Technical Memo

Superintelligence Research Lab at the Frontier of Deterministic Algorithms

echea.ai

Summary

We propose replacing multi-step, approximate stochastic gradient descent with one-shot, exact, deterministic discovery of optimal neural network weights. By specifying a target loss and reverse-finding the weights that achieve it, we eliminate iteration and approximation. Our approach leverages breakthroughs in algorithms for subclasses of problems for #P-complete counting problems to solve the underlying NP-complete optimization problem.

SAT solvers have seen a 10,000x improvement since 1980 driven entirely by algorithmic breakthroughs. Empirical scaling laws have produced better practical results in the current era, but the deterministic paradigm will return.

With this long-horizon goal in mind, the intermediate goal is to get these SAT solver algorithms working in industries like chip design, quantitative finance, and routing, where combinatorial optimization problems are commonplace. This is before generalizing towards becoming a frontier lab.

Problem

The empirical laws underpinning Deep Learning are the three scaling laws: data, compute, and algorithms. The last of these, namely the algorithmic training paradigm, has seen the least fundamental change. Gradient descent, the dominant optimization method, is a probabilistic and approximate technique. There are two core limitations:

- **Inefficiency.** Gradient descent requires many iterative steps, each computing approximate gradients over the loss landscape. The future capex requirements will reach trillions in order to fuel the cycle of growth.
- **No optimality guarantee.** There is no assurance that the converged solution is anywhere near the global optimum, routinely settling into local minima or saddle points. The large approximations necessary are barely understood leading to expensive computation and data requirements to iterate through the combinatorial explosion.

Modern SAT solvers share this limitation, also relying on heuristic search rather than principled algorithms with provable guarantees. Both fields have built impressive systems on top of fundamentally heuristic foundations.

These limitations are sufficient to reach a workable product, albeit computationally expensive, but the next paradigm requires deeper understanding.

Solution

Our approach is to develop fast, exact, and deterministic algorithms for NP-complete problems that can solve directly for the optimal weights of a loss landscape, bypassing gradient descent entirely.

NP-Completeness is broad. It can affect every domain of the superintelligence tech tree including algorithms (self-optimizing program synthesis), data (optimal Kolmogorov compression/complexity), compute (optimal efficiency of chips), and encryption (one way functions).

With a fast enough NP-Complete algorithm, it is possible to subsume all other potential LLM use-cases.

LLMs approximate the solution to NP-Complete problems, we want to solve them. This is superintelligence.

Current Stage

Our recent progress in counting the special case of tree graphs has given us a clear map of the surrounding territory. Beyond the tree case, slight variations of the underlying matrix have demonstrated the existence of matrices whose determinants compute certain #P-complete problems, even though the closed-form algorithm producing those matrices is not yet known. We now understand the structure of the problem well enough to brute-force closed-form solutions on broader graph families with a relatively modest amount of compute.

This converts the search for breakthroughs from mathematical insight into a predictable, repeatable computational process. The four-color theorem is the classical analogy: once the problem was reduced to a finite set of configurations, a computer verified each case exhaustively. We are now in the equivalent position, where further progress depends on systematic enumeration rather than on waiting for new theorems.

Although the general problem remains exponential, we have narrowed the sub-cases requiring verification to a tractable set.

Philosophy

We believe reaching superintelligence requires a fundamental shift away from the empirical scaling of heuristics and toward rational discoveries rooted in pure mathematics. The current paradigm treats training as an engineering problem, but no amount of scaling will overcome the fact that gradient descent offers no principled guarantee of finding the best solution. The philosophy of "make it, then make it better" has been achieved in AI. Now is the time to make it better with mathematics.

Superintelligence is a civilizational discovery, and historically, all discoveries of that magnitude have come from mathematics. It will emerge from a fundamental mathematical result that makes exact optimization tractable where only approximation existed before.

Subjects such as physics have already seen the changing outlook of discoveries from deterministic to stochastic to deterministic. Artificial Intelligence has seen one revolution in the stochastic paradigm, but the next one is deterministic. There are still many step-wise changes left to fully realize artificial intelligence.

Benchmark Roadmap

Our goal with the initial fundraise is to get through Phase 1 and Phase 2. Once we have completed these phases, we will be at the stage where we can start selling our algorithms as APIs to chip companies. We are currently benchmarking our algorithms against established competitions and internal tests, in the following sequence:

- **Phase 1 - Logic (Spring/Summer 2026).** Model Counting Competition (mccompetition.org) and SAT Solver Competition (satcompetition.github.io). Target: top-tier finish on structured instance families where our determinant methods apply.
- **Phase 2 - Chip Design (Fall 2026).** ISPD Contest benchmarks and internal synthetic benchmarks. Target: measurable speedup over state-of-the-art placement and routing tools.
- **Phase 3 - LLM Training (Early 2027).** Direct application to model pre-training optimization. Target: end-to-end training of a small foundation model via exact weight optimization.

Initial Applications

Neural network pre-training is one instance of a vast class of constrained optimization problems. Because NP-complete reductions are general-purpose, the same algorithmic advances apply across industries. This will form an API/MCP style CUDA environment where all these applications can be built on top of the same underlying algorithms. Initial applications include:

- **Chip design.** Well-studied reductions already exist from our logical formulations to chip-design NP-complete problems (e.g., placement and routing). This also opens potential partnerships with chip manufacturers.
- **Quantitative trading.** Optimal order scheduling and portfolio construction reduce to combinatorial problems in our target class.
- **Pharmaceuticals.** Molecular optimization and structure search map directly onto NP-complete counting over graph families.
- **Model training.** Using our algorithms to train an in-house foundation model via exact weight optimization, potentially faster, cheaper, and closer to optimal than gradient-descent-based training.

Lab Positioning

Echea Labs has operated as a research lab for four years, predating the recent wave of interest in alternative AI research paradigms. The majority of our pre-product, pre-revenue foundational work is already complete.

Our timeline to productization and real-world verification is approximately six months to one year. Unlike labs that are still exploring the space of possible paradigm shifts, we have identified a concrete and verifiable target and made substantial progress along it.

Appendix: Technical Approach

Formal Framing

The optimization that gradient descent approximates, non-convex function minimization, can be stated as two equivalent NP-complete problems:

- **Decision form:** "Does there exist a set of weights such that the output loss is less than k?"
- **Search form:** "Produce a set of weights such that the output loss is less than k."

NP-Complete algorithms come in the form where there is a quick way to verify if the solution is correct, but it is unknown how easy it is to find a solution. A sufficiently fast algorithm for the latter would eliminate the need for gradient descent altogether.

Research Direction: From NP-Complete to #P-Complete

Our original research goal was to build a fast SAT solver. However, progress in general-purpose SAT solving has plateaued, and the field occupies a niche position in most academic institutions. The source of this plateau is that the algorithmic breakthroughs in NP-complete solving have themselves been heuristic, in the same mold as gradient descent: clever search strategies, branching heuristics, and conflict-driven clause learning (domains of proof complexity) rather than pure mathematical which has a wider mathematical toolkit. The general field of proof complexity, combined with heuristics, has fewer technical resources than the field of #P-complete solving. This foreshadows the same plateau that gradient-descent-based algorithms will eventually hit, for the same reason.

We have shifted focus to a strictly more powerful complexity class: #P-completeness. NP-complete solvers ask whether a solution exists, whereas a #P-complete solver counts the number of solutions, subsuming SAT as a special case. Specifically, if we can count the number of solutions, we can trivially determine whether a solution exists.

We target this direction because #P-complete problems have stronger mathematical foundations and a richer history of surprising polynomial-time results, drawing on algebraic graph theory, spectral graph theory, and algebraic topology rather than heuristic search.

Our approach builds specifically on Turing Award laureate Leslie Valiant's foundational work in #P-completeness. Two landmark results in this area, Kirchhoff's Matrix Tree Theorem and the FKT algorithm for counting matchings in planar graphs, both rely on matrix determinants and demonstrate that exact polynomial-time counting is achievable for important problem families.

Our Contributions

We extend this class of determinant-based algorithms by deriving faster exact algorithms for several sub-families of graphs. Our results apply to problems including:

- Counting solutions to Monotone-2SAT logic formulas (equivalent to counting vertex covers and independent sets)
- Counting matchings and perfect matchings
- Extensions to broader graph families beyond the planar case

These breakthroughs draw on techniques at the intersection of linear algebra, algebraic combinatorics, algebraic graph theory, spectral graph theory, and algebraic topology.