Pessimistic Verification for Open-Ended Math Questions

Yanxing Huang[1,2], Zihan Tang[1], Zejin Lin[1], Peng Li[2], Yang Liu[2,3]

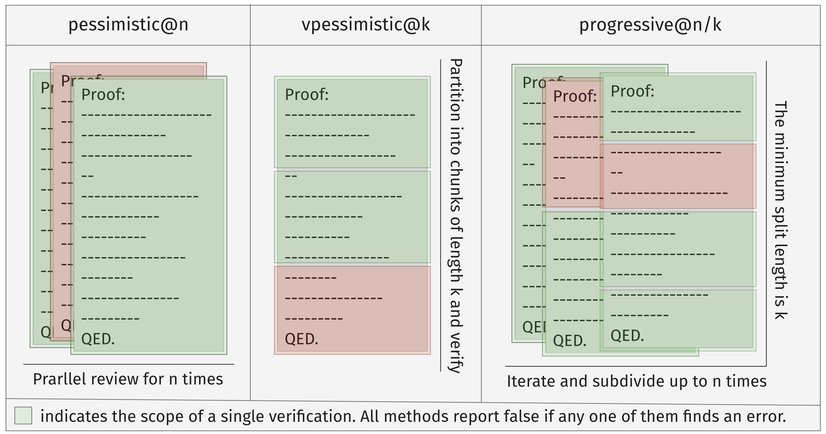
1. Qiuzhen College, Tsinghua University, Beijing, China
2. Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China
3. Dept. of Comp. Sci & Tech., Institute for AI, Tsinghua University, Beijing, China

Correspondence to: Peng Li <lipeng@air.tsinghua.edu.cn>, Yang Liu <liuyang2011@tsinghua.edu.cn>

ICML International Conference On Machine Learning

Background & Method

Automatic verification is a critical component in building math-solving agents and reinforcement learning. We argue that the primary bottleneck of verification lies in **error detection** capability, and proposed pessimistic verification, pessimistic verification (pverify), a paradigm of agentic workflows that rejects a solution if any of multiple parallel verifiers identifies a flaw.



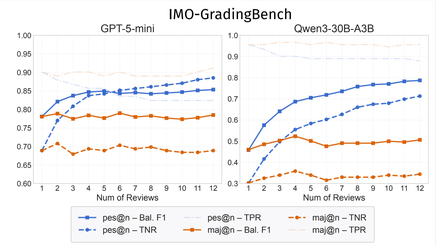
Progressive pessimistic verification further employs fine-grained **proof decomposition** that significantly enhance verification accuracy and efficiency.

Key Findings

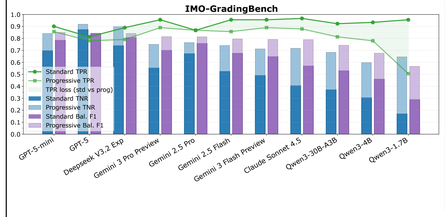
We adopted balanced F1 score as the main metric, and empirically ovserved these key findings.

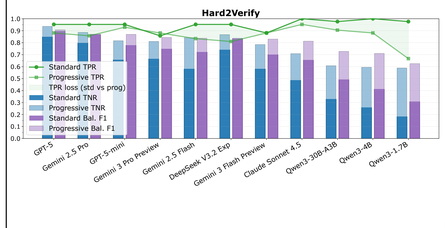
$$\text{bal. F1} = \frac{2\,\text{TPR}\,\text{TNR}}{\text{TPR} + \text{TNR}}$$

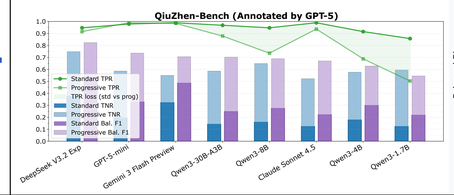
TPR: true positive rate,
TNR: true negative rate.

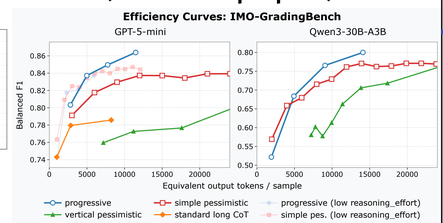


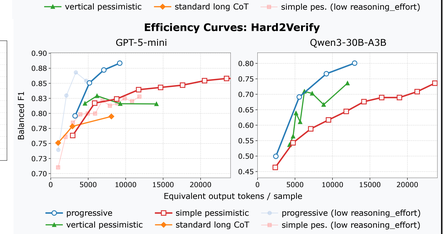
1. pverify can significantly improve TNR while keeping TPR stable, and **steadily improve bal. F1 score**.
2. majority voting has almost no effect on verification.
3. pverify is **more effective than long CoT** in test-time scaling for verification with extented token usage.
4. progressive pverify exhibits even higher efficiency, and is **effective across many models**.
5. the effectiveness of pverify on cutting-edge models is hindered by annotation noise (see full paper).

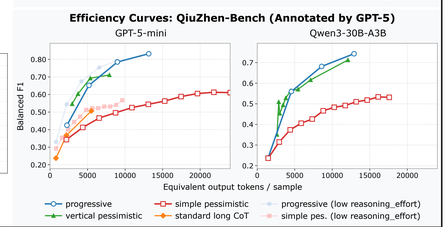


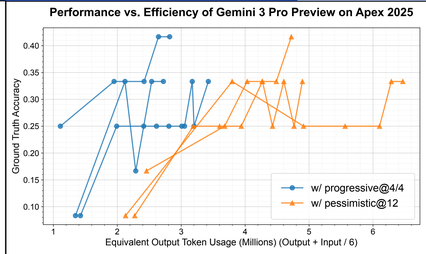


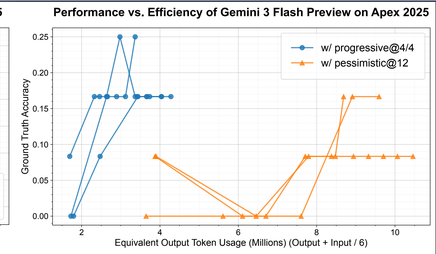












We further validated pverify on a verifier-centered iterative workflow for math problem solving on two extremely challenging datasets, IMO 2025 and Apex 2025. Frontier models presented **significant performance boost** compared to single-pass generation, and progressive pverify has notably higher efficiency on these tasks.

Discussions

Pverify could be especially useful in these scenarios:
1. Verification process typically dominates the token usage in **math agents**, while integrating pverify would significantly reduce the cost with a improved perf.
2. It can be used in **reinforcement learning for math** to provide better signal for training verifiers and eventually the prover.
pverify might also be effective in other tasks such as code reviewing in coding agents.

Thank you for your attention & star!