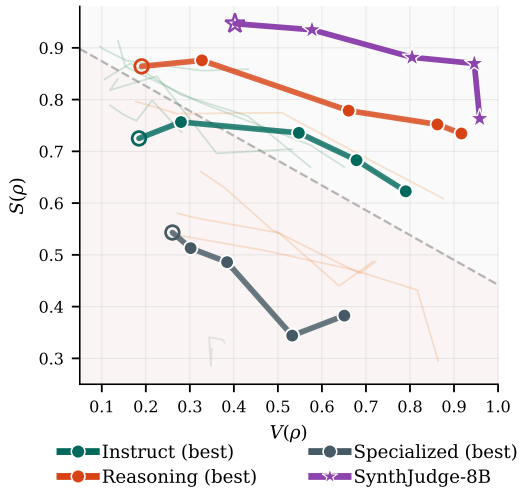

Figure 6: $S(\rho)$ vs. $V(\rho)$ trajectories as ρ increases from 0.01 to 0.15. Thick lines trace the best-performing model per category from Figure 3; thin lines show remaining individual models. The shaded region marks the trade-off observed in §4.1. **SynthJudge-8B** operates above even the strongest baselines across all augmentation rates.

shown in Table 3, all three variants yield marginal Accuracy gains over the base model, and ChemDict raises Validity (0.310 → 0.357) but degrades Stability (0.763 → 0.697). This pattern is consistent across variants. Each improves one metric at the expense of the other, shifting the trade-off rather than resolving it.

SynthJudge-8B, trained on 92K augmented contrastive pairs, advances all three metrics simultaneously, raising Accuracy from 0.514 to 0.851. Figure 6 plots the $S(\rho)$–$V(\rho)$ trajectory of the trained judge alongside the best-performing model per category from Figure 3. While even the strongest baselines remain bound by the trade-off, our model operates above it across all augmentation rates. On Validity, this advantage is most pronounced at low rates, where baseline judges struggle most. At $\rho = 0.01$, where only one or two entities are modified, it already detects 40.2% of injected errors, compared to 18.0% for the best *Instruct* model and 33.0% for the best *Reasoning* model. On Stability, it remains above 0.78 across all rates, declining at a pace comparable to the best *Instruct* model despite substantially higher Validity, confirming that its error sensitivity does not come at the cost of representational tolerance. Appendix D.6 presents representative case studies illustrating how **SynthJudge-8B** resolves both failure modes at the individual recipe level.

Table 4: Pearson correlation with three materials science experts on 20 reference-free evaluations.

Model	Expert r
Qwen3-8B-Think	**.537**
GPT-4o	.416
Qwen3-8B (base)	.405
Llama-3.3-70B	.356
o4-mini	.117
SynthJudge-8B	.412

Human Expert Agreement. A natural concern is whether improving Validity and Stability through contrastive training degrades agreement with human expert judgments. Table 4 reports Pearson correlation with three materials science experts on 20 reference-free evaluations. Agreement in this setting is substantially lower than in reference-based evaluation (Kim et al., 2025), as neither the judge nor the expert has access to a ground-truth recipe. As a result, **SynthJudge-8B** achieves $r = 0.412$, comparable to GPT-4o ($r = 0.416$) and the base Qwen3-8B ($r = 0.405$), confirming that resolving the trade-off does not come at the cost of expert-aligned scoring. Full per-model results are reported in Appendix D.7.

6 Conclusion

We presented **ASSAY**, a probe suite that diagnoses the faithfulness of reference-free LLM judges through two complementary augmentation modes. Evaluation of thirteen judge models revealed a systematic Stability–Validity trade-off, in which higher sensitivity to scientific errors entails lower tolerance for equivalent notational variation. **SynthJudge-8B**, trained via DPO on contrastive pairs derived from the same augmentations, resolves this trade-off by simultaneously improving Validity and Stability while maintaining expert agreement. We release our probe suite and trained model to support faithful automated evaluation in scientific domains beyond materials synthesis.

Limitations

Our work has several limitations that suggest directions for future research. First, **ASSAY** is designed for materials synthesis recipe evaluation, and extending it to other scientific domains would require domain-specific entity types and error taxonomies tailored to each field's representational conventions and failure modes. Second, our evaluation relies on a 1–5 rubric-based scoring paradigm inherited from AlchemyBench (Kim et al., 2025), which may not capture all dimensions of scientific validity. Alternative evaluation formats such as pairwise comparison or verification-based assessment remain unexplored. Third, the `Formula` swap type exhibits the lowest inter-annotator agreement among all types (AC1 = 0.30), reflecting cases in which automated formula assignment produces non-equivalent representations for polymers and large organic molecules. While we mitigate this through majority-vote filtering, the limitation reduces the effective sample size for this type. Finally, our human expert evaluation is conducted on a small set of 20 recipes with three annotators, which limits the statistical power of expert agreement estimates.

References

Walid Ahmad, Elana Simon, Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. 2022. Chemberta-2: Towards chemical foundation models. *arXiv preprint arXiv:2209.01712*.

Hussam Alkaissi and Samy I McFarlane. 2023. Artificial hallucinations in chatgpt: implications in scientific writing. *Cureus*, 15(2).

Anthropic. 2025. System card: Claude sonnet 4.5. https://www.anthropic.com/claude-sonnet-4-5-system-card.

Ikram Belmadani, Oumaima El Khettari, Pacôme Constant dit Beaufils, Richard Dufour, and Benoit Favre. 2026. Who judges the judge? evaluating LLM-as-a-Judge for French medical open-ended QA. In *Proceedings of the First Workshop on Healthcare and AI: Language, Inference, and Generation (HEALING)*. Association for Computational Linguistics.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, and 1 others. 2025. Llama-nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. 2023. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578.

Ginevra Carbone, Francesca Cuturello, Luca Bortolussi, and Alberto Cazzaniga. 2022. Adversarial attacks on protein language models. *bioRxiv*, pages 2022–10.

Cayque Monteiro Castro Nascimento and André Silva Pimentel. 2023. Do large language models understand chemistry? a conversation with chatgpt. *Journal of Chemical Information and Modeling*, 63(6):1649–1655.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yaochen Wang, Yinuo Liu, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. 2024a. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. In *Forty-first International Conference on Machine Learning*.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. 2024b. Humans or llms as the judge? a study on judgement bias. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8301–8327.

Nuo Chen, Zhiyuan Hu, Qingyun Zou, Jiaying Wu, Qian Wang, Bryan Hooi, and Bingsheng He. 2025. Judgelrm: Large reasoning models as a judge. *arXiv preprint arXiv:2504.00050*.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Emma Croxford, Yanjun Gao, Elliot First, Nicholas Pellegrino, Miranda Schnier, John Caskey, Madeline Oguss, Graham Wills, Guanhua Chen, Dmitriy Dligach, and 1 others. 2025. Evaluating clinical ai summaries with large language models as judges. *npj Digital Medicine*, 8(1):640.

Jennifer D'Souza, Hamed Babaei Giglou, and Quentin Münch. 2025. Yescieval: Robust llm-as-a-judge for scientific question answering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13749–13783.

Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. 2022. Translation between molecules and natural language. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 375–413.

Veronika Ganeeva, Kuzma Khrabrov, Artur Kadurin, and Elena Tutubalina. 2025. Measuring chemical llm robustness to molecular representations: a smiles variation-based framework. *Journal of Cheminformatics*, 17(1):1–15.

Google DeepMind. 2026. Gemini 3.1 Pro model card. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-1-Pro-Model-Card.pdf.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Taicheng Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh Chawla, Olaf Wiest, Xiangliang Zhang, and 1 others. 2023. What can large language models do in chemistry? a comprehensive benchmark on eight tasks. *Advances in neural information processing systems*, 36:59662–59688.

Kilem Li Gwet. 2008. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Heegyu Kim, Taeyang Jeon, Seungtaek Choi, Ji Hoon Hong, Dong Won Jeon, Ga-Yeon Baek, Gyeong-Won Kwak, Dong-Hee Lee, Jisu Bae, Chihoon Lee, and 1 others. 2025. Towards fully-automated materials discovery via large-scale synthesis dataset and expert-level llm-as-a-judge. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 1302–1312.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4334–4353.

Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, and 1 others. 2023. Pubchem 2023 update. *Nucleic acids research*, 51(D1):D1373–D1380.

Md Tahmid Rahman Laskar, Israt Jahan, Elham Dolatabadi, Chun Peng, Enamul Hoque, and Jimmy Huang. 2025. Improving automatic evaluation of large language models (llms) in biomedical relation extraction via llms-as-the-judge. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25483–25497.

Seongyun Lee, Seungone Kim, Sue Park, Geewook Kim, and Minjoon Seo. 2024. Prometheus-vision: Vision-language model as a judge for fine-grained evaluation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11286–11315.

Hao Li, He Cao, Bin Feng, Yanjun Shao, Xiangru Tang, Zhiyuan Yan, Li Yuan, Yonghong Tian, and Yu Li. 2025. Beyond chemical qa: Evaluating llm's chemical reasoning with modular chemical operations. *arXiv preprint arXiv:2505.21318*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tzyy-Shyang Lin, Connor W Coley, Hidenobu Mochigase, Haley K Beech, Wencong Wang, Zi Wang, Eliot Woods, Stephen L Craig, Jeremiah A Johnson, Julia A Kalow, and 1 others. 2019. Bigsmiles: a structurally-based line notation for describing macromolecules. *ACS central science*, 5(9):1523–1531.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 2511–2522.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. 2024. Augmenting large language models with chemistry tools. *Nature machine intelligence*, 6(5):525–535.

Arash Marioriyad, Omid Ghahroodi, Ehsaneddin Asgari, Mohammad Hossein Rohban, and Mahdieh Soleymani Baghshah. 2026. The judge who never admits: Hidden shortcuts in llm-based evaluation. *arXiv preprint arXiv:2602.07996*.

Adrian Mirza, Nawaf Alampara, Sreekanth Kunchapu, Martiño Ríos-García, Benedict Emoekabu, Aswanth Krishnan, Tanya Gupta, Mara Schilling-Wilhelmi, Macjonathan Okereke, Anagha Aneesh, and 1 others. 2024. Are large language models superhuman chemists? *arXiv preprint arXiv:2404.01475*.

Adrian Mirza, Nawaf Alampara, Sreekanth Kunchapu, Martiño Ríos-García, Benedict Emoekabu, Aswanth Krishnan, Tanya Gupta, Mara Schilling-Wilhelmi, Macjonathan Okereke, Anagha Aneesh, and 1 others. 2025. A framework for evaluating the chemical knowledge and reasoning abilities of large language models against the expertise of chemists. *Nature Chemistry*, 17(7):1027–1034.

Jiwon Moon, Yerin Hwang, Dongryeol Lee, Taegwan Kang, Yongil Kim, and Kyomin Jung. 2025. Don't judge code by its cover: Exploring biases in llm judges for code evaluation. *arXiv preprint arXiv:2505.16222*.

Milad Moradi and Matthias Samwald. 2022. Improving the robustness and accuracy of biomedical language models through adversarial training. *Journal of Biomedical Informatics*, 132:104114.

Heewoong Noh, Gyoung S Na, Namkyeong Lee, and Chanyoung Park. 2026. Msp-llm: A unified large language model framework for complete material synthesis planning. *arXiv preprint arXiv:2602.07543*.

OpenAI. 2025. Introducing GPT-4.1 in the API.

OpenAI. 2025. OpenAI o3 and o4-mini system card. https://cdn.openai.com/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Ricardo Rei, José GC De Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André FT Martins. 2022. Comet-22: Unbabel-ist 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585.

Nicholas T Runcie, Charlotte M Deane, and Fergus Imrie. 2025. Assessing the chemical intelligence of large language models. *Journal of Chemical Information and Modeling*, 66(1):216–227.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, and 1 others. 2025. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*.

Andreas Stephan, Dawei Zhu, Matthias Aßenmacher, Xiaoyu Shen, and Benjamin Roth. 2025. From calculation to adjudication: Examining llm judges on mathematical reasoning tasks. In *Proceedings of the Fourth Workshop on Generation, Evaluation and Metrics (GEM^2)*, pages 759–773.

Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar, Tanjin He, David Milsted, Matthew J McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, and 1 others. 2023. An autonomous laboratory for the accelerated synthesis of inorganic materials. *Nature*, 624(7990):86.

Weixi Tong and Tianyi Zhang. 2024. Codejudge: Evaluating code generation with large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20032–20051.

Prapti Trivedi, Aditya Gulati, Oliver Molenschot, Meghana Arakkal Rajeev, Rajkumar Ramamurthy, Keith Stevens, Tanveesh Singh Chaudhery, Jahnavi Jambholkar, James Zou, and Nazneen Rajani. 2024. Self-rationalization improves llm as a fine-grained judge. *arXiv preprint arXiv:2410.05495*.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965.

Minghao Wu and Alham Fikri Aji. 2025. Style over substance: Evaluation biases for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 297–312.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2025. Evaluating mathematical reasoning beyond accuracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27723–27730.

Bing Yan, Angelica Chen, and Kyunghyun Cho. 2025. Inconsistency of llms in molecular representations. *Digital Discovery*, 4(10):2876–2892.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, and 1 others. 2024a. Justice or prejudice? quantifying biases in llm-as-a-judge. *arXiv preprint arXiv:2410.02736*.

Ziyi Ye, Xiangsheng Li, Qiuchi Li, Qingyao Ai, Yujia Zhou, Wei Shen, Dong Yan, and Yiqun Liu. 2024b. Beyond scalar reward model: Learning generative judge from preference data. *arXiv preprint arXiv:2410.03742*.

Mohd Zaki, NM Anoop Krishnan, and 1 others. 2024. Mascqa: investigating materials science knowledge of large language models. *Digital Discovery*, 3(2):313–327.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. 2023. Evaluating large language models at evaluating instruction following. *arXiv preprint arXiv:2310.07641*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and

chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Terry Yue Zhuo. 2024. Ice-score: Instructing large language models to evaluate code. In *Findings of the association for computational linguistics: EACL 2024*, pages 2232–2242.

A Task Setup

This appendix details the evaluation rubric, judge prompt templates, and model configurations used in our experiments.

A.1 Evaluation Rubric

We adopt the evaluation rubric of Alchemy-Bench (Kim et al., 2025) with minor modifications for the reference-free regime. The rubric organizes a synthesis recipe into four components—materials, equipment, procedure, and characterization—and the judge rates the candidate on a 1–5 scale in 0.5 increments. The reference-free mode evaluates five criteria (materials appropriateness, equipment appropriateness, procedure completeness, procedure feasibility, and characterization appropriateness) and an overall score.

A.2 Judge Prompts

Each prompt assigns the judge the role of an expert materials scientist and provides (1) the target material description and (2) the candidate recipe. The judge is instructed to produce step-by-step reasoning followed by a JSON object containing scores for each criterion. All scores use a 1–5 scale with 0.5 increments. The complete prompt template is shown in Figure 7. JudgeLRM-7B (Chen et al., 2025) uses a separate prompt tailored to its native format, which elicits a single overall score on a 1–10 scale within structured `<think>/<answer>` tags. Scores are linearly rescaled to the 1–5 range used by all other models.

A.3 Judge Models

Table 5 lists the thirteen baseline judge models evaluated in §4, grouped by category. Models served via OpenRouter use greedy decoding (temperature 0). Models served via vLLM use temperature 0.6, the engine default.

Table 5: Judge models evaluated in this work.

Category	Model
Instruct	Llama-3.1-8B (Grattafiori et al., 2024) Llama-3.3-70B Qwen3-8B (Yang et al., 2025) Qwen3-32B GPT-4o (Hurst et al., 2024) Claude-Sonnet-4.5 (Anthropic, 2025)
Reasoning	Qwen3-8B-Think Qwen3-32B-Think o4-mini (OpenAI, 2025) GPT-5-mini (Singh et al., 2025) Gemini-3.1-Pro (Google DeepMind, 2026)
Specialized	JudgeLRM-7B (Chen et al., 2025) Prometheus-7B-v2 (Kim et al., 2024)

B ASSAY Details

This appendix complements §3, providing the full NER prompts (§B.1), swap type implementations (§B.2), and label verification results (§B.3).

B.1 Materials Entity Recognition Prompts

The entity extraction step uses two NER prompts, one per augmentation mode, that share the same instruction format but extract disjoint entity types. The prompt for *Equivalence Rewriting* extracts the four representational entity types (`formula`, `compound_name`, `abbreviation`, `iupac_name`) and is shown in Figure 8; the prompt for *Error Injection* extracts the four content-bearing entity types (`element`, `numerical_value`, `equipment`, `action_verb`) and is shown in Figure 9. Both prompts use GPT-4.1-mini (OpenAI, 2025) with deterministic decoding with temperature of 0.

B.2 Swap Type Implementation

Each swap type takes as input the entities identified by the NER step and rewrites a subset of them into either a scientifically invalid alternative (*Error Injection*) or a semantically equivalent representation (*Equivalence Rewriting*). The subset to be rewritten is selected per recipe by the augmentation rate ρ, whose implementation is detailed at the end of this section. Outputs are deterministic given the same input recipe, NER cache, and random seed.

***Error Injection* swap types.** The four swap types are implemented as rule-based substitutions over typed entity slots, each backed by a hand-curated dictionary. `Element` replaces an element with another from the same periodic group (82 elements across 19 categories). `Numerical` scales a value by a unit-specific multiplier designed to push it outside physically feasible ranges (10 unit categories). `Equipment` replaces laboratory equipment with an incompatible alternative, with auxiliary abbreviation resolution (51 rules across 6 categories). `Action` replaces a procedural verb with its antonym, preserving inflection (55 verb pairs across 5 categories). Full dictionaries are provided in the released codebase.

***Equivalence Rewriting* swap types.** `Formula`, `Name`, and `IUPAC` share a common resolution pipeline. Each entity identified by the *Equiva-*

Reference-free judge prompt

You are an expert materials scientist tasked with evaluating an AI-generated synthesis recipe. You will be provided with:
1. Target material description
2. AI-generated recipe

Please evaluate the AI-generated recipe according to the following criteria on a scale of 1–5 (1.0: poor, 5.0: excellent, 0.5 step). Provide detailed justification for each score.

Guidelines for evaluation:
1. Materials — **Appropriateness**: Are the selected materials suitable for the target synthesis?
2. Equipment — **Appropriateness**: Is the selected equipment suitable?
3. Procedure — **Completeness**: Are all necessary steps included with sufficient detail? **Feasibility**: Can this procedure be realistically executed in a lab?
4. Characterization — **Appropriateness**: Are the metrics suitable for validating success?

Output format: Step-by-step reasoning first, then JSON score result:

```
{"materials_appropriateness_score": float,
"equipment_appropriateness_score": float,
"procedure_completeness_score": float,
"procedure_feasibility_score": float,
"characterization_appropriateness_score": float,
"overall_score": float}
```

Figure 7: Reference-free judge prompt used for all models except JudgeLRM-7B, which uses a dedicated prompt with a 1–10 scale and `<think>/<answer>` format.

lence Rewriting NER step is first queried against the PubChem REST API (Kim et al., 2023) for its molecular formula, IUPAC name, and common name. When the entity is an abbreviation accompanied by a parenthetical definition in the recipe text (e.g., *poly(ethylene oxide) (PEO)*), the parenthetical full name is queried preferentially to avoid mis-resolution of polymer abbreviations whose acronyms often clash with unrelated compounds. Entities for which PubChem (Kim et al., 2023) returns no result are forwarded to a general-purpose LLM, which returns the same target fields and is required to output `null` for any field on which it lacks confidence. Each candidate is validated before use: formulas must match a chemical-formula regex, IUPAC names must lie within a 3–120 character window, and any candidate identical to the original entity text is discarded.

`Cross-lingual`. This swap type rewrites the full recipe text from English into a target language using GPT-4.1-mini (OpenAI, 2025). The implementation supports ten target languages (Korean, Japanese, Chinese, German, French, Spanish, Portuguese, Russian, Arabic, Hindi); all experiments in this paper use Korean as the target language. Unlike the other seven swap types, `Cross-lingual` does not consume the entity annotations from the NER step.

Augmentation rate. The body text of §3.3 introduces the augmentation rate $\rho \in (0, 1]$ as the fraction of identified entities augmented. Implementation-wise, ρ is realized through a token-level measure that more directly captures the magnitude of textual change. Define the unique token set of a recipe as the set of distinct alphanumeric tokens after case-folding. For an original recipe y^* and its augmented variant y', we measure

$$\rho_{\text{actual}} = \frac{|T(y^*) \setminus T(y')|}{|T(y^*)|}, \tag{8}$$

that is, the fraction of unique tokens present in y^* but absent from y'. To realize a requested ρ, the swap type iteratively selects entities and applies its rewriting rule until the accumulated token rate matches ρ within a tolerance of one token or 0.5 percent of the unique-token count, retrying at most three times under different random orderings if the tolerance is not met.

B.3 Label Verification

Protocol. Three materials science researchers independently annotated 595 augmented entities sampled across the eight swap types, with samples stratified by type to ensure balanced coverage. Each augmentation was presented alongside its corresponding ground-truth recipe and synthesis target. For *Error Injection* outputs, annotators

NER prompt for *Equivalence Rewriting*

You are a materials science NER system. Extract ALL chemical entities from the given synthesis recipe text.

For each entity, classify its type as one of:
- `"formula"`: Chemical formulas like H2O, SiO2, TiO2, NaCl, Fe3O4, CH3OH
- `"compound_name"`: Common chemical names like "ethanol", "titanium dioxide", "sodium hydroxide"
- `"abbreviation"`: Chemical abbreviations like "TEOS", "PVA", "CTAB", "DMF", "DMSO"
- `"iupac_name"`: IUPAC systematic names like "tetraethyl orthosilicate", "poly(vinyl alcohol)"

Rules:
- Extract every distinct chemical entity mentioned in the text
- Include elements when used as chemical substances (e.g., "gold", "silicon", "Ar" as argon gas)
- Include solvents, precursors, reagents, catalysts, dopants, substrates, and products
- DO NOT include: equipment names (furnace, autoclave), techniques (XRD, SEM, TEM), units (mL, mg, nm), or generic terms (solution, mixture, sample, substrate when not a specific material)
- DO NOT include process parameters (temperature, pressure, time values)
- If the same entity appears multiple times, include it only ONCE

Return a JSON object with a single key `"entities"` containing an array of objects, each with `"text"` (exact string as it appears) and `"type"` fields.

Example output:
```
{"entities": [{"text": "TiO2", "type": "formula"},
{"text": "ethanol", "type": "compound_name"},
{"text": "TEOS", "type": "abbreviation"}]}
```

Figure 8: NER prompt for *Equivalence Rewriting*. The prompt extracts the four representational entity types that serve as inputs to the `Formula`, `Name`, `IUPAC`, and `Cross-lingual` swap types.

judged whether the injected modification rendered the recipe scientifically invalid; for *Equivalence Rewriting* outputs, annotators judged whether the substituted representation preserved the chemical identity of the original entity. Annotation was binary in both cases.

Agreement metric. We report inter-annotator agreement using Gwet's AC1 (Gwet, 2008) rather than Fleiss' κ. The *Error Injection* labels exhibit a heavily skewed marginal distribution under which Fleiss' κ is known to underestimate agreement due to the prevalence paradox. Gwet's AC1 is designed to remain stable in this regime.

Per-type results. Table 6 reports observed agreement and AC1 for each of the eight swap types. On *Error Injection*, all four types achieve AC1 above 0.89, confirming that the intended labels are highly reliable. On *Equivalence Rewriting*, `Name` and `IUPAC` achieve high to moderate AC1, while `Formula` is the principal source of disagreement (AC1 = 0.30, observed agreement 69.7%), reflecting cases in which automated formula assignment produces non-equivalent representations for polymers and large organic molecules. We treat this as a documented limitation of the `Formula` swap type, as the affected outputs remain a minority and the residual disagreement is concentrated on a single type.

Mode	Swap Type	Agreement	AC1
Error Injection	`Element`	0.99	0.98
	`Numerical`	0.95	0.89
	`Equipment`	0.94	0.95
	`Action`	0.99	0.98
Equivalence Rewriting	`Name`	0.91	0.89
	`IUPAC`	0.79	0.49
	`Formula`	0.70	0.30

Table 6: Per-type inter-annotator agreement and Gwet's AC1. *Error Injection* shows consistently high agreement, while *Equivalence Rewriting* shows type-dependent variability, with `Formula` as the principal source of disagreement.

C Additional Results

C.1 Per-Rate Validity and Stability

Table 7 reports per-rate Validity and Stability scores underlying the AUC values in Figure 3. The pink gradient confirms that Validity increases monotonically with ρ for all models, while the blue gradient shows that Stability degrades more gradually, consistent with the asymmetric sensitivity discussed in §4.1.

NER prompt for *Error Injection*

You are a materials science NER system. Extract ALL relevant entities from the given synthesis recipe text.

For each entity, classify its type as one of:
- `"element"`: Chemical element symbols used as substances or in recognizable positions within formulas. Include elements at word boundaries or recognizable formula boundaries (e.g., "Li" in LiTFSI, "P" in P(EO)14, "Ti" in TiO2, "Na", "Fe", "Ar"). Do NOT extract elements buried inside compound formulas where they are inseparable (e.g., do not extract "O" from H2O or "Si" from SiO2). Focus on elements that could be independently substituted.
- `"numerical_value"`: Number+unit combinations representing measurable quantities. Include the number AND unit as a single entity. Examples: "500 °C", "10 hours", "5 mg", "0.1 M", "20 nm", "3000 rpm", "5%", "2 mL", "150 W", "1 atm".
- `"equipment"`: Laboratory equipment, instruments, and apparatus. Examples: "furnace", "autoclave", "centrifuge", "SEM", "XRD", "TGA", "ball mill", "glovebox", "hot plate", "magnetic stirrer", "oven", "mortar", "pestle".
- `"action_verb"`: Synthesis procedure verbs in their exact inflected form as they appear in the text. Examples: "heated", "stirring", "dissolved", "calcined", "dried", "added", "mixed", "annealed", "sintered", "washed", "filtered", "cooled", "ground", "dispersed".

Rules:
- Extract every distinct entity mentioned in the text
- If the same entity text appears multiple times, include it only ONCE
- For numerical_value: always include both the number and its unit together as one entity
- For action_verb: extract the verb in its exact form (past tense, gerund, etc.)
- For element: only extract when the element symbol is clearly identifiable and could be substituted

Return a JSON object with a single key `"entities"` containing an array of objects, each with `"text"` (exact string as it appears) and `"type"` fields.

Example output:
```
{"entities": [{"text": "Ti", "type": "element"},
{"text": "500 °C", "type": "numerical_value"},
{"text": "furnace", "type": "equipment"},
{"text": "heated", "type": "action_verb"}]}
```

Figure 9: NER prompt for *Error Injection*. The prompt extracts the four content-bearing entity types that serve as inputs to the `Element`, `Numerical`, `Equipment`, and `Action` swap types.

C.2 Per-Model Per-Type Breakdown

Table 8 reports detection and stability rates for each of the thirteen judge models at full per-type augmentation, expanding the category averages in Table 2. On *Error Injection*, `Action` yields the lowest detection rate for 7 of 13 models, confirming that the pattern identified in §4.2 holds broadly, though individual models occasionally find other types more challenging. On *Equivalence Rewriting*, `Name` yields the highest stability rate for 7 of 13 models, while the remaining models favor `Cross-lingual` (5) or `Formula` (1), indicating that common-name tolerance is the most consistent but not universal pattern.

D Mitigation Training Details

This appendix provides implementation details for the contrastive pair construction and DPO training described in §5.1.

D.1 Judge Panel

The four judge models used for pair construction are Qwen3-8B, Qwen3-32B (Yang et al., 2025), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), and Gemini-2.5-Flash (Comanici et al., 2025). All four models receive the reference-free judge prompt (Figure 7) and produce rationale-and-score responses. Qwen3-8B, Qwen3-32B, and Llama-3.1-8B-Instruct are served via vLLM (temperature 0.6); Gemini-2.5-Flash is served via API (temperature 0).

D.2 Augmentation Configurations

Contrastive pairs are drawn from 17 augmentation settings applied to 2,000 recipes sampled from the AlchemyBench training set. For *Error Injection*, we use five combined-rate settings ($\rho \in \{0.01, 0.02, 0.05, 0.10, 0.15\}$), in which entities are sampled across all four swap types, plus four type-specific settings that replace all identified entities of a single type (`Element`, `Numerical`, `Equipment`, `Action`). For *Equivalence Rewrit-*

Table 7: Per-rate Validity and Stability for 13 baseline judges. Per-rate cells use a pink gradient for Validity and a blue gradient for Stability; darker shading indicates higher values. AUC columns correspond to the values reported in Figure 3.

	Model	Validity (↑)						Stability (↑)						Acc. (↑)
		ρ=.01	ρ=.02	ρ=.05	ρ=.10	ρ=.15	AUC	ρ=.01	ρ=.02	ρ=.05	ρ=.10	ρ=.15	AUC	AUC
Base	Llama-3.1-8B	0.12	0.15	0.18	0.21	0.28	0.20	0.79	0.77	0.76	0.80	0.74	0.77	0.45
	Llama-3.3-70B	0.10	0.10	0.16	0.33	0.43	0.26	0.90	0.90	0.88	0.86	0.86	0.87	0.53
	Qwen3-8B	0.14	0.12	0.22	0.36	0.53	0.31	0.84	0.82	0.85	0.70	0.70	0.76	0.51
	Qwen3-32B	0.11	0.17	0.28	0.43	0.57	0.36	0.84	0.87	0.81	0.77	0.67	0.78	0.55
	GPT-4o	0.14	0.15	0.32	0.57	0.65	0.44	0.91	0.86	0.80	0.71	0.67	0.76	0.59
	Claude-Sonnet-4.5	0.18	0.28	0.55	0.68	0.79	0.59	0.73	0.76	0.74	0.68	0.62	0.70	0.64
	Average (n=6)	0.13	0.16	0.28	0.43	0.54	0.36	0.83	0.83	0.80	0.75	0.71	0.77	0.55
Reasoning	Qwen3-8B-Think	0.27	0.31	0.50	0.64	0.72	0.55	0.58	0.57	0.54	0.44	0.49	0.50	0.54
	Qwen3-32B-Think	0.33	0.38	0.49	0.69	0.72	0.58	0.66	0.63	0.55	0.46	0.49	0.52	0.56
	o4-mini	0.28	0.46	0.67	0.82	0.86	**0.71**	0.54	0.51	0.47	0.43	0.30	0.43	0.60
	GPT-5-mini	0.18	0.27	0.51	0.72	0.88	0.60	0.80	0.77	0.77	0.68	0.61	0.71	0.66
	Gemini-3.1-Pro	0.19	0.33	0.66	0.86	0.92	**0.71**	0.86	0.88	0.78	0.75	0.73	**0.78**	**0.75**
	Average (n=5)	0.25	0.35	0.57	0.74	0.82	0.63	0.69	0.67	0.62	0.55	0.52	0.59	0.62
Spec.	JudgeLRM-7B	0.26	0.30	0.38	0.53	0.65	0.47	0.54	0.51	0.49	0.34	0.38	0.42	0.45
	Prometheus-7B-v2	0.35	0.35	0.34	0.38	0.38	0.36	0.32	0.29	0.34	0.34	0.33	0.33	0.35
	Average (n=2)	0.31	0.33	0.36	0.46	0.52	0.42	0.43	0.40	0.42	0.34	0.36	0.38	0.40

ing, we use the same five combined-rate settings plus three type-specific settings (`Formula`, `Name`, `IUPAC`). `Cross-lingual` is excluded from type-specific pair construction because full-text translation does not admit partial entity-level control. This yields 9 *Error Injection* settings and 8 *Equivalence Rewriting* settings for a total of 17.

D.3 Anchor Filtering Criteria

After constructing direction-aware preference pairs (§5.1), we apply three filters to remove low-quality pairs.

Score margin. Pairs whose absolute score difference falls below $\delta_{\min} = 0.5$ are discarded, ensuring that the preferred and dispreferred responses reflect a meaningful quality gap.

Anchor-relative bounds. Responses that deviate substantially from the anchor $a(x)$ (Equation 5) are excluded. The filtering logic is direction-specific. For *Equivalence Rewriting* pairs, the preferred response must satisfy $J_i(x, \hat{y}) \geq a(x) - 1.5$, ensuring that the chosen response remains near the consensus quality estimate and preventing pairs in which both responses assign anomalously low scores. For *Error Injection* pairs, the dispreferred response must satisfy $J_j(x, \tilde{y}) \geq a(x) - 2.0$, ensuring that the rejected response is not already a strong error detector and preserving a meaningful contrast.

Per-sample cap. Up to five pairs are retained per recipe per augmentation setting to prevent any single sample from dominating the training distribution.

D.4 DPO Training Hyperparameters

SynthJudge-8B is initialized from Qwen3-8B (Yang et al., 2025) and trained with Direct Preference Optimization (Rafailov et al., 2023). Table 9 lists the hyperparameters.

Computing Infrastructure. All open-weight models are served on a single NVIDIA A6000 48GB GPU via vLLM. SynthJudge-8B DPO training takes approximately 10 GPU hours on 8×A100 80GB. API-served models (GPT-4o, GPT-5-mini, o4-mini, Gemini-3.1-Pro, Claude-Sonnet-4.5, Gemini-2.5-Flash) are accessed via OpenRouter[1] with greedy decoding.

D.5 Knowledge Augmentation Baselines

Table 3 in §5.2 reports three knowledge augmentation (KA) variants applied to Qwen3-8B at inference time. This section details their implementation.

ChemDict. We prepend a chemistry dictionary to the judge prompt that maps all entities appearing in the candidate recipe to their canonical representations. Entities are identified using both the

[1] `https://openrouter.ai`

Table 8: Per-model detection rate (*Error Injection*, left) and stability rate (*Equivalence Rewriting*, right) at full per-type augmentation. Shading is normalized per row; darker cells indicate higher values within each model.

		Error Injection ($\tilde{y}$)				***Equivalence Rewriting*** ($\hat{y}$)			
Category	Model	Elem.	Num.	Equip.	Action	Form.	Name	IUPAC	Cross.
Base	Llama-3.1-8B	.557	.548	.554	.575	.768	.816	.810	.777
	Llama-3.3-70B	.649	.579	.485	.563	.892	.890	.868	.851
	Qwen3-8B	.479	.527	.556	.544	.757	.781	.716	.799
	Qwen3-32B	.527	.509	.592	.485	.787	.775	.740	.793
	GPT-4o	.580	.621	.556	.574	.834	.845	.750	.935
	Claude-Sonnet-4.5	.583	.659	.679	.560	.732	.749	.655	.677
	Average	.562	.574	.570	.550	.795	.809	.756	.805
Reas.	Qwen3-8B-Think	.598	.568	.527	.598	.509	.669	.592	.521
	Qwen3-32B-Think	.669	.627	.645	.568	.562	.592	.556	.574
	o4-mini	.604	.728	.663	.550	.426	.515	.462	.550
	GPT-5-mini	.746	.769	.716	.568	.702	.804	.686	.798
	Gemini-3.1-Pro	.716	.793	.789	.708	.667	.775	.696	.772
	Average	.666	.697	.668	.598	.573	.671	.598	.643
Spec.	JudgeLRM-7B	.331	.331	.355	.308	.420	.515	.408	.521
	Prometheus-7B-v2	.259	.360	.290	.309	.259	.288	.280	.222
	Average	.295	.346	.323	.308	.340	.401	.344	.371

Table 9: DPO training hyperparameters for **SynthJudge-8B**.

Hyperparameter	Value
Base model	Qwen3-8B
DPO β	0.1
Learning rate	1e-6
LR scheduler	cosine
Warmup ratio	0.1
Batch size	16
Max sequence length	8,192
Training epochs	1
Optimizer	AdamW
Training pairs	92K train / 10K val

Equivalence Rewriting and *Error Injection* NER prompts (Appendix B.1), and the dictionary is organized into four sections. Compound entries map chemical names to their molecular formulas, IUPAC systematic names, and common names using cached PubChem data (Kim et al., 2023). Element entries provide periodic-group membership, equipment entries list incompatible alternatives, and action entries list antonym verbs. This provides the judge with explicit equivalence and error-relevant information for all identified entities.

RAG. We retrieve the five most similar recipes from the Open Materials Guide (OMG) training set (16,026 recipes) (Kim et al., 2025) using cosine similarity over `text-embedding-3-large` [2] embeddings. The query consists of the target material description only. Retrieved recipes are prepended to the judge prompt as in-context references, providing domain-specific examples of valid synthesis procedures without a ground-truth recipe for the specific target.

CD+RAG. This variant combines both augmentations by prepending the chemistry dictionary followed by the retrieved recipes to the judge prompt.

All three variants use the same Qwen3-8B model, decoding configuration, and scoring rubric as the unaugmented baseline.

D.6 Case Studies

Figure 10 presents two representative cases illustrating how **SynthJudge-8B** resolves both failure modes at the individual recipe level. Case A demonstrates false negative resolution, where baseline judges penalize notation-level changes as content deficiencies while **SynthJudge-8B** correctly preserves its score. Case B demonstrates false positive resolution, where baseline judges fail to detect scientifically infeasible parameter values while **SynthJudge-8B** identifies specific physical and chemical inconsistencies.

Case A: False negative resolution. *Equivalence Rewriting* converts compound names and chemical identifiers into alternative representations in a graphene synthesis recipe (e.g., liquid nitrogen $\rightarrow$ N_2). All four baseline judges penalize the rewriting as a content deficiency, with score drops ranging

[2] https://platform.openai.com/docs/guides/embeddings

Case A: False Negative Resolution (Stability) | *Graphene* | Electronics / Hybrid

Ground truth ($s = 4.5$): *"... **Graphite powder** was exposed to microwave irradiation... quenched using **liquid nitrogen**... ultrasonication in **IPA** for 90 minutes..."*

Equivalence Rewriting ($\rho = 0.10$): IPA → C_3H_8O , liquid nitrogen → N_2 , graphene → C

Model	Ground Truth	Equiv. Rewriting	Stable?
SynthJudge-8B	4.5	4.3 ✓	$\lvert\Delta\rvert = 0.2$
Gemini-3.1-Pro	—	2.5 ✗	$\lvert\Delta\rvert = 2.0$
GPT-4o	—	3.0 ✗	$\lvert\Delta\rvert = 1.5$
Claude-Sonnet	—	2.5 ✗	$\lvert\Delta\rvert = 2.0$
Qwen3-32B	—	3.5 ✗	$\lvert\Delta\rvert = 1.0$

SynthJudge-8B: "well-structured... suitable materials, equipment, and characterization methods."
Gemini: "the recipe uses molecular formulas (C_3H_8O, N_2) instead of chemical names... hindering readability and verification."

Case B: False Positive Resolution (Validity) | *Iron Tin (FeSn)* | Electronics / MBE

Ground truth ($s = 3.5$): *"... Degas by heating to **400 °C** for **60 min**. Anneal at **∼1150 °C**... for **60 min**... substrate maintained at **∼530 °C**..."*

Error Injection (Numerical): 400 → 600 °C , 60 → 600 min , ∼1150 → ∼1725 °C , ∼530 → ∼265 °C

Model	Ground Truth	Error Injection	Detected?
SynthJudge-8B	3.5	2.2 ✓	$\Delta = 1.3$
Gemini-3.1-Pro	—	3.6 ✗	$\Delta = -0.1$
GPT-4o	—	4.1 ✗	$\Delta = -0.6$
Claude-Sonnet	—	3.5 ✗	$\Delta = 0.0$
Qwen3-32B	—	4.5 ✗	$\Delta = -1.0$

SynthJudge-8B: "1725°C exceeds the melting point of $SrTiO_3$... 600 minutes is impractically long for degassing... 265°C is too low for epitaxial growth."
GPT-4o: "the materials selected are appropriate... the equipment listed is suitable... the procedure is feasible."

Figure 10: Two case studies comparing **SynthJudge-8B** and baseline judges. **Case A** (Stability): *Equivalence Rewriting* converts compound names to molecular formulas. **SynthJudge-8B** correctly preserves its score ($|\Delta| = 0.2$), recognizing notation-level changes, while all four baselines penalize the rewriting as a content deficiency (false negatives). **Case B** (Validity): *Error Injection* introduces scientifically infeasible temperatures and durations. **SynthJudge-8B** detects the errors ($\Delta = 1.3$), while all four baselines assign scores equal to or higher than the ground truth (false positives).

from 1.0 to 2.0. **SynthJudge-8B** correctly recognizes these as notation-level changes and preserves its score ($|\Delta| = 0.2$).

Case B: False positive resolution. *Error Injection* introduces scientifically infeasible numerical values into an iron tin (FeSn) MBE synthesis recipe, including a degassing temperature of 600°C (originally 400°C), a duration of 600 minutes (originally 60), and a substrate temperature halved to 265°C. All four baseline judges assign scores equal to or higher than the ground truth, failing to detect any of the injected errors. **SynthJudge-8B** detects the errors ($\Delta = 1.3$), correctly identifying that 1725°C exceeds the melting point of $SrTiO_3$, that 600 minutes is impractically long for degassing, and that 265°C is too low for epitaxial growth.

D.7 Full Human Expert Agreement

Table 10 extends Table 4 by reporting Pearson r with both a high-confidence expert panel (HC, $n = 3$) and the full expert panel (Full, $n = 8$) for all fourteen evaluated models on 20 reference-free evaluations. The high-confidence panel consists of three senior materials science researchers with the highest self-reported confidence; the full panel includes five additional annotators with varying expertise levels.

Agreement varies substantially across models

Table 10: Pearson r with expert panels on reference-free evaluations. HC = high-confidence panel ($n = 3$, expert mean $= 4.21$); Full = full panel ($n = 8$, expert mean $= 3.32$).

Category	Model	HC r	Full r
Base	GPT-4o	.416	.374
	Qwen3-8B	.405	.021
	Llama-3.3-70B	.356	**.408**
	Claude-Sonnet-4.5	.070	.124
	Qwen3-32B	.041	.062
	Llama-3.1-8B	$-.107$	.046
Reasoning	Qwen3-8B-Think	**.537**	.391
	GPT-5-mini	.222	.292
	Gemini-3.1-Pro	.163	.339
	Qwen3-32B-Think	.149	$-.014$
	o4-mini	.117	.098
Specialized	JudgeLRM-7B	.467	.372
	Prometheus-7B-v2	.286	$-.041$
Ours	**SynthJudge-8B**	.412	.216

and panels. With the high-confidence panel, Qwen3-8B-Think achieves the highest correlation ($r = 0.537$), followed by JudgeLRM-7B ($r = 0.467$) and GPT-4o ($r = 0.416$). **SynthJudge-8B** achieves $r = 0.412$, comparable to GPT-4o and the base Qwen3-8B ($r = 0.405$). Agreement is consistently lower with the full panel, as the inclusion of stricter annotators lowers the expert mean score from 4.21 to 3.32, widening the gap with judge scores that tend toward the upper end of the 1–5 scale. Notably, the ranking shifts across panels for several models. Gemini-3.1-Pro rises from $r = 0.163$ (HC) to $r = 0.339$ (Full), as its relatively low judge mean (3.38) is better calibrated with the stricter expert panel, while models with high score inflation such as Qwen3-8B ($r = 0.405 \rightarrow 0.021$) and Prometheus-7B-v2 ($r = 0.286 \rightarrow -0.041$) degrade sharply.