Survey on Process Supervision for Multi-Step Reasoning: A Structured Review of Process Reward Models

Amir Malekhosseini, Mohammadmahdi Nouriborji, Nazanin Yousefi,
Danial Parnian, Shaygan Adim

Abstract

Large language models continue to struggle with complex, multi-step mathematical and logical reasoning because errors made early in a solution silently propagate to the final answer. *Process supervision* addresses this by scoring the correctness of every intermediate reasoning step rather than only the final outcome, giving rise to *Process Reward Models* (PRMs). This survey reviews the resulting literature along four axes that jointly determine whether a PRM is usable in practice: (i) *annotation reliability*, by re-examining whether Monte Carlo (MC) rollouts and Best-of-N evaluation actually measure what we think they measure; (ii) *generalization*, by training PRMs that are policy-agnostic and robust to the reflective, self-correcting behavior of long chain-of-thought (CoT) models; (iii) *scope*, by extending PRMs beyond single-policy mathematics to multimodal reasoning and open-domain, label-free settings bootstrapped from outcome reward models; and (iv) *grounding*, by replacing the noisy, generic MC proxy with domain-native verification signals – unit-test pass rates, retrieved clinical guidelines, scientific tool execution, optimization solvers, and deterministic rule-based decision procedures. We organize the literature into eight categories spanning the full PRM lifecycle, from human annotation to automated labeling, generalization-aware training, implicit/RL-based supervision, generative verification, architectural innovations, domain-specialized and verifiable supervision, and benchmarking. For each category we distill the dominant methods, their shared limitations, and open directions. We further contribute a taxonomy figure, a PRM development-loop diagram, a spectrum of supervision signals, and comparison tables/figures that make the trade-offs between annotation cost, reward noise, out-of-distribution generalization, and domain portability explicit.

1 Introduction

While increasing model scale has consistently yielded performance gains across a wide array of natural language processing tasks, multi-step mathematical reasoning remains a distinct vulnerability for state-of-the-art language models [Cobbe et al., 2021]. Because standard autoregressive models generate solutions sequentially without an inherent mechanism to self-correct, early logical errors quickly derail the entire reasoning trajectory [Lightman et al., 2023]. To overcome the limitations of pure generation, researchers introduced verification techniques during the decoding phase, wherein a separate reward model ranks multiple candidate solutions and selects the optimal path [Cobbe et al., 2021].

Historically, these verifiers functioned as *outcome reward models* (ORMs) that only evaluated the correctness of the final answer [Uesato et al., 2022]. However, outcome supervision struggles with credit assignment in lengthy reasoning chains and frequently rewards spurious solutions that arrive at the correct answer through flawed logic [Lightman et al., 2023]. To address these flaws, the field increasingly adopted *process supervision*, which trains models to evaluate the correctness of every intermediate step [Lightman et al., 2023]. Despite its empirical success, the reliance on expensive, domain-expert human annotation severely bottlenecked the scalability of early PRMs [Wang et al., 2023].

This survey covers the resulting wave of automation, but it also foregrounds a second-generation set of concerns that only became visible once automated PRMs were deployed at scale: *is the automated label actually correct, does the resulting PRM generalize beyond the policy and reasoning style it was trained on, and can process supervision be extended cheaply to modalities and domains where step-level labels are hard to define?* One group of papers speaks directly to these questions – a systematic critique of Monte Carlo (MC) annotation and Best-of-N (BoN) evaluation [Zhang et al., 2025b]; a universal, policy-agnostic PRM training

framework [Tan et al., 2025]; a step-level self-rewarding paradigm for math reasoning [Zhang et al., 2025c]; a multimodal PRM with a matching benchmark [Wang et al., 2025]; a reasoning-driven ("think-then-judge") PRM [She et al., 2025]; a re-annotation scheme for reflective, long-CoT reasoning [Yang et al., 2025]; and an open-domain PRM bootstrapped from existing ORMs without new step labels [Zhang et al., 2025d].

A second group of five papers changes the terms of the discussion again. Rather than trying to make the *generic* MC-rollout signal less noisy, each of them replaces or augments it with a *domain-native verification signal* that is externally checkable: code execution and unit-test pass rates for program synthesis [Li et al., 2025]; documents retrieved from clinical guidelines and textbooks for medical reasoning [Yun et al., 2025]; real scientific tool execution – sandboxed code, molecular libraries, literature retrieval – for scientific reasoning [Zhao et al., 2026]; an external optimization solver for operations research [Zhou et al., 2025]; and a deterministic, guideline-derived decision procedure for structured risk assessment [Pronesti et al., 2026]. Together they suggest that many of the pathologies documented in §4.2 – label noise, first-error truncation, reward hacking, false negatives on sound-but-unlucky steps – are not intrinsic to process supervision but artifacts of using answer-reachability as a stand-in for step validity.

Structurally, this survey groups the literature into eight method families, attaches an explicit *Limitations & Future Directions* box to each, and supports the discussion with a taxonomy figure, a PRM development-loop figure, a supervision-signal spectrum, and comparison tables. Papers that extend the classical mathematical PRM setting along any of the four axes above are marked * on first mention within each subsection.

2 Background

In multi-step problem solving, an ORM assigns a single scalar score to an entire solution sequence based on the validity of its final answer [Uesato et al., 2022]. A PRM instead evaluates an incremental reasoning path, predicting the correctness or value of *each* step [Uesato et al., 2022]. Formally, given a query x and a partial solution prefix $s_{1:t}$, a discriminative PRM parameterized by θ outputs

$$r_t = \sigma\big(f_\theta(x, s_{1:t})\big) \in (0, 1), \tag{1}$$

trained with a pointwise (cross-entropy/MSE) or pairwise preference loss over step labels y_t. PRMs are used in two primary settings: *test-time verification*, where they re-rank candidate solutions (e.g., Best-of-N, beam search), and *reinforcement learning*, where they supply dense, step-level rewards to optimize a policy via PPO or GRPO [Wang et al., 2023, Cui et al., 2025].

To automate step labeling without human annotation, most pipelines estimate the expected future accuracy of a step by sampling multiple completions (*rollouts*) and checking how often they reach the correct final answer [Luo et al., 2024]. This frames reasoning as a Markov Decision Process (MDP), where a step's value is its Monte Carlo (MC) estimate of the Q-value – the empirical probability that continuing from this prefix yields a correct outcome [Li and Li, 2024]. As we discuss in Section 4.2, this MC-estimated quantity is subtly *not* the same thing as "is this step logically valid" – a distinction that motivates a large fraction of the papers surveyed here. Forward-looking Q-values can further be combined with backward-looking step-correctness scores for bidirectional supervision [Chen et al., 2025], and advantage-style formulations score a step by its *relative* improvement in success probability rather than its absolute value [Setlur et al., 2024].

A further complication, largely unaddressed by first-generation PRMs, is that reasoning traces from long-CoT models frequently contain *reflection*: a step may be wrong, but a later step notices and corrects it. Conventional annotation, which truncates supervision at the first error under the assumption that everything afterward is also wrong, is misspecified for this behavior [Yang et al., 2025]. We revisit this issue in Section 4.3.

Finally, a distinct line of work asks not how to *denoise* the MC signal but whether it is the right signal at all. In domains where an intermediate claim can be checked against an external artifact – a test suite, a clinical guideline, a tool output, a solver status, or a written decision rule – the step label can be obtained by *verification* rather than by *estimation*. Figure 3 places these signal sources on a common spectrum, and Section 4.7 reviews the methods that exploit them.

3 Methodology and Taxonomy

We organize the literature into eight categories that together span the PRM lifecycle: (§4.1) foundational human-annotated supervision; (§4.2) automated discriminative supervision and its annotation pitfalls; (§4.3) generalization across policies and reflective long-CoT reasoning; (§4.4) implicit process rewards and reinforcement learning; (§4.5) generative verification and test-time scaling; (§4.6) advanced reward architectures together with multimodal and open-domain extensions; (§4.7) domain-specialized and verifiable process supervision; and (§4.8) benchmarking. Figure 1 maps the papers reviewed onto this taxonomy, Figure 2 situates the eight categories within the closed loop of *generating data* → *training a PRM* → *using the PRM* → *producing better data/policies*, and Figure 3 orders the supervision signals used across all categories by how directly they measure step validity. Tables 1 and 2 summarize the twelve papers that extend the classical setting, the category each belongs to, its core mechanism, and its headline result.

Table 1: Seven papers addressing annotation reliability, generalization, and scope.

Paper	Category	Core Mechanism	Headline Result
Zhang et al. [2025b]	§4.2	Consensus-filters MC rollouts against an LLM judge; hard labels; audits BoN evaluation for policy–PRM misalignment, score inflation, and process-to-outcome shift.	ProcessBench avg. F1 73.5/78.3 (7B/72B); BoN@8 acc. 67.6/69.3, beating majority voting on 7/7 tasks.
Tan et al. [2025]	§4.3	Samples outputs from diverse policies × prompts; ensembles multiple LLM-judge prompting strategies; reverse verification against the (masked) ground-truth answer.	74.3 F1 on ProcessBench, 74.5 F1 on the new UniversalBench (largest margin on long-CoT/OOD policies).
Yang et al. [2025]	§4.3	Introduces *Error Propagation/Error Cessation* rules for reflective CoT; injects them into a reflective LLM judge (o1) for annotation; trains on all post-error steps.	96.3% human-verified annotation accuracy (vs. 66.8% GPT-4o); PRM@64 = 0.816 on MATH500 (best); OOD gain grows with difficulty (+8.0% on AIME24).
Zhang et al. [2025c]	§4.4	Step-level pairwise self-judging (more reliable than absolute scoring); iterative DPO on self-generated preference data, no new external labels per round.	Judge accuracy stays 92–96% across 4 rounds; test-time scaling gap widens from 55.9→58.2 (round 1) to 60.6→62.4 (round 4).
Wang et al. [2025]	§4.6	MC-rollout labeling over multimodal solutions (VisualPRM400K); value-based multi-turn scoring; new VisualProcessBench with all-step human labels.	Lifts every tested MLLM policy by 3.7–8.4 pts under BoN; 62.0 F1 on VisualProcessBench (beats GPT-4o); transfers to text-only GPQA (+6.6).
She et al. [2025]	§4.5	PRM generates a 4-angle analysis before a judgment; SFT on filtered (analysis, judgment) tuples; DPO on self-sampled trajectories; K-way inference-time averaging.	ProcessBench F1 65.2 → 70.4 (SFT → DPO), beating a PRM trained on 4× more data; PRMBench 66.8.
Zhang et al. [2025d]	§4.6	Bootstraps a PRM from existing ORMs with *no* new step labels, via process-level preference trees built from repeated sampling and shared response prefixes.	RewardBench avg. 91.1 (+3.5 over base ORM), +12.1 Chat-Hard, +7.1 Reasoning; length-bias correlation drops from 0.37 to 0.05.

4 Review of Key Papers

4.1 Foundational Human-Annotated Process Supervision

Initial efforts to enhance reasoning through verification demonstrated that training models to evaluate solutions step-by-step significantly outperforms solution-level scoring [Cobbe et al., 2021]. To rigorously compare process and outcome supervision, Uesato et al. [2022] conducted extensive evaluations on grade-school math, finding that while outcome supervision achieves similar final-answer error rates, it suffers from severe *trace*

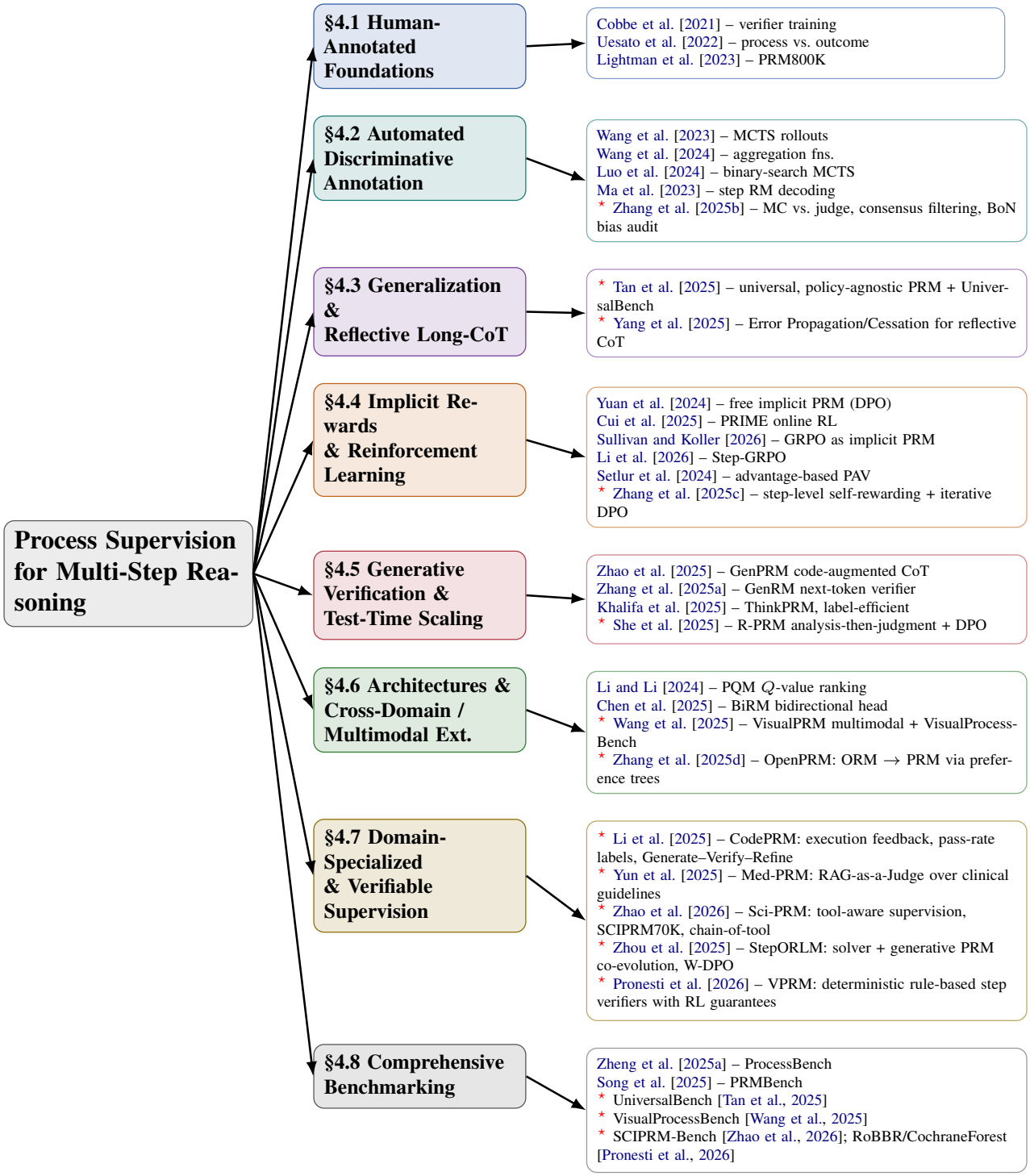

Figure 1: Taxonomy of the surveyed literature. Papers marked ⋆ extend the classical single-policy mathematical PRM setting along one of the four axes discussed in §1: annotation reliability, generalization, scope, or grounding.

errors in which models hallucinate logic to reach correct final numbers; process supervision drastically reduces these trace errors, though outcome models retain some ability to emulate process-level feedback internally. Scaling this paradigm to the more difficult MATH dataset, Lightman et al. [2023] released PRM800K, hundreds of thousands of human step annotations, and showed that process supervision decisively outperforms outcome

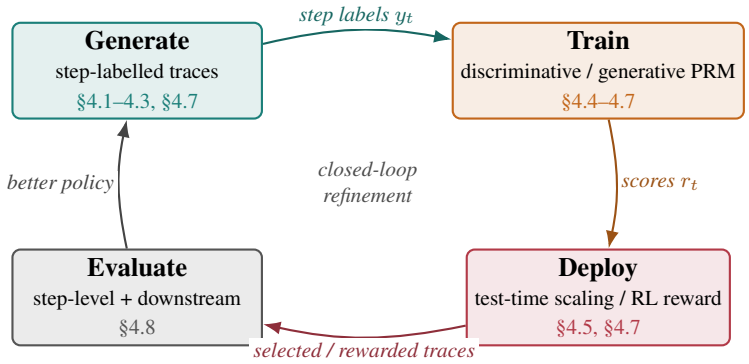

Figure 2: The PRM development loop. Each method family contributes to one or more stages. Most work occupies a single stage; the domain-specialized methods of §4.7 are unusual in touching three at once, since a domain verifier simultaneously changes how step labels are produced, what the PRM is trained against, and – in the self-evolving case [Zhou et al., 2025] – how the loop is closed.

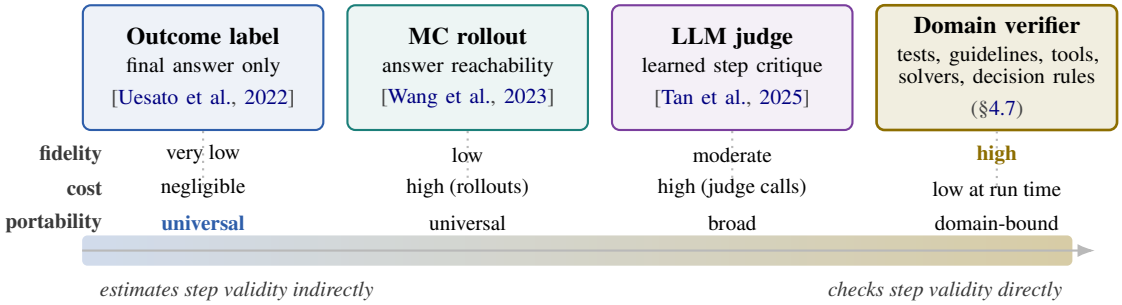

Figure 3: A spectrum of process-supervision signals. Moving right, the signal measures step validity more directly and becomes externally auditable, but requires a domain in which a checkable artifact exists. The papers in §4.7 all occupy the right-hand end, and most retain an outcome term as well, so the two ends of the spectrum are complementary rather than mutually exclusive.

supervision and actively penalizes misaligned, deceptive reasoning (incurring a small negative alignment tax).

Limitations & Future Directions

Human annotation remains the gold standard for label fidelity, but it does not scale: PRM800K required hundreds of thousands of expert judgments for a single domain, and the annotation burden grows with problem difficulty precisely where supervision is most needed. As policy models increasingly approach or exceed human problem-solving ability, human annotators risk becoming an unreliable oracle for evaluating novel but valid reasoning, foreshadowing a "superalignment" bottleneck. This motivates two complementary lines of future work: (i) automated or semi-automated pipelines (§4.2–§4.3) that trade some label fidelity for scale, and (ii) *AI-assisted* human oversight, where annotators verify the output of formal tools (code interpreters, theorem provers, guideline macros) rather than raw reasoning, decoupling supervision quality from human cognitive load – an idea that §4.7 pushes to its logical conclusion by removing the human from the inner loop entirely wherever a deterministic verifier exists.

4.2 Automated Discriminative Process Supervision

To eliminate the cost of human labeling, Math-Shepherd [Wang et al., 2023] uses Monte Carlo Tree Search (MCTS) rollouts to assign correctness scores based on a step's empirical potential to reach the correct final answer, and applies this automated data to both re-ranking and step-by-step RL. Wang et al. [2024] show that LLM-generated rollouts introduce significant noise and that aggregation functions favoring high predicted

Table 2: Five domain-specialized papers (§4.7). The third column identifies the externally checkable artifact that replaces or augments the generic MC-rollout label.

Paper	Domain	Verification Signal	Core Mechanism	Headline Result
Li et al. [2025]	Code generation	Unit-test execution (public/private pass rates, error traces)	Tree search collects thought steps; each thought labeled with the average pass rate of all code descending from it; PRM scores thoughts *jointly with* the code and execution feedback; Generate–Verify–Refine inference.	Best pass rate/pass@1 on APPS and Codeforces at all difficulties (e.g. 48.17/29 on APPS-Comp); without execution feedback a code PRM is no better than random reward.
Yun et al. [2025]	Clinical reasoning	Retrieved clinical guidelines, textbooks, StatPearls	RAG-as-a-Judge: a large LLM labels each step conditioned on question, gold answer, and retrieved evidence, avoiding MC false negatives; retrieved documents are also part of the PRM's input at inference.	72.6/73.5 avg. over seven medical benchmarks (BoN / SC+RM), SOTA under 10B; lifts Meerkat-8B by +13.70 to 80.35 on MedQA – first 8B model past 80%; expert correlation 0.71 on hard items vs. 0.31–0.34 for MC labels.
Zhao et al. [2026]	Scientific reasoning (bio/chem/-physics/earth)	Sandboxed tool execution + literature/-DOI verification	SCIPRM70K "chain-of-tool" trajectories with real execution in the loop; two-stage labeling (execution-based technical check, then MCTS consistency with an LLM critic over tool *selection*, *calling*, and *result utilization*); SFT + DAPO on Qwen3-VL-8B.	Overall step-level F1 0.769 vs. 0.759 for GPT-5-Mini and 0.682 for the backbone; best tool-calling F1 (0.562; 0.846 on physics); citation-hallucination detection 0.882 HDR; +14.4 pts over an ORM as an RL reward on Mol-Instructions; 0.76 s vs. 282.7 s for live execution.
Zhou et al. [2025]	Operations research	External optimization solver (status + objective value)	Self-evolving co-evolutionary loop: policy and a *generative* PRM improve one another using dual feedback (solver outcome + holistic trajectory critique); preference pairs weighted by the correct-step-ratio gap and optimized with Weighted DPO.	8B model reaches 79.0 avg. Pass@1 over six OR benchmarks, beating DeepSeek-V3 (671B, 75.4) and GPT-4o agentic pipelines; 85.6 with the co-evolved GenPRM as verifier; the same GenPRM lifts a different policy (ORLM) from 65.0 to 75.0.
Pronesti et al. [2026]	Structured risk assessment (medical evidence synthesis)	Deterministic guideline-derived decision rules (Cochrane RoB 2 macros)	Verifiable Process Reward Models: each step's identifier and label are checked by a rule-based verifier; reward $R(Y;x) = \sum_t r_t + r_{\text{label}}$ is fully computable without a neural judge; proves GRPO/DAPO assign positive expected advantage to rule-consistent trajectories.	87.9 acc. / 76.7 macro-F1 on CochraneForest vs. 81.5/70.2 for outcome-only GRPO and 78.2/56.1 for the best neural-PRM baseline; coherence 89.5 vs. $\leq$50.7 for prompted LLMs; OOD gains of +6.2/+3.3 on CRiskEval and a financial-risk set.

scores (max/summed logits) outperform the minimum-score aggregation traditionally used for noise-free human data. OmegaPRM [Luo et al., 2024] accelerates labeling with a divide-and-conquer binary search inside MCTS that isolates the first logical error in logarithmic time, achieving a 75-fold efficiency gain and over 1.5M annotations. Beyond mathematics, Ma et al. [2023] deploy heuristic greedy search with step-level reward models during decoding, with an analogous pipeline for code generation using Abstract Syntax Trees and unit tests.

Zhang et al. [2025b]⋆**.** This paper directly interrogates the two practices above – MC estimation for annotation and BoN for evaluation – and argues both are flawed. It formally distinguishes a PRM (which verifies the correctness of the *current* step) from a value model (which predicts the future probability of reaching the correct answer); MC estimation, which rolls out completions and checks final-answer accuracy, actually trains the latter, producing noisy labels whenever a flawed step happens to reach the right answer (or vice versa). Comparing MC estimation, LLM-as-a-judge, and human annotation (PRM800K), the authors find human

annotation achieves the best step-level accuracy with the least data, while MC estimation gives the worst step localization despite using the most data. They propose *consensus filtering*: retain a training instance only if MC estimation and an LLM judge agree on the first-error location, discarding ∼60% of MC-labeled data but matching LLM-judge quality at a fraction of the compute. They further show hard labels beat soft labels after filtering, and identify three biases in BoN evaluation: (i) *policy–PRM misalignment*, since policies increasingly reach correct answers via flawed reasoning as task difficulty grows; (ii) *score inflation*, where PRMs that cannot detect flawed-but-correct-answer processes still score well under BoN; and (iii) *process-to-outcome shift*, where many PRMs' minimum-step score falls disproportionately on the *last* step, meaning they behave like ORMs in practice. Their resulting Qwen2.5-Math-PRM-7B/72B models reach ProcessBench F1 of 73.5/78.3 (beating GPT-4o and all other open PRMs) and BoN@8 accuracy of 67.6%/69.3%.

Limitations & Future Directions

Automated pipelines close most of the scale gap with human annotation, but the underlying MC-rollout signal remains a noisy proxy for step validity rather than a direct measurement of it – a distinction Zhang et al. [2025b] makes precise and quantifies via consensus filtering, and one that Yun et al. [2025] independently confirm in medicine by showing MC-derived labels correlate with clinical experts at only 0.31–0.34 on hard questions. A second, largely orthogonal risk is evaluation-side: BoN, the default way to test a PRM, can reward PRMs that are secretly acting like ORMs (process-to-outcome shift) or that look strong purely because policy–PRM misalignment grows with task difficulty. Future work should (i) pair BoN with step-level, first-error-localization benchmarks by default rather than reporting BoN alone; (ii) develop rollout-free or cheaper proxies for step correctness that avoid the $O(\text{rollouts} \times \text{steps})$ compute of MCTS-style pipelines; and (iii) standardize evaluation protocols that explicitly control for policy–PRM misalignment across difficulty levels.

4.3 Generalization Beyond Training Distributions: Policy-Agnostic and Reflective Long-CoT PRMs

The two papers in this category share a common diagnosis: standard PRM training and annotation make implicit assumptions – that supervision transfers across policies, and that everything after a solution's first error is also wrong – that break down once a PRM is deployed against unfamiliar policies or long, self-correcting CoT traces.

AURORA [Tan et al., 2025]⋆. Automated annotation pipelines (MC rollouts, single-prompt LLM judges) are typically generated or tuned under one specific sampling policy, so the resulting PRM effectively learns a *policy-conditioned* reward function rather than a universal one, hurting generalization to out-of-distribution policies and long-CoT outputs. AURORA addresses this with a three-stage framework: (1) *Universal Policy Output Generation* samples from a diverse set of policies (7B–72B, multiple LLM families) crossed with diverse prompt templates, using only final-answer correctness (not full human-verified solutions) to yield broad coverage of the output-distribution space cheaply; (2) *Ensemble Prompting* first performs semantic step separation (an LLM splits a response into JSON-structured logical steps, robust to formatting variation), then scores each step under H different prompting strategies, each sampled and majority-voted, and averaged across strategies – interpretable as Bayesian model averaging over diverse inductive biases; (3) *Universal PRM Learning with Reverse Verification* additionally feeds the (probabilistically masked) ground-truth answer to the PRM, letting it cross-check steps against the known outcome, and trains with L2 loss against the dense soft ensemble rewards, supervising *every* step rather than only those up to the first error. AURORA also introduces **UniversalBench**, built because ProcessBench under-represents long-CoT policies and only scores the first error; it sources GSM8K/MATH/OlympiadBench plus harder IMO/AIME/AMC problems, with candidate solutions from 7 diverse policies including long-CoT models (e.g., QwQ-32B), all human-annotated at the step level. Universal-PRM-7B reaches 74.3 F1 on ProcessBench and 74.5 F1 on UniversalBench, with the largest margin over baselines precisely on the long-CoT/OOD-policy subset.

Beyond the First Error [Yang et al., 2025]⋆. Conventional PRM data construction uses all steps of a

correct solution but only the steps up to the first error of an incorrect one, assuming everything afterward stays wrong – an assumption that breaks for long-CoT (o1-style) models, which routinely self-correct or find an alternative path to the right answer. The authors build an Error-Free (EF) set and a Reflection-Based (RB) set (containing an error *and* a self-correction) and show that existing PRMs (Qwen2.5-Math-PRM-7B, MathShepherd-PRM-7B, Skywork-PRM-7B) all suffer large accuracy drops (10 to over 50 points) from EF to RB, confirming that reflective reasoning breaks standard PRMs. Their method fine-tunes a dedicated segmenter (since naive line-break splitting breaks semantic coherence in long CoT) and introduces two annotation rules for a reflective LLM judge (o1): *Error Propagation* (a step that builds on a prior error without correcting it is still wrong) and *Error Cessation* (a step that introduces a new, error-free approach or explicitly corrects the mistake is correct). These rules, injected into the judge's prompt, achieve 96.3% human-verified annotation accuracy, well ahead of GPT-4o (66.8%) and Claude-3.5-Sonnet (72.6%). The resulting 7B PRM, trained via binary classification on 1.7M LLM-judge-annotated samples, reaches PRM@64 = 0.816 and step-level F1 = 0.828 on MATH500 (both best among baselines), and its advantage over baselines grows with problem difficulty – from +0.4% on GSM8K to +8.0% on AIME24 – and is more consistent across generator models than MC-based annotation, whose cross-model label agreement is only 79% and further degrades as solutions lengthen.

Limitations & Future Directions

Both papers show that PRMs trained under a single, narrow generating distribution (one policy family, first-error truncation) silently overfit to that distribution and degrade sharply once deployed against unfamiliar policies or genuinely reflective reasoning – exactly the setting increasingly produced by long-CoT models. Their fixes (diverse-policy sampling with reverse verification; reflective annotation rules with a capable judge) are effective but still expensive: AURORA needs many policies $\times$ prompts $\times$ judge calls, and Beyond-the-First-Error's 96.3% annotation accuracy is bounded by the reflective judge's (o1's) own reasoning ability. Neither fully closes the gap on the hardest OOD problems (e.g., AIME). Promising future directions include: combining AURORA's diverse-policy sampling with Beyond-the-First-Error's reflective segmentation and annotation rules into a single pipeline; extending both ideas to code, agentic tool-use, and the domains of §4.7, where reflection and diverse policies are equally common and where a deterministic verifier could replace the expensive judge entirely; and building diagnostic EF/RB-style stress tests as a standard companion to any new PRM benchmark, rather than an afterthought.

4.4 Implicit Process Rewards and Reinforcement Learning

A major theoretical result shows that explicit PRMs are not strictly necessary for step-level supervision: parameterizing an outcome reward as the log-likelihood ratio between a policy and a reference model, standard algorithms such as DPO or cross-entropy naturally learn the exact Q-value for any intermediate step, yielding an implicit PRM "for free" [Yuan et al., 2024]. Building on this, PRIME [Cui et al., 2025] deploys the mechanism for scalable *online* RL, continuously updating the implicit PRM using only policy rollouts and outcome-level labels, bypassing a costly offline reward-modeling phase and mitigating the reward hacking typical of static PRMs; by extracting token-level dense rewards and combining them with sparse outcome rewards, PRIME integrates with PPO, GRPO, and RLOO. Parallel work shows that standard GRPO inherently functions as an implicit PRM because multiple sampled trajectories naturally share overlapping prefixes, though a step-frequency normalization term is needed to prevent highly frequent paths from distorting the signal [Sullivan and Koller, 2026]. Step-GRPO [Li et al., 2026] further aggregates dense step-level PRM signals into sparse trajectory-level feedback via a step-attention mechanism with a soft positional bias, capturing inter-step dependencies while avoiding overfitting to isolated local signals. Finally, Setlur et al. [2024] show that optimizing for absolute Q-values in online RL often fails to encourage exploration, and instead set step-level rewards as the *advantage* under a complementary prover policy, improving both exploration efficiency and sample efficiency.

Process-based Self-Rewarding Language Models [Zhang et al., 2025c]⋆. Self-Rewarding LMs, where a single model acts as both policy and reward model, hurt math performance in their original whole-response formulation, for two reasons: whole-response rewarding is too coarse a signal for long reasoning chains, and absolute scoring of a whole solution is unreliable compared to pairwise comparison. This paper brings self-rewarding down to the *step* level. A human-agreement study confirms LLMs are far more consistent at pairwise step comparison (0.84/0.88 agreement) than absolute scoring (0.72/0.32). The pipeline initializes the model with instruction-following and evaluation fine-tuning data, then iterates: the model generates step-by-step solutions, judges its own steps pairwise, and updates via DPO on the resulting self-generated preference data, without external labels at each round. Across iterations ($M_1 \rightarrow M_4$), models take fewer but longer, higher-quality reasoning steps (e.g., 72B on MATH: 8.41 steps at 61.0 words $\rightarrow$ 5.54 steps at 96.6 words); step-wise judge accuracy stays 92–96% throughout; and test-time scaling (sampling 6 candidates per step, selecting the best via the model's own judge) beats greedy decoding, with the margin widening across iterations (55.9$\rightarrow$58.2 at M_1; 60.6$\rightarrow$62.4 at M_4).

Limitations & Future Directions

Implicit and self-rewarding methods make dense process supervision essentially free at training time, but they inherit a subtler dependency: implicit-PRM extraction relies on the calibration of the reference/policy pair [Yuan et al., 2024, Cui et al., 2025], GRPO-derived signals must be explicitly de-biased for path frequency [Sullivan and Koller, 2026], and step-level self-rewarding is bottlenecked by the quality of its cold-start model M_1 and has so far only been iterated a handful of times due to compute constraints [Zhang et al., 2025c]. No unifying theory yet explains when DPO-, GRPO-, and self-rewarding-derived implicit rewards agree or diverge, although Pronesti et al. [2026] provide a partial answer in the opposite direction by proving that an *externally verified* process reward yields a clean sign separation in the expected GRPO/DAPO advantage. Future work should pursue (i) a shared theoretical account of implicit reward extraction across these formulations; (ii) cold-start-free or cold-start-robust initialization for iterative self-rewarding; and (iii) compute-efficient variants that can run substantially more than 3–4 self-rewarding rounds to test whether the observed gains saturate or continue to compound.

4.5 Generative Verification and Test-Time Scaling

Traditional discriminative PRMs output static scalar values, which limits interpretability and prevents scaling verification compute at inference time. GenPRM [Zhao et al., 2025] reformulates verification as a generative task, producing detailed chain-of-thought rationales and executing Python code before issuing a judgment, trained with Relative Progress Estimation to filter unhelpful reasoning steps. GenRM [Zhang et al., 2025a] recasts reward modeling entirely as next-token prediction, generating an intermediate rationale before predicting correctness, unifying generation and verification and enabling test-time scaling via majority voting over sampled verification paths – and remains sample-efficient even trained on purely synthetic, model-generated rationales. ThinkPRM [Khalifa et al., 2025] leverages long-CoT reasoning models to fine-tune a generative verifier with only 8,000 process labels, outperforming discriminative classifiers trained on two orders of magnitude more data, and supports dynamic test-time scaling via parallel or sequential verification chains.

R-PRM [She et al., 2025]⋆. Existing PRMs output a direct scalar score per step, which is data-hungry, uninterpretable, and cannot explain *why* a step is wrong. R-PRM makes the PRM reason about each step before scoring it: for step s_i, it first generates an analysis A_i covering four angles (history of prior steps, the objective/data source of the current step, coherence with preceding steps, and validity of computational transformations), then a judgment J_i conditioned on that analysis. Seed data is produced by prompting a stronger LLM with PRM800K examples to yield (analysis, judgment) pairs, keeping only those whose final judgment matches the human label, and fine-tuning via standard next-token prediction over the concatenated sequence. A DPO stage then samples multiple evaluation trajectories per step, splitting them into chosen (judgment matches the label) and rejected (does not) without any new labels, directly rewarding reasoning

processes that lead to correct judgments. At inference, K independent analysis-judgment trajectories are sampled and their "yes" probabilities averaged (default K=10, up to K=32) to reduce variance. R-PRM-7B-SFT already beats Qwen2.5-Math-7B-PRM800K on ProcessBench F1 (65.2 vs. 58.5); DPO adds a further +5.2 to reach 70.4 without new labels; on PRMBench, R-PRM-7B-DPO scores 66.8 overall, best among compared models and even ahead of GPT-4o on sensitivity; and with only 64K training samples it already beats a PRM trained on 265K, illustrating strong data efficiency. A case study shows R-PRM correctly flags a solution that skips verifying one candidate (scoring it 0.05), where score-only PRMs confidently ($\geq$0.86) call it correct.

Limitations & Future Directions

Generative verifiers trade inference-time compute for accuracy and interpretability, and R-PRM's think-then-judge formulation extends this trend with an explicit, structured analysis stage and label-free DPO refinement. The shared cost, however, is latency: GenPRM's code execution and R-PRM's K-way trajectory sampling both multiply inference cost relative to a single discriminative forward pass, and R-PRM in particular remains untested at 70B scale, with advanced search strategies (MCTS, beam search) as a reward source left unexplored. Zhou et al. [2025] extend the generative paradigm in a different direction, arguing that generative critics are valuable not only for interpretability but because a *holistic, post-hoc* critique of a completed trajectory resolves interdependencies that per-step scoring cannot see. Future work should focus on (i) distilling generative verifiers' reasoning-then-judging behavior into smaller, faster discriminative heads without losing interpretability; (ii) adaptive selection of K (or of generation vs. discrimination) conditioned on task difficulty, to avoid paying maximal inference cost on easy problems; and (iii) integrating generative verifiers directly as the reward source inside tree-search decoding, rather than only as a post-hoc re-ranker.

4.6 Advanced Reward Architectures and Cross-Domain / Multimodal Extensions

Beyond generative approaches, several papers redesign the underlying mathematical or structural architecture of discriminative PRMs. PQM [Li and Li, 2024] recasts process reward modeling as a Q-value ranking problem governed by MDP dynamics, using a comparative loss with a tunable margin to ensure optimal Q-values rise on correct steps and fall sharply on incorrect ones, capturing step interdependencies that independent per-step classification ignores. BiRM [Chen et al., 2025] observes that standard PRMs are one-directional, evaluating only the backward correctness of a step without estimating the forward probability of eventually reaching the answer; inspired by A* search, BiRM adds a value-model head that estimates a partial solution's future success potential, mitigating the "scaling decline" of Best-of-N sampling and improving step-level beam search.

VisualPRM [Wang et al., 2025]⋆. Test-time scaling via BoN is well studied for text LLMs but under-explored for multimodal LLMs (MLLMs), for two reasons: existing open-source MLLMs are poor critics (BoN with them barely beats no test-time scaling), and no benchmark cheaply evaluates multimodal step-wise critics. VisualPRM contributes three pieces: **VisualPRM400K**, built from MMPR v1.1 images/questions with step-by-step solutions sampled from InternVL2.5 models and each step labeled via MC rollout (mc_i = fraction of sampled continuations from that step reaching the correct answer; $\sim$400K samples, $\sim$2M step labels, $\sim$10% of steps incorrect); **VisualPRM-8B**, trained as a multi-turn chat model (image+question+first step in turn one, one new step per subsequent turn), comparing *value-based* scoring ($mc_i > 0$) against *advantage-based* scoring ($mc_i - mc_{i-1} > 0$) and finding value-based wins because the automatic labeling noise makes relative-improvement labels less reliable; and **VisualProcessBench**, 2,866 samples / 26,950 step labels across five multimodal reasoning sources, with solutions from five leading MLLMs (GPT-4o, Claude-3.5-Sonnet, Gemini-2.0-Flash, QvQ-72B, InternVL2.5-78B) and human experts labeling *every* step as positive/negative/neutral. As a BoN critic, VisualPRM-8B lifts every tested policy MLLM by 3.7–8.4 points, including a 78B model, while the strong InternVL2.5-8B itself is a weak critic; on VisualProcessBench it scores 62.0 F1, beating GPT-4o (60.3) and matching Gemini-2.0-Flash (62.3), while open-source MLLM judges cluster near a random-guess baseline of 50.0. Ablations show that supervising all steps (not stopping at the first error) and averaging step

scores (rather than min/max) both help, and VisualPRM keeps improving up to Best-of-128 where an ORM plateaus past N=64. Notably, VisualPRM's gains transfer to *pure-text* reasoning despite being trained only on multimodal data (e.g., +6.6 points on GPQA-Diamond for Qwen2.5-72B under BoN).

OpenPRM [Zhang et al., 2025d]⋆. PRMs have driven large gains in math and code, but open-domain tasks (chat, instruction-following) lack exact answers, making step-level supervision hard to obtain; meanwhile ORMs are abundant and strong but suffer cumulative error, since they only see the final result and cannot correct early mistakes that do not change the outcome. OpenPRM asks whether a PRM can be derived from existing ORMs with no new human labels. A diagnostic study shows that rewarding only the first-n processes of a response with existing open-domain ORMs improves accuracy monotonically on easy Chat tasks but rises-then-falls on harder Chat-Hard/Reasoning tasks – direct evidence of cumulative error and length bias. The proposed fix identifies the most divergent process steps between a chosen and a rejected response and adds pairwise-ranking supervision there, combined with the standard ORM loss ($\mathcal{L} = \mathcal{L}_{\text{ORM}} + \lambda\mathcal{L}_{\text{PRM}}$). Concretely, OpenPRM builds *Process-Level Preference Trees*: (1) repeatedly sample many candidate responses per prompt from open LLMs; (2) segment each response into sentences and merge similar sentences across candidates (via an edit-distance threshold) into a shared prefix tree, which is cheaper than generating new rollouts as in math/code PRM pipelines; (3) backpropagate outcome rewards from an ensemble of existing ORMs at the leaves up to internal nodes, whose value is the mean reward of all descendant leaves. Training combines process pairs (from tree divergence) and outcome pairs under one unified ranking objective, continually fine-tuning base ORMs (FsfairX, InternRM) on this data. On RewardBench, OpenPRM(InternRM) reaches 91.1 average (vs. 87.6 base), with the largest gains on Chat-Hard (+12.1) and Reasoning (+7.1) – exactly the categories shown to degrade under plain ORMs; BoN and Process-level Beam Search with OpenPRM comfortably beat majority voting on IFEval, GPQA, MMLU-Pro, and MATH500; and OpenPRM shows much lower correlation with response length (0.05) than the base ORM (0.37), indicating reduced length bias. Even so, oracle coverage (pass@N) climbs toward 100% as N scales from 1 to 128, while *no* reward model, including OpenPRM, comes close to this ceiling – BoN accuracy with any RM plateaus or degrades past $N \approx 16$–32.

Limitations & Future Directions

Architectural innovations solve specific structural blind spots (Q-value interdependence, one-directional scoring, modality mismatch, missing open-domain step labels), but each solution remains fairly domain- or modality-specific: PQM and BiRM target math reasoning, VisualPRM's value-based labeling still inherits MC-rollout noise (the same concern raised in §4.2), and OpenPRM's bootstrapped signal, while effective, still leaves a large, unresolved gap to the oracle/coverage ceiling under scaled inference-time sampling – a gap that persists even for the strongest reward model tested. Future work could (i) unify Q-value ranking, bidirectional scoring, and preference-tree formulations under one general MDP-theoretic framework applicable across math, code, chat, and multimodal domains; (ii) extend multimodal PRMs beyond static images to video, audio, and embodied/agentic settings; and (iii) directly target the reward-model-to-oracle coverage gap via exploration-aware reward shaping or hybrid search strategies, rather than only improving pointwise scoring accuracy.

4.7 Domain-Specialized and Verifiable Process Supervision

The five papers in this category start from a shared observation: outside of mathematics, "did a rollout from this prefix reach the right answer" is often both a poor proxy for step validity *and* unnecessary, because the domain already supplies an externally checkable artifact against which an intermediate claim can be tested. Each paper identifies a different such artifact – a test suite, a clinical guideline, a scientific tool, an optimization solver, a written decision procedure – and rebuilds the annotation pipeline around it (Figure 3, right). A secondary theme is that the definition of a "step" itself must be renegotiated per domain: a thought that produces code is not a line of code [Li et al., 2025], a scientific step bundles tool selection, invocation, and interpretation [Zhao et al., 2026], and an operations-research step is only interpretable in the context of the steps that follow it [Zhou

et al., 2025].

CodePRM [Li et al., 2025]⋆. Building a PRM for code generation is harder than for mathematics for two reasons: the granularity of a "step" has no consistent definition (a line, a token, or a natural-language thought), and unlike mathematics – where intermediate steps contain partial computation results that make errors traceable – code thoughts are abstract statements about algorithmic design and data structures, so a failing program does not localize the flawed thought. The paper's key negative result is that training a code PRM with the standard mathematical recipe (thoughts only, binary labels) yields BoN performance *no better than random reward assignment*, establishing that a new paradigm is required rather than a transplant. CodePRM's proposal is that thoughts should never be scored in isolation: the PRM's input is $(x \oplus T \oplus c \oplus f)$, the problem, the thought chain, the code derived from it, and the execution feedback (public/private pass rates plus error traces). Training data comes from a tree search in which each node stores its state, code, feedback, and pass rates, and each thought is labeled with the *average private pass rate* of all code descending from it, backpropagated up the tree as new programs are executed; the PRM (Qwen2.5-Coder-7B-Instruct with a value head) is trained with MSE against this soft label. At inference, the Generate–Verify–Refine pipeline uses the PRM as a live verifier inside search: candidate thought chains are generated (BoN or MCTS), failing programs trigger step-level scoring, low-scoring thoughts are flagged (either by a threshold or by locating the single minimum), and the generator rewrites only those thoughts. Empirically, CodePRM(GVR-MCTS) is the strongest method on both APPS and Codeforces at all difficulty levels and for both GPT-4o-mini and DeepSeek-Coder-V2-Lite backbones (e.g. 48.17 pass rate / 29 pass@1 on APPS-Competition, versus 42.50/28 for the closest refinement-based baseline). Four ablations are especially informative: execution feedback is necessary rather than merely helpful; soft pass-rate labels beat binary ones because a thought can yield several programs all passing $>$0.9 of tests yet be labeled 0 under binarization; threshold-based verification only outperforms locate-minimum when the verifier is precise enough (with a weaker GPT-4o-mini verifier it over-corrects); and refining *more* thoughts does not monotonically help, since excessive refinement converts correct thoughts into erroneous ones. Finally, CodePRM-7B substantially outperforms general-purpose and mathematical open PRMs (Skywork-PRM-1.5B/7B, Math-Shepherd-Mistral-7B) on APPS, confirming that code process supervision does not come for free from math-trained models.

Med-PRM [Yun et al., 2025]⋆. Clinical decision making is a natural fit for process supervision – a single wrong inference invalidates a diagnosis – but MC-style auto-labeling is actively harmful there: it produces *false negatives*, penalizing factually correct and logically coherent early steps merely because no sampled continuation happened to reach the gold answer. Med-PRM replaces this with RAG-AS-A-JUDGE: for each question, evidence is retrieved from four biomedical corpora (clinical guidelines, StatPearls, medical textbooks, a rare-disease corpus) using a MedCPT bi-encoder with cross-encoder reranking (400 candidates $\rightarrow$ top 32), and a large LLM assigns each step a binary label conditioned on the question, the gold answer, and that evidence. Crucially, retrieval is used at *both* stages: the trained PRM (a fully fine-tuned Llama-3.1-8B scoring the $+/-$ token at each step marker) also receives the retrieved documents as input, so $\mathrm{RM}(D, q, S)$ rather than $\mathrm{RM}(q, S)$. Across seven medical benchmarks Med-PRM averages 72.59 (Best-of-N) and 73.50 (self-consistency + reward model), outperforming all open-source language, reasoning, medical, and medical-PRM baselines under 10B parameters and beating the MCTS-labeled MedS^3 by 2.44–3.08 points. Used as a plug-and-play verifier it improves every policy tested, most strikingly lifting Meerkat-8B by 13.70 points to 80.35 on MedQA – reported as the first time an 8B-scale model crosses the 80% threshold on that benchmark. Two analyses are worth highlighting. First, expert alignment: on a 180-annotation study, Med-PRM's step labels correlate with physician judgments at 0.74 (easy) and 0.71 (hard), whereas soft and hard MC labels collapse from 0.64/0.70 to 0.34/0.31 on the hard subset – the gap opens exactly where supervision matters most, echoing Yang et al. [2025]'s finding that MC labels degrade with difficulty. Second, cost: the entire labeling budget was under $20, against roughly $20,000 for a comparable human-curated medical dataset. An ablation also reproduces the Song et al. [2025] concern from the opposite direction: conventional PRMs beat self-consistency under SC+RM but *underperform* it under the stricter Best-of-N, while Med-PRM wins under

both, suggesting Best-of-N is the more discriminative protocol for comparing reward models.

Sci-PRM [Zhao et al., 2026]⋆. Scientific reasoning in biology, chemistry, physics, and earth science differs from mathematics in that correctness depends on evidence grounding against rapidly evolving factual databases rather than symbolic derivation, and in that many claims are *tool-verifiable* via molecular property libraries, sequence-alignment services, equation solvers, or literature retrieval. The paper's motivating measurement is a *tool gap*: frontier judges such as GPT-5-Mini and Gemini-3-Flash retain high step-level F1 on ordinary reasoning steps (0.83, 0.79) but fall to 0.53–0.56 on steps that involve a tool call, a gap of 0.23–0.30 that no general-purpose judge closes. Sci-PRM addresses this with SCIPRM70K, a corpus of 17,818 "chain-of-tool" trajectories ($\sim$72K steps, $\sim$25K tool invocations) generated interactively: when a model emits a tool call, generation pauses, the call is really executed (sandboxed Python with RDKit, SymPy, Biopython, or a search engine returning top-5 results), and the observation is appended so downstream reasoning is grounded in actual rather than hallucinated feedback. Labeling proceeds in two stages: technical verification first (unparsable queries, empty retrievals, syntax errors, and 300-second timeouts are labeled negative), then MCTS consistency, where a step is positive only if K rollouts reach the gold answer above a confidence threshold γ while an LLM critic simultaneously assesses three distinct failure modes – tool *selection*, argument *accuracy*, and *utilization* of the returned evidence. The resulting model (Qwen3-VL-8B, SFT then DAPO) attains the best overall step-level F1 on SCIPRM-Bench (0.769 vs. 0.759 for GPT-5-Mini and 0.682 for its own backbone), the best tool-calling F1 (0.562 globally, 0.846 on physics), and the best citation-hallucination detection (accuracy 0.746, F1 0.627, hallucination-detection rate 0.882), where baselines fail in two opposite ways – blind trust (LLaMA-3.2-11B detects 5% of fabricated citations) or over-rejection (GPT-5-Mini reaches 0.87 detection but only 0.18 recall on genuine papers). Downstream, Sci-PRM beats majority voting and a general-purpose PRM under Best-of-N on all four held-out benchmarks and, as an RL reward, exceeds an outcome reward model by 14.4 points on Mol-Instructions. The efficiency analysis supplies the practical argument for the whole approach: live execution raises accuracy from 0.51 to 0.86 but costs 282.7 seconds per query because of remote BLAST calls, whereas Sci-PRM recovers 0.76 accuracy at 0.76 seconds – making it viable as an inner-loop reward where genuine execution is not.

StepORLM [Zhou et al., 2025]⋆. Operations research exposes both failure modes at once. Outcome rewards suffer credit assignment: the paper's case study shows a travelling-salesman formulation that omits subtour-elimination constraints yet returns the same optimal objective value, so a solver-verified reward silently reinforces an invalid model. But conventional discriminative process supervision is *myopic*: the correctness of an early step such as a variable definition often cannot be judged until later constraints are written, so per-step scoring in isolation is unreliable. StepORLM therefore argues for holistic, trajectory-level process supervision delivered by a *generative* PRM that reasons over the completed trajectory before assigning credit. The framework has two stages. A warm-up stage synthesizes OR problems and step-by-step solutions with a teacher LLM, validates every one deterministically by executing the solver code and inspecting status and objective value, and retries with an automatic refinement loop, yielding a verified corpus for SFT. The co-evolution stage then alternates: the policy samples k trajectories, each evaluated from two complementary sources – definitive outcome verification from the external solver and a holistic post-hoc critique from the GenPRM – and the dual signal is distilled into preference pairs (a solver-verified trajectory always wins; ties are broken by the ratio of correct steps) that align the policy via Weighted DPO, where solver-distinguished pairs receive weight 1.0 and the rest are weighted by their process-quality gap, while the same evaluation data fine-tunes the GenPRM. The 8B StepORLM reaches 79.0 average Pass@1 across six OR benchmarks, outperforming DeepSeek-V3 (671B, 75.4), Qwen2.5-72B-Instruct, specialized OR models, and GPT-4o-based agentic pipelines; adding the co-evolved GenPRM as an inference-time verifier pushes this to 85.6, with the largest gains on the hardest benchmarks. Two results speak to generality: the GenPRM outperforms majority voting, solver execution, and a discriminative PRM as an inference-scaling strategy, and it transfers to a *different* policy model, lifting the independent ORLM baseline from 65.0 to 75.0 – evidence that it has learned model-agnostic principles of valid OR reasoning rather than idiosyncrasies of its co-evolved partner. Ablations

confirm each component matters, with warm-up SFT the most critical (79.0 → 64.6 without it) and freezing the GenPRM costing 3.7 points, validating the co-evolutionary dynamic itself.

VPRM [Pronesti et al., 2026]⋆. This paper pushes the logic of the category to its endpoint: if reinforcement learning with verifiable rewards (RLVR) avoids reward hacking by grounding the *outcome* signal in a deterministic check, why should the *process* signal remain a learned neural judge, with all the opacity, bias, and hackability that entails? Verifiable Process Reward Models make every step externally checkable. The setting is risk-of-bias assessment for medical evidence synthesis, chosen because Cochrane RoB 2 prescribes, for each bias domain, a fixed sequence of assessment questions with a closed label set and a decision tree that deterministically maps step labels to a low/moderate/high risk verdict. Formally, each step t emits a step identifier s_t and a step label $\hat{\ell}_t$; two bounded scoring functions compare them to the gold identifier and label implied by the rules, giving $r_t(Y;x) = w_t^{\mathrm{n}} s_t^{\mathrm{n}}(s_t, s_t^\star) + w_t^{\mathrm{l}} s_t^{\mathrm{l}}(\hat{\ell}_t, \ell_t^\star)$, and the full reward is $R(Y;x) = \sum_{t=1}^{T} r_t(Y;x) + r_{\text{label}}$ – computable end-to-end without any learned critic. The paper also proves a reward-separation guarantee: under finite variance, $\mu_c > \mu_i$, and large group size, the expected GRPO and DAPO advantage satisfies $\mathbb{E}[\hat{A}(Y) \mid \mathcal{C}] > 0$ and $\mathbb{E}[\hat{A}(Y) \mid \mathcal{C}^c] < 0$, so rule-consistent trajectories receive positive expected weight; a verifiable outcome reward model falls out as the special case with degenerate intermediate label spaces. Training data comes from CochraneForest/CochraneForestExt (2,946 paper–risk pairs from 104 systematic reviews) enriched with silver step labels from Llama-3.1-405B, whose quality a manual audit puts at 100% coherent traces and 96.7% correct labels. Qwen2.5-7B-GRPO-VPRM reaches 87.9 accuracy / 76.7 macro-F1 on CochraneForest, against 81.5/70.2 for outcome-only GRPO, 68.4/45.5 for Llama-3.1-405B zero-shot, and 78.2/56.1 for the best neural-PRM-supervised variant – the comparison against neural PRMs is controlled, since both conditions receive the same verifiable outcome reward, isolating the effect of process supervision alone. Beyond accuracy the paper introduces *coherence* (does the final verdict match what the model's own step labels imply under the decision function?) and *coherent accuracy*; VPRM-trained models reach 89.5 and 75.0 respectively, while prompted large models sit at 36–51 coherence and 25–29 coherent accuracy, i.e. their step-level reasoning is largely decorative. Ablations show the outcome term remains essential (removing it drops accuracy to 40.2) and that verifying step *content*, not merely step *structure*, is what separates full VPRM from a steps-only variant. Gains also transfer out of distribution, to a Chinese AI-safety risk benchmark (+6.2) and a synthetic financial-risk dataset (+3.3), suggesting the model learns a transferable habit of rule-following rather than dataset-specific regularities.

Limitations & Future Directions

This category resolves the annotation-validity problem of §4.2 by construction wherever a verifier exists, and the empirical payoff is consistent across five very different domains (Figure 4). But the resolution is purchased with portability, and each paper is explicit about the bill. Verifiability requires structure: Pronesti et al. [2026] note that tasks without well-defined intermediate steps cannot benefit directly, and that a mismatch between the model's output format and the verifier's expectations silently degrades the reward – a risk that grows for smaller models. Grounding is only as good as its ground: Med-PRM inherits whatever its retrieval corpus omits, Sci-PRM inherits the coverage of its tool suite and the reliability of its LLM critic, and VPRM inherits any incompleteness in the underlying guidelines. Several pipelines also remain dependent on the very machinery they set out to replace – Sci-PRM still runs MCTS consistency checks for its second labeling stage, and CodePRM's pass-rate labels are still a reachability statistic, merely a much better-grounded one. Finally, evaluation here is fragmented: each paper reports on its own benchmark with its own metric, so no cross-domain comparison of process-supervision quality is currently possible. Promising directions include: (i) a general recipe for eliciting or synthesizing verifiers in domains where guidelines are implicit rather than written, potentially by having an LLM draft candidate decision rules that human experts only audit; (ii) hybrid rewards that fall back gracefully from deterministic verification to judge-based scoring on the sub-steps a verifier cannot cover, with calibrated weighting between them; (iii) transferring Zhou et al. [2025]'s co-evolutionary loop and Li et al. [2025]'s verify-then-refine inference

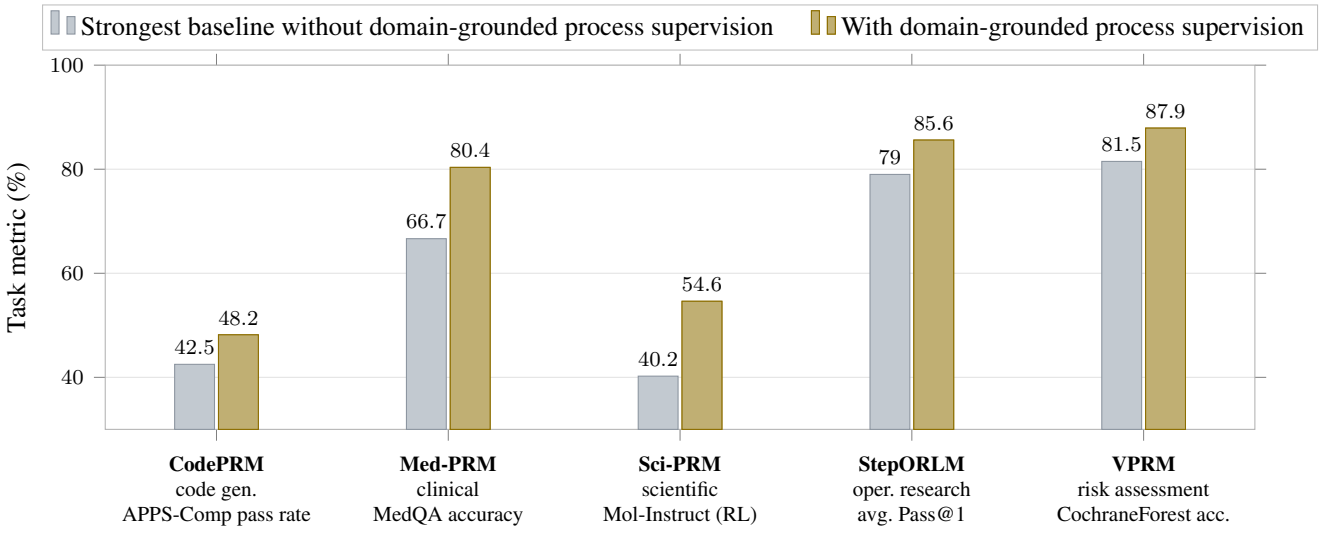

Figure 4: Effect of domain-grounded process supervision in each of the five domains of §4.7. Each pair contrasts the strongest reported system *without* the paper's process signal against the same setup *with* it, holding the policy and evaluation protocol fixed within a pair. Baselines are, left to right: the best refinement-based search method (RethinkMCTS), the unaided Meerkat-8B policy, an outcome reward model used as the RL signal, StepORLM without its GenPRM verifier, and outcome-only GRPO. Metrics differ by domain, so bars are comparable *within* but not *across* pairs; the consistent within-pair gap (+5.7 to +14.4 points) is the point of the figure.

to the other domains, since neither is intrinsically OR- or code-specific; and (iv) a shared cross-domain evaluation suite so that the claim "domain-grounded supervision beats generic MC supervision" can be tested under one protocol rather than five.

4.8 Comprehensive Benchmarking

As automated verification techniques proliferate, rigorous evaluation is required to gauge true capability. ProcessBench [Zheng et al., 2025a] provides 3,400 expert-annotated cases focused on pinpointing the earliest erroneous step in Olympiad-level mathematics, and shows that widely used, synthetically trained discriminative PRMs generalize poorly to challenging mathematics relative to human-annotated models. PRMBench [Song et al., 2025] evaluates over 83,000 step-level labels across nine fine-grained dimensions (simplicity/redundancy, soundness/counterfactual detection, sensitivity to missing prerequisites and deceptive traps), and finds that current PRMs consistently fall short of human accuracy and exhibit a pronounced bias toward assigning positive rewards – i.e., they systematically miss false positives – while a severe negative correlation between PRMBench performance and BoN accuracy demonstrates continued susceptibility to reward hacking.

Several further benchmarks broaden this picture along the axes Tables 1 and 2 highlight as under-tested. **UniversalBench** [Tan et al., 2025] targets cross-policy and long-CoT generalization specifically, sourcing harder competition problems (IMO/AIME/AMC) and requiring solutions from seven diverse policies, with the largest performance gaps among PRMs appearing exactly on this OOD-policy, long-CoT subset. **VisualProcessBench** [Wang et al., 2025] extends step-level, all-error benchmarking to the multimodal setting, showing that open-source MLLM judges cluster near random guessing (50.0 F1) even though the best PRMs and proprietary critics reach 60–65. **SCIPRM-Bench** [Zhao et al., 2026]* adds a dimension none of the mathematical benchmarks captures: it partitions every evaluation into tool-calling and non-tool-calling steps, which converts a single aggregate F1 into a diagnostic *tool gap* and reveals that even frontier judges lose 0.23–0.30 F1 the moment a step invokes an external instrument. Pronesti et al. [2026]* likewise repurpose CochraneForest and RoBBR as process-level testbeds and introduce *coherence* and *coherent accuracy*, which measure whether a

model's final verdict actually follows from its own intermediate labels under an externally specified decision function – a property invisible to accuracy alone, and one on which strong prompted LLMs score below 51%. Complementing these, the EF/RB diagnostic pairs introduced by Yang et al. [2025] function as a targeted stress test for reflective reasoning specifically, rather than a general-purpose leaderboard, and reveal 10–50 point accuracy drops in existing PRMs that standard benchmarks miss entirely. Table 3 summarizes all benchmarks discussed in this survey, and Figure 5 compares reported ProcessBench average F1 across a representative subset of models.

Table 3: Representative benchmarks for process-level evaluation discussed in this survey.

Benchmark	Domain	Scale	Distinguishing Focus
ProcessBench [Zheng et al., 2025a]	Math (Olympiad)	3,400 cases	Earliest-error localization on competition-level problems.
PRMBench [Song et al., 2025]	Math	6K+ problems, 80K+ step labels	Nine-dimensional error taxonomy (simplicity, soundness, sensitivity); exposes positive-label bias and reward hacking.
UniversalBench* [Tan et al., 2025]	Math (IMO/-AIME/AMC)	7 diverse policies (long-CoT)	Cross-policy and OOD generalization; all-step human labels.
VisualProcess-Bench* [Wang et al., 2025]	Multimodal	2.9K samples, 27K step labels	All-step, multi-source (5 MLLMs) human labeling for multimodal critics.
EF/RB diagnostic sets* [Yang et al., 2025]	Math (long-CoT)	Paired error-free / reflection-based sets	Targeted stress test for self-correcting, reflective reasoning.
RewardBench (used by Zhang et al. [2025d])	Open-domain (chat, safety, reasoning)	Standard RM leaderboard	Evaluates label-free, ORM-bootstrapped PRMs beyond math/code.
APPS / Codeforces (used by Li et al. [2025])*	Code generation	3 difficulty tiers each	Pass rate and pass@1 under a fixed search budget; isolates the value of execution feedback in the PRM's input.
SCIPRM-Bench* [Zhao et al., 2026]	Scientific (bio, chem, physics, earth)	1.2–1.5K held-out trajectories	Splits every metric by tool-calling vs. non-tool-calling steps, exposing a *tool gap*; separate citation-authenticity track.
CochraneForest / RoBBR* [Pronesti et al., 2026]	Structured medical risk assessment	1.8K / 3.4K test instances	Adds *coherence* and *coherent accuracy*: does the final verdict follow from the model's own step labels under a fixed decision rule?
OR suite (NL4Opt, MAMO, NLP4LP, ...)* [Zhou et al., 2025]	Operations research	Six cleaned benchmarks	Solver-checked Pass@1; used to show a generative PRM transfers as a verifier across policy models.

Limitations & Future Directions

Existing benchmarks remain largely domain-siloed – ProcessBench and PRMBench evaluate math, VisualProcessBench multimodal reasoning, RewardBench open-domain preferences, and each paper in §4.7 introduces a bespoke testbed for its own domain – so no single benchmark yet lets us test the "universal PRM" claims of §4.3 across domains and modalities simultaneously. Figure 5 makes the cost concrete: the five domain-specialized systems cannot appear on it at all, because none is evaluated on a shared axis with the mathematical PRMs. Most benchmarks also still focus on the *first* error rather than the full trajectory, under-testing exactly the reflective-reasoning failure mode that Yang et al. [2025] shows is common in long-CoT outputs. Future benchmarking work should (i) construct a single cross-domain, cross-modality, policy-diverse suite combining the methodologies of UniversalBench, VisualProcessBench, PRMBench,

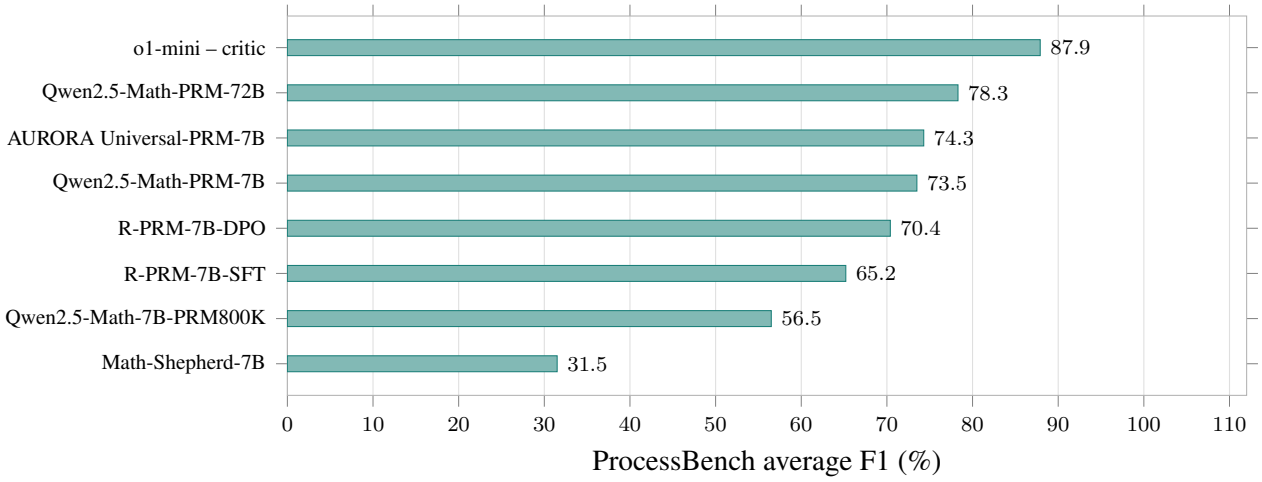

Figure 5: ProcessBench average F1 for a representative subset of PRMs discussed in this survey (values as reported in the respective papers, [Zhang et al., 2025b, She et al., 2025, Tan et al., 2025, Zheng et al., 2025a]). o1-mini is a general-purpose generative critic rather than a dedicated PRM, shown for reference. The domain-specialized models of §4.7 are necessarily absent: none reports ProcessBench, which is itself the fragmentation problem noted below.

and SCIPRM-Bench; (ii) make all-step (not first-error-only) labeling, reflective-reasoning stress tests, and tool-calling splits the default rather than specialized add-ons; (iii) adopt Pronesti et al. [2026]'s coherence metric more widely, since it detects the decorative-reasoning failure mode without requiring per-step gold labels wherever a decision function exists; and (iv) report downstream utility (BoN/RL gains) alongside static classification accuracy, since Song et al. [2025] show these can be negatively correlated and Yun et al. [2025] show the choice of test-time protocol can reverse a ranking.

5 Comparison and Discussion

The transition from human-annotated labels to automated MC pipelines significantly lowers data collection cost but introduces varying degrees of systemic noise. Zhang et al. [2025b] formalize this noise as a value-model/PRM mismatch and mitigate it via consensus filtering; Wang et al. [2024] instead mitigate it via score-favoring aggregation functions; Yang et al. [2025] show the noise compounds specifically as solutions lengthen; and Yun et al. [2025] quantify it from the clinical side, where MC-derived labels correlate with expert judgment at only 0.31–0.34 on hard questions against 0.71 for evidence-grounded labels. Across all four, the common thread is that *raw MC rollouts are not a free lunch* – some combination of filtering, aggregation, judge-based re-annotation, or outright replacement is needed to make automated labels trustworthy.

A second axis of comparison is *discriminative vs. generative* scoring. Discriminative PRMs (PQM, BiRM, VisualPRM, OpenPRM, CodePRM, Med-PRM) are cheap at inference time but have historically been one-directional and unable to explain their scores; recent architectural work narrows this gap via Q-value ranking [Li and Li, 2024] and bidirectional heads [Chen et al., 2025]. Generative verifiers (GenPRM, GenRM, ThinkPRM, R-PRM, Sci-PRM, StepORLM's GenPRM) bypass these limitations by natively supporting test-time compute scaling and producing an interpretable rationale, at the cost of higher inference latency. Zhou et al. [2025] contribute a distinct argument for the generative form that is not about interpretability at all: because OR steps are interdependent, a critic must read the *completed* trajectory before it can judge an early step, so holistic generative critique is a correctness requirement rather than a presentational preference. This connects directly to the structural motivation behind PQM and BiRM (§4.6), which attack the same interdependence problem from within a discriminative architecture.

A third axis is *single-policy/single-domain vs. universal*. Most PRMs before AURORA and VisualPRM

were trained and evaluated within one policy family and one modality; both show that broadening the training distribution yields the largest gains precisely on the hardest, most out-of-distribution subsets, and VisualPRM's multimodal-to-text transfer suggests these gains may be more general than modality-specific training would imply. OpenPRM extends the same intuition to the open domain by showing that even a *label-free*, ORM-bootstrapped PRM captures failure modes that plain ORMs exhibit on harder tasks. Interestingly, the domain-specialized papers supply evidence on both sides of this axis: CodePRM shows that math-trained open PRMs transfer poorly to code, and Sci-PRM shows the same for general-purpose PRMs on scientific tool use, yet StepORLM's GenPRM transfers cleanly to a different OR policy and VPRM's rule-following behavior transfers to unrelated risk domains. The emerging picture is that PRMs generalize across *policies* more readily than across *verification regimes*.

A fourth axis is *proxy estimation vs. grounded verification* (Figure 3). Sections 4.2–4.6 largely try to make a generic proxy less noisy; Section 4.7 replaces the proxy where the domain permits. The five papers reach this conclusion independently and from different starting points – CodePRM from the observation that a math-style code PRM is no better than random, Med-PRM from MC false negatives on clinically sound steps, Sci-PRM from the tool gap in frontier judges, StepORLM from a solver-verified optimum reached by an invalid model, and VPRM from the reward-hacking surface that any learned judge exposes – which is itself evidence that the underlying limitation is structural rather than incidental. Two further patterns are worth noting. First, none of them discards outcome supervision: CodePRM's labels are pass rates, StepORLM prioritizes solver-verified trajectories in every preference pair, and VPRM's ablation shows that removing the outcome term collapses accuracy from 87.9 to 40.2, so process and outcome verification are complements, exactly as Uesato et al. [2022] first suggested. Second, verification changes what can be measured: coherence [Pronesti et al., 2026] and the tool gap [Zhao et al., 2026] are diagnostics that only exist because an external decision function or tool boundary exists, and both reveal failures – decorative step-level reasoning, judge collapse on instrumented steps – that accuracy on a mathematical benchmark would never surface.

Finally, the discovery of implicit process rewards [Yuan et al., 2024] and step-level self-rewarding [Zhang et al., 2025c] together suggest that, at least for math reasoning, a substantial fraction of the value of explicit PRM training data can be recovered directly from outcome-level preference learning or from a model's own step-wise self-judgments – provided the base/cold-start model and the DPO reference policy are well calibrated. StepORLM's co-evolutionary loop can be read as the domain-grounded counterpart of this idea: rather than trusting a model to judge itself, it anchors each self-improvement round to a solver, so the loop cannot drift. Taken together, Tables 1 and 2 and Figures 5 and 4 summarize how the twelve papers surveyed above relate to and extend this landscape: they neither replace the classical method families nor sit outside them, but rather stress-test and generalize each family along the axis of annotation reliability, distributional robustness, domain scope, or signal grounding.

6 Open Problems and Future Directions

Beyond the category-specific limitations already discussed in each *Limitations & Future Directions* box, we highlight six cross-cutting open problems that the surveyed work makes especially visible.

(a) Annotation validity, not just annotation cost. Early work on automating PRM data focused almost entirely on *reducing* annotation cost [Wang et al., 2023, Luo et al., 2024]. Zhang et al. [2025b] reframe the problem: even abundant automated labels can measure the wrong thing (a value model rather than a PRM), and even a well-established evaluation protocol (BoN) can systematically favor PRMs that behave like ORMs. Future PRM papers should report step-level localization metrics (as in ProcessBench/PRMBench) alongside BoN, and should audit for the specific biases Zhang et al. identify.

(b) Generalization across policies, modalities, and domains. AURORA, VisualPRM, and OpenPRM each generalize a PRM along a different axis (policy diversity, modality, domain) using a different mechanism, and

the papers of §4.7 add a fourth axis, verification regime. No existing work combines all four, and no existing benchmark tests them simultaneously (§4.8). Building a single universal PRM – and a single benchmark to test it – that spans math, code, multimodal, scientific, clinical, and open-domain reasoning under diverse policies remains open.

(c) Reflective and self-correcting reasoning at scale. Yang et al. [2025] show that first-error truncation, the default assumption behind most PRM training data, actively mis-annotates the growing fraction of long-CoT solutions that self-correct. As reasoning models increasingly produce long, reflective traces by default, re-annotating or re-collecting training data under Error Propagation/Cessation-style rules (or an automated analogue that does not depend on an expensive judge like o1) is likely to become necessary for most PRM pipelines. Notably, the tool-augmented traces of Zhao et al. [2026] are reflective by construction – the model reacts to real execution feedback – so this concern is not confined to mathematics.

(d) Efficient generative and self-supervised verification. Generative verifiers (§4.5) and self-rewarding / implicit approaches (§4.4) both reduce the need for external step labels, but at the cost of either inference-time compute or training-time iteration count. Zhao et al. [2026] give the sharpest version of this trade-off outside mathematics: real tool execution buys ten accuracy points at a $370\times$ latency penalty, which is affordable at evaluation time and prohibitive inside an RL loop. A unified cost–accuracy analysis across generative, implicit, and verifier-in-the-loop scoring – and hybrids that adaptively choose among them by task difficulty – remains largely unexplored.

(e) The proxy-reward gap. Even the strongest PRMs in this survey leave a large gap to the oracle/coverage ceiling under scaled inference-time sampling: OpenPRM shows pass@N climbing toward 100% as N grows to 128 while BoN accuracy with *any* reward model plateaus or degrades past $N \approx 16$–32 [Zhang et al., 2025d], and Zhao et al. [2026] report the same shape against an empirical Best-of-N upper bound in four scientific domains. This mirrors the general finding that classification accuracy on a static benchmark does not guarantee proportional downstream utility during search or RL [Song et al., 2025]. Closing this gap likely requires reward models that are explicitly calibration-aware and exploration-aware, rather than purely optimized for pointwise or pairwise classification accuracy.

(f) The reach and limits of verifiability. §4.7 shows that where a deterministic verifier exists, grounding the process reward in it dominates learned judging on accuracy, interpretability, and resistance to reward hacking, and Pronesti et al. [2026] supply a theoretical guarantee to match. The open question is how far this extends. Three sub-problems stand out: *elicitation* – can verifiers be synthesized, or drafted by an LLM and audited by experts, for domains whose norms are implicit rather than codified? *Partial coverage* – most realistic trajectories mix verifiable sub-steps with genuinely open-ended ones, so we need principled hybrid rewards that degrade gracefully to judge-based scoring with calibrated weights rather than switching regimes abruptly. And *format brittleness* – a verifiable reward silently vanishes when the model's output drifts from the schema the verifier expects, a failure mode that disproportionately affects small models and that no current work measures systematically.

7 Conclusion

Process supervision has fundamentally transformed how large language models are optimized and verified for complex, multi-step reasoning. This survey organizes the resulting literature into eight categories spanning the full PRM lifecycle – foundational human annotation, automated discriminative labeling, generalization-aware and reflective-CoT training, implicit/RL-based supervision, generative verification, architectural and cross-domain innovations, domain-specialized and verifiable supervision, and benchmarking. It covers twelve

papers that push the field toward more *reliable* annotation and evaluation [Zhang et al., 2025b], more *general* PRMs that withstand unfamiliar policies and reflective long-CoT reasoning [Tan et al., 2025, Yang et al., 2025], more *label-efficient* self-supervised and generative verification [Zhang et al., 2025c, She et al., 2025], broader *scope* into multimodal and open-domain reasoning [Wang et al., 2025, Zhang et al., 2025d], and more firmly *grounded* supervision signals drawn from unit tests, clinical guidelines, scientific tools, optimization solvers, and deterministic decision rules [Li et al., 2025, Yun et al., 2025, Zhao et al., 2026, Zhou et al., 2025, Pronesti et al., 2026].

Taken together with the taxonomy figure, development-loop diagram, supervision-signal spectrum, and comparison tables introduced here, the surveyed work suggests that the next phase of PRM research will be defined less by whether step-level supervision helps – that question is now well answered – and more by *what the step label is actually measuring*. The trajectory of the field runs from outcome labels, to rollout-based estimates of answer reachability, to learned judges, to externally checkable verification; each move buys fidelity at the price of generality, and the central engineering problem of the coming years is how to combine them, so that process supervision can be made reliable, general, and efficient enough to supervise reasoning models whose behavior – long, reflective, multimodal, tool-using, and increasingly deployed in domains where being right for the wrong reason is unacceptable – keeps outpacing the assumptions baked into first-generation PRM pipelines.

References

Wenxiang Chen, Wei He, Zhiheng Xi, Honglin Guo, Boyang Hong, Jiazheng Zhang, Nijun Li, Tao Gui, Yun Li, Qi Zhang, and Xuanjing Huang. 2025. Better process supervision with bi-directional rewarding signals. *arXiv preprint arXiv:2503.04618*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. 2025. PRIME: Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. 2025. ThinkPRM: Process reward models that think. *arXiv preprint arXiv:2503.09999*.

Qingyao Li, Xinyi Dai, Xiangyang Li, Weinan Zhang, Yasheng Wang, Ruiming Tang, and Yong Yu. 2025. CodePRM: Execution feedback-enhanced process reward model for code generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8169–8182.

Weijie Li, Jin Wang, Liang-Chih Yu, and Xuejie Zhang. 2026. Step-GRPO: Enhancing reasoning quality and efficiency via structured PRM-based reinforcement learning. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence (AAAI-26)*.

Wendi Li and Yixuan Li. 2024. Process reward model with Q-value rankings. *arXiv preprint arXiv:2410.11287*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

Qianli Ma, Haotian Zhou, Tingkai Liu, Jianbo Yuan, Pengfei Liu, Yang You, and Hongxia Yang. 2023. Let's reward step by step: Step-level reward model as the navigators for reasoning. *arXiv preprint arXiv:2310.10080*.

Massimiliano Pronesti, Anya Belz, and Yufang Hou. 2026. Beyond outcome verification: Verifiable process reward models for structured reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 32187–32202.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. 2024. Rewarding progress: Scaling automated process verifiers for LLM reasoning. *arXiv preprint arXiv:2410.08146*.

Shuaijie She, Junxiao Liu, Yifeng Liu, Jiajun Chen, Xin Huang, and Shujian Huang. 2025. R-PRM: Reasoning-driven process reward modeling. *arXiv preprint arXiv:2503.21295*.

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. 2025. PRMBench: A fine-grained and challenging benchmark for process-level reward models. *arXiv preprint arXiv:2501.03124*.

Michael Sullivan and Alexander Koller. 2026. GRPO is secretly a process reward model. *arXiv preprint arXiv:2602.04567*.

Xiaoyu Tan, Tianchu Yao, Chao Qu, Bin Li, Minghao Yang, Dakuan Lu, Haozhe Wang, Xihe Qiu, Wei Chu, Yinghui Xu, and 1 others. 2025. AURORA: Automated training framework of universal process reward models via ensemble prompting and reverse verification. *arXiv preprint arXiv:2502.11520*.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Peiyi Wang, Lei Li, Zhihong Shao, R.X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. 2023. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*.

Zihan Wang, Yunxuan Li, Yuexin Wu, Liangchen Luo, Le Hou, Hongkun Yu, and Jingbo Shang. 2024. Multi-step problem solving through a verifier: An empirical analysis on model-induced process supervision. *arXiv preprint arXiv:2402.02658*.

Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, and 1 others. 2025. VisualPRM: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*.

Zhaohui Yang, Chenghua He, Xiaowen Shi, Linjing Li, Qiyue Yin, Shihong Deng, and Daxin Jiang. 2025. Beyond the first error: Process reward models for reflective mathematical reasoning. *arXiv preprint arXiv:2505.14391*.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. 2024. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*.

Jaehoon Yun, Jiwoong Sohn, Jungwoo Park, Hyunjae Kim, Xiangru Tang, Yanjun Shao, Yonghoe Koo, Minhyeok Ko, Qingyu Chen, Mark Gerstein, Michael Moor, and Jaewoo Kang. 2025. Med-PRM: Medical reasoning models with stepwise, guideline-verified process rewards. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16554–16571.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2025. Generative verifiers: Reward modeling as next-token prediction. In *International Conference on Learning Representations (ICLR)*.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Shimao Zhang, Xiao Liu, Xin Zhang, Junxiao Liu, Zheheng Luo, Shujian Huang, and Yeyun Gong. 2025. Process-based self-rewarding language models. *arXiv preprint arXiv:2503.03746*.

Kaiyan Zhang, Jiayuan Zhang, Haoxin Li, Xuekai Zhu, Ermo Hua, Xingtai Lv, Ning Ding, Biqing Qi, and Bowen Zhou. 2025. OpenPRM: Building open-domain process-based reward models with preference trees. In *The Thirteenth International Conference on Learning Representations (ICLR)*.

Jian Zhao, Runze Liu, Kaiyan Zhang, Zhimu Zhou, Junqi Gao, Dong Li, Jiafei Lyu, Zhouyi Qian, Biqing Qi, Xiu Li, and Bowen Zhou. 2025. GenPRM: Scaling test-time compute of process reward models via generative reasoning. *arXiv preprint arXiv:2504.00891*.

Xiangyu Zhao, Henry Hengyuan Zhao, Yiheng Wang, Wanghan Xu, Yuhao Zhou, Qinglong Cao, Zhiwang Zhou, Lei Bai, Wenlong Zhang, and Xiao-Ming Wu. 2026. Sci-PRM: A tool-aware process reward model for scientific reasoning verification. *arXiv preprint arXiv:2606.04579*.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. ProcessBench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*.

Congmin Zheng, Jiachen Zhu, Zhuoying Ou, Yuxiang Chen, Kangning Zhang, Rong Shan, Zeyu Zheng, Mengyue Yang, Jianghao Lin, Yong Yu, and Weinan Zhang. 2025. A survey of process reward models: From outcome signals to process supervisions for large language models. *arXiv preprint arXiv:2510.08049*.

Chenyu Zhou, Tianyi Xu, Jianghao Lin, and Dongdong Ge. 2025. StepORLM: A self-evolving framework with generative process supervision for operations research language models. *arXiv preprint arXiv:2509.22558*.