# Compute

Institutional Equity Research
November 28, 2025

## This Thanksgiving We Were Thankful for Open-Source AI Labs

*Thanksgiving reminded us just how grateful we are to have open-source AI labs as DeepSeek released DeepSeekMath-V2 alongside a detailed technical paper that lays out, step by step, how they think about training a frontier math model. Below we share our high-level takeaways, with more detailed analysis of the paper on the following pages.*

**DeepSeekMath-V2 shows that verification could be the key unlock to frontier math capabilities.** For the past year, most reinforcement learning for math has rewarded models on whether they correctly hit the final numeric answer, which is easy to score in benchmarks but a poor proxy for correct reasoning and useless for formal theorem proving. DeepSeekMath-V2 inverts this setup, and instead of training directly against an answer key, DeepSeek trained a verifier LLM on natural-language proofs to assign 0/0.5/1 scores and produce detailed critiques. They then layer on a meta-verifier that reads the original proof plus the verifier's analysis and judges whether the criticisms are real, the math checks out, and the final score is justified. Human experts seed both components, but the meta-verifier ultimately becomes the gatekeeper for verifier behavior. On top of this stack, the proof generator is optimized with GRPO to do two things at once: (1) produce proofs that earn high verifier scores and (2) produce self-evaluations whose scores and listed issues line up with what the verifier + meta-verifier will later conclude. In other words, the system is trained to write proofs that would pass an IMO grader and to calibrate its own confidence to that internal grading loop. *More details on the following pages.*

**Scaling verifier compute at training and inference becomes essential.** Once verification is the training signal, the real bottleneck is grading capacity, not model size. Because human expert annotation quickly becomes the constraint, DeepSeek pushes as much as possible onto the verifier stack. During training, they use multiple verifier samples per proof and then multiple meta-verifier passes per verifier analysis to decide which critiques reflect real issues. If enough independent analyses converge on a low score with validated issues, the proof is auto-labeled. But if no issues survive meta-verification across many attempts, then the proof is labeled correct. In the last two training iterations, this fully automatic pipeline replaces human annotation entirely, which is how they escape the obvious "expert grader bottleneck" you'd otherwise hit. At inference, the model performs an aggressive search where it generates 64 proofs and analyses, refines low-scoring proofs under new analyses, and then iterates for up to 16 rounds or until one proof survives all 64 verification attempts.

**DeepSeek always comes back when the open-source community needs them most.** When efficiency looked like a closed-lab advantage, DeepSeekMoE gave everyone a concrete path to order-of-magnitude compute savings. When reasoning looked like a proprietary trick, DeepSeek-R1 showed that long-chain thinking could be trained with relatively simple RL. This year, OpenAI and Google made it seem as though frontier-level contest math would remain a closed frontier after their IMO golds, but DeepSeekProver-V2 and now DeepSeekMath-V2 have handed the field a clear recipe for scaling mathematical capabilities to that level. If anything, the pattern is that DeepSeek drifts off-cycle, then reappears with a release that turns apparent "secret sauce" into a public training loop. Which makes us more, not less, confident that DeepSeek-V4 will matter.

**Math capabilities in frontier models are still widely understated.** Even having a prover of this caliber in hand, we'd argue we are much closer to "frontier math research in the commons" than it looked even six months ago. The market still mostly treats math as a niche benchmark story rather than an emerging application domain where specialized prover-verifier stacks are likely to drive real progress ahead of general-purpose models. DeepSeek themselves frame this release as an early step toward self-verifiable mathematical reasoning and point explicitly to research-level mathematics as a long-term target.

## INDUSTRY UPDATE

Price (11/28/25)
Industry: TECHNOLOGY

**Alexander Platt**
(503) 603-3045
AJPlatt@dadco.com

### DaVinci Overview

D.A. Davidson's DaVinci initiative focuses our technical-oriented research, data-driven insights, and prescient think pieces under one unified framework. We note that for our DaVinci coverage of deep tech businesses, we employ an early-stage venture approach focusing on technical foundations, disruptive potential, and long-term strategic value, rather than near-term financial and valuation metrics given the unique growth trajectories of pre-inflection markets.

# Taking a Closer Look at DeepSeek's Approach

Below, we provide a break-down the architecture and design of DeepSeekMath-V2 by going through the Methods section of the technical report in order to gain a better grasp of DeepSeek's approach and the benefits of doing so.

## Proof Verification

Conceptually, DeepSeek treats proof verification as its own sequence to sequence task where, given a problem statement X and a natural language proof Y, a verifier policy takes X and Y as input and produces two outputs. First it produces a structured analysis V written in natural language. Second it produces a single numerical score which can be {0, 0.5, 1}. This score is meant to behave like the score an Olympiad grader would assign to that solution. The central design choice is that the verifier is not trained to give a simple correct or incorrect label on the final answer. Instead, it is trained to mimic human grading rubrics that care about whether the proof is complete, whether every step is justified, and whether referenced lemmas are actually proved in the solution rather than appealed to as black boxes. In effect the verifier is pushed to internalize the hidden evaluation function that human judges apply to long form Olympiad solutions, rather than acting as a checksum on the final boxed answer. To get an initial training set for this verifier, they assemble a proof problem corpus by scraping on the order of 17,000 proof style problems from Art of Problem Solving. They bias toward problems after roughly 2010 and toward Olympiad and selection problems that explicitly request proofs rather than short answers. On top of these problems they run a non theorem optimized DeepSeek V3.2 experimental thinking model and use it as a noisy proof generator. They prompt this model to iteratively refine its own solutions so that the outputs become long and detailed rather than terse one line arguments. This produces a broad distribution of proofs that range from completely wrong, to partially correct with gaps, to fully correct and rigorous. Human experts then grade sampled proofs using a three level rubric. A score of 1 means the proof is fully rigorous. A score of 0.5 means the proof is essentially correct but has minor gaps or presentation issues. A score of 0 means the argument is fundamentally broken. This yields a reinforcement learning dataset where each example consists of a problem X, a candidate proof Y, and a human score s that is either 0, 0.5, or 1.

The verifier is initialized from a DeepSeek V3.2 experimental SFT model that has already been supervised on math and code reasoning. They then further train it with reinforcement learning so that, when given X and Y, it produces both a textual analysis V prime and a predicted score s prime that match the human labels while also obeying a rigid output format. The reward shaping here is simple but crucial. There is a format reward, which we can call the format term, that acts like an indicator. It only gives positive reward if the verifier output starts its analysis with the exact phrase “Here is my evaluation of the solution” and ends with a score written inside a boxed template after a specific fixed phrase. This forces the verifier to follow a structured protocol that later components can reliably parse. On top of the format term sits a score reward that depends on how close the predicted score s prime is to the human score s. If the two scores exactly match, this reward is at its maximum. As the difference between s prime and s increases, the reward decreases linearly. The overall training objective is to choose verifier behavior so that, when you average over problems and proofs in the dataset and over the verifier’s own sampled outputs, the product of the format reward and the score reward is as large as possible. This turns the verifier into a graded critic that must do two things at once. It must classify proofs into one of three bands that correspond to fully correct, almost correct, and incorrect. At the same time it must emit a human-readable commentary that downstream systems and human readers can consume. The content of this commentary is not directly supervised beyond the weak requirement that it accompany the right scalar score. That weak supervision creates a serious failure mode. Under this objective the verifier can cheat in a way that does not hurt its reward. Given a flawed proof whose true score is 0 or 0.5, the model can maximize its reward by outputting the correct numerical score while hallucinating arbitrary issues in the analysis text, including defects that do not actually exist or misidentifying where the proof goes wrong. Because the reinforcement learning signal only cares about whether the numeric score matches the human label, and not about whether the critique is factually correct, there is no training pressure that pushes the explanation to align with the proof.

This behavior is catastrophic if you want to use the verifier as a reward model for theorem proving reinforcement learning, because then the generator will learn to optimize against a critic that rewards fake scrutiny and may overlook real errors. The authors address this by introducing a second model, the meta verifier, whose job is to grade the verifier’s analysis itself. The meta verifier sees the original problem X, the proof Y, the verifier’s analysis V, and a separate set of rubrics that describe what a good evaluation should do. Those rubrics require that the analysis accurately restates the relevant parts of the proof, correctly confirms or rejects alleged defects, avoids expression errors and hallucinations, and assigns a final quality score that reflects whether the identified issues are real and whether the proof score is justified. The meta verifier then outputs its own analysis and a meta score which again takes one of three values 0, 0.5, or 1 and reflects how well the verifier has done its job on this particular proof. Training the meta verifier mirrors the first stage. They first freeze an initial verifier and run it across the verifier training dataset to produce analyses V for many proofs. Human experts then annotate these verifier analyses with meta scores under the meta rubrics, which creates a meta dataset where each entry consists of a problem, a proof, a verifier analysis, and a human meta score. The meta verifier is then trained with reinforcement learning using the same two part reward structure as before, with a format term that enforces a rigid output template and a score term that rewards agreement between the meta verifier’s scalar score and the human meta score. In this stage the scalar target is the meta score rather than the original proof score.

Once the meta verifier is competent, they fold it back into verifier training by augmenting the verifier's reward with an additional meta term. Concretely, for each verifier output they compute the format reward, the score reward that measures how close the verifier's proof score is to the human proof score, and a meta reward that reflects the quality score assigned by the meta verifier to this analysis. The total reward for the verifier is the product of these three pieces. They then retrain the verifier jointly on the original proof dataset and the meta dataset under this augmented objective. Under this new training signal, a verifier that gives the right numeric proof score but invents bogus issues in the commentary will receive a low meta reward and therefore a low overall reward. The only way to achieve high return is to produce an analysis that both matches expert proof scores and describes issues in a way that survives a second pass audit by the meta verifier, which has been optimized to detect misaligned, spurious, or hallucinated criticism. In experiments on a held out part of the verifier dataset, the average meta verifier score of verifier outputs rises from ~0.85 to ~0.96 while the accuracy of the proof scores themselves stays about the same. This is strong evidence that the augmented training has removed a large fraction of hallucinated issues without degrading the calibration of the scalar proof scores.

## Proof Generation

Once the verifier is in place, DeepSeek repurposes it as a generative reward model for a theorem proving policy that plays the role of the proof generator. The verifier has been trained to emulate expert grading over natural language proofs, so the generator's job is to produce proofs that this learned critic scores highly. In practice, they sample Olympiad style problems from the Art of Problem Solving distribution, draw candidate proofs from the generator policy for each problem, and feed the problem and candidate proof into the verifier under the established rubrics. The verifier returns a proof score, again 0, 0.5, or 1, which is interpreted as the reward R sub Y for that proof. The reinforcement learning objective for the generator can be understood as follows. You sample a problem from the problem distribution, sample a proof from the generator conditioned on that problem, feed problem and proof into the verifier, and read off the resulting score. You then adjust the generator so that the expected value of that score is as large as possible. In words the only way for the generator to achieve high return is to produce proofs that the verifier judges to be rigorous and complete under the human like rubrics it has learned. This means the generator is never trained directly against an answer key or final numeric answer. It is optimized against the learned rubric based notion of proof quality that the verifier has internalized. That is a conceptual break from standard reinforcement learning setups for math which reward models for hitting the right final answer without regard to the reasoning path. Here the generator's behavior is shaped by a critic that cares about stepwise justification, coverage of all cases, and the absence of fatal gaps rather than merely matching a hidden integer.

The authors then confront a failure mode that is familiar from current reasoning models. If you prompt a strong language model to both solve a contest problem and comment on its own solution in one shot, it typically declares success even when an external verifier can easily find bugs. There is a gap between generation and verification inside the same model. The generator can refine proofs when a dedicated verifier hands it structured feedback, yet when the generator is asked to judge itself it defaults to overconfident self praise. The self verification training phase is designed to close this gap by forcing the generator to internalize the verifier's attitudes and grading standards so that its self evaluations become constrained by the same rubrics that govern training. During self verification training, the generator is prompted to produce a two part output for each problem. First it emits a candidate proof Y. Then, in the same completion, it emits a self-analysis Z that mirrors the verifier protocol. The self-analysis is expected to be a prose discussion of issues framed by the same rubrics as the verifier, followed by a self-assigned proof score s prime which again can be 0, 0.5, or 1. The verifier stack from the verification section is then used to score both parts. The proof Y is sent through the verifier, which returns a rubric based proof score s. This proof score is treated as the reward R sub Y for the proof. The self-analysis Z is sent through the meta verification machinery, which returns a meta score that describes how accurate and justified the self critique is. The system also compares the self-assigned score s prime to the verifier's score s.

From these pieces they construct a second reward component for the generator, which we can call R sub Z. This self-analysis reward is high when two conditions hold at once. First the self-assigned score s prime agrees with the verifier's score s, so the generator is honest about how good its proof is. Second the meta verifier judges the self-analysis as factually correct and well justified. If either condition fails, the self-analysis reward drops. The overall reward for the generator on each example is the product of a format term that enforces the required output template and a weighted sum of the proof reward and the self-analysis reward. In the implementation they set the weight on the proof reward to about 0.67 and the weight on the self-analysis reward to about 0.24, so the proof quality remains the dominant term but self evaluation quality still matters. This reward design encodes a specific set of behavioral incentives. The generator is still pushed hardest toward emitting actually correct proofs under the verifier's grading. At the same time the self-analysis term means the model can improve its expected return even when it cannot yet reach perfect correctness by learning to truthfully diagnose its own failures. If the generator outputs a flawed proof but candidly acknowledges the defect, assigns itself a low score that matches the verifier, and describes the issue in a way that passes meta verification, it earns more reward than if it tries to bluff its way to a perfect self score. Conversely, if the generator claims the proof is fully correct when the verifier assigns a lower score, the agreement term collapses toward 0 and the meta verifier is likely to give a low quality score to the self-analysis, which sharply penalizes dishonest self praise. The optimal behavior in this regime is to search for proofs that the verifier will score as fully correct and, when that is not yet possible, to surface genuine issues and use them as guidance for further refinement. Over time this training loop pushes the generator to approximate an internal copy of the verifier's judgment function rather than maintaining the naive prior that its own outputs are always right.

## Synergies Between Proof Verification and Generation

The starting observation in the synergy section is straightforward. Once you have a reasonably strong verifier, you can use it as a reward model to push the generator up the difficulty curve. As the generator improves, it starts producing proofs that lie exactly at or just beyond the verifier's current decision boundary. Those hard to verify proofs are often the ones where the verifier may miss subtle gaps or mis score borderline arguments, which makes them ideal new training data for improving verification. In that sense the generator is not only a consumer of the verifier. It is also a source of adversarially challenging proofs that expose weaknesses in the verification stack. The main bottleneck is how to label these newly generated proofs with trustworthy correctness scores. In principle you can ask human experts to score them, but at IMO and Putnam levels the proofs and error modes become long and subtle, so manual grading quickly grows expensive and slow. DeepSeek's solution is to use the verifier and meta verifier stack as an annotation accelerator. For each candidate proof they do not trust a single verifier pass to decide correctness. Instead they run multiple independent verification analyses per proof. Each run starts from a different random seed or sampling trajectory, so the verifier explores different ways of critiquing the proof. The goal of this verification ensemble is not to average scores. It is to surface as many distinct potential issues as possible. Humans or downstream models can then inspect these issues rather than having to discover them from scratch.

During this AI assisted annotation process they make two key empirical observations. First, simply increasing the number of verifier samples per proof significantly raises the chance that, if the proof is flawed, at least one analysis will catch a real bug. Second, checking whether a claimed issue is real is substantially easier than finding issues from nothing, both for humans and for language models. Reviewing the verifier's list of alleged issues is exactly the meta verification task that the meta verifier was trained for, and they already see that this is a more sample efficient learning problem. They then crystallize these insights into a fully automated labeling pipeline that, in later training iterations, removes humans from the loop. For each proof emitted by the generator they first generate a fixed number n of independent verifier analyses. This yields a collection of analyses. Some of them may assign a perfect score of 1 with no listed issues. Others may assign a score of 0 or 0.5 and list specific defects. For every analysis that reports issues, meaning any analysis that does not give a perfect score, they apply the meta verifier a fixed number m of times to that analysis. Each meta verification run produces a judgment about whether the reported issues are real and whether the assigned score is justified. They treat this as a voting process. An analysis is considered valid only if a majority of these meta verification runs confirm its findings. This step filters out verifier hallucinations and overreactions by using the meta verifier as a check on the verifier's own critiques.

Next they look across all analyses for that proof and focus on those that assign the lowest score present among 0, 0.5, and 1. If at least a threshold number k of these lowest score analyses are validated by meta verification, the proof itself is labeled with that lowest score, since the system has converged on a robust diagnosis that the proof is fundamentally flawed or only partially correct. If no legitimate issues survive this pipeline across all analyses, the proof is labeled with score 1. If neither condition is met, for example if some analyses claim issues, but meta verification cannot agree on their validity and there is no strong consensus, they discard the proof or, in earlier phases, send it to human experts. The result is an automatic annotator that treats correctness as the absence of any validated issues after scaled verification attempts, rather than as whatever the first verifier pass happens to say. In the final two training iterations they report that this fully automated pipeline replaced human annotation entirely, with spot checks indicating that the labels match expert judgments closely. At that point verification and generation enter a genuine virtuous cycle. The verifier and meta verifier jointly produce labels for proofs generated by the theorem proving policy. Those labels are used to further train and sharpen the verifier. The improved verifier then supports stronger generator training. The stronger generator in turn produces even more challenging proofs, which feed into the next round of verifier refinement.

## Potential Risks

## Required Disclosures

D.A. Davidson & Co, or any of its affiliates, does or seeks to do business with companies covered in its research reports. As a result, investors should be aware that the firm may have a conflict of interest that could affect the objectivity of this report. Investors should consider this report as only a single factor in making their investment decision.

D.A. Davidson & Co. is a full service investment firm that provides both brokerage and investment banking services. Alexander Platt, the research analyst principally responsible for the preparation of this report has received and is eligible to receive compensation, including bonus compensation, based on D.A. Davidson's overall operating revenues, including revenues generated by its investment banking and institutional equities activities. D.A. Davidson & Co.'s analysts, however, are not directly compensated for involvement in specific investment banking transactions.
I, Alexander Platt, attest that (i) all the views expressed in this research report accurately reflect my personal views about the common stock of the subject company, and (ii) no part of my compensation was, is, or will be, directly or indirectly, related to the specific recommendations or views expressed in this report.

**Rating Information**
**D.A. Davidson & Co.'s Institutional Research Rating Scale Definitions** (maintained since October 10, 2017); information regarding our previous definitions is available upon request:
**BUY:** Expected to produce a total return of over 15% on a risk adjusted basis over the next 12-18 months
**NEUTRAL:** Expected to produce a total return of -15% to +15% on a risk adjusted basis over the next 12-18 months
**UNDERPERFORM:** Expected to lose value of over 15% on a risk adjusted basis over the next 12-18 months

| Rating Distribution (as of 9/30/25) | Coverage Universe Distribution | | | Investment Banking Distribution | | |
|---|---|---|---|---|---|---|
| | IR | WMR | Combined | IR | WMR | Combined |
| BUY (Buy) | 59% | 85% | 62% | 8% | 0% | 7% |
| NEUTRAL (Hold) | 40% | 13% | 37% | 4% | 0% | 3% |
| UNDERPERFORM (Sell) | 1% | 2% | 1% | 0% | 0% | 0% |

*IR denotes Institutional Research; WMR denotes Wealth Management Research whose rating scale is Buy/Add, Neutral, Sell/Reduce. Investment Banking Distribution denotes companies from whom D.A. Davidson & Co. has received compensation in the last 12 months. Best-of-Breed: Expected to outperform on a risk adjusted basis over a five-year time horizon.*

Target prices are our Institutional Research Department's evaluation of price potential over the next 12 months, based upon our assessment of future earnings and cash flow, comparable company valuations, growth prospects and other financial criteria. Certain risks may impede achievement of these price targets including, but not limited to, broader market and macroeconomic fluctuations and unforeseen changes in the subject company's fundamentals or business trends.
While the Best-of-Breed designation does not contain a separate rating and/or price target from that of the standard ratings system referenced above, the expectation is that the security, based on the 12 criteria utilized in assessing the "Best-of-Breed" designation, will outperform over a five-year time horizon, not the standard 12-18 month time horizon.

For a copy of the most recent reports containing all required disclosure information for covered companies referenced in this report, please contact your D.A. Davidson & Co. representative or call 1-800-755-7848.

## Other Disclosures

Information contained herein has been obtained by sources we consider reliable, but is not guaranteed and we are not soliciting any action based upon it. Any opinions expressed are based on our interpretation of data available to us at the time of the original publication of the report. These opinions are subject to change at any time without notice. Investors must bear in mind that inherent in investments are the risks of fluctuating prices and the uncertainties of dividends, rates of return and yield. Investors should also remember that past performance is not necessarily an indicator of future performance and D.A. Davidson & Co. makes no guarantee, express or implied, as to future performance. Investors should note this report was prepared by D.A. Davidson & Co.'s Institutional Research Department for distribution to D.A. Davidson & Co.'s institutional investor clients and assumes a certain level of investment sophistication on the part of the recipient. Readers, who are not institutional investors or other market professionals, should seek the advice of their individual investment advisor for an explanation of this report's contents, and should always seek such advisor's advice before making any investment decisions. Consensus estimates are obtained from Capital IQ. Further information and elaboration will be furnished upon request.

## Other Companies Mentioned in this Report

| Company Name | Ticker | Rating | Price |
|---|---|---|---|
| **Apple Inc.** | AAPL | NEUTRAL | $277.55 |
| **Amazon.com, Inc.** | AMZN | BUY | $229.16 |
| **CoreWeave, Inc.** | CRWV | UNDERPERFORM | $74.29 |
| **Alphabet Inc.** | GOOGL | NEUTRAL | $319.95 |
| **Meta Platforms, Inc.** | META | BUY | $633.61 |

| Company Name | Ticker | Rating | Price |
|---|---|---|---|
| **Microsoft Corporation** | MSFT | BUY | $485.50 |
| **Nebius Group N.V.** | NBIS | BUY | $94.69 |
| **NVIDIA Corporation** | NVDA | BUY | $180.26 |
| **Oracle Corporation** | ORCL | NEUTRAL | $204.96 |