Traceoid

Machine learning done right

Adam Nemecek, 2024

Problem: Machine learning is hitting a wall

- Machine learning is slow and expensive
- Developing models is painful
- Models are black boxes, lack explainability
- MLOps is complicated

Our goal

- Traceoid will make machine learning
  - **Cheap**: training is too compute intensive
  - **Unified**: based on a single concept
  - **Understandable**: custom solutions and architectures are viable
  - **Predicability**: model behavior is understandable & explainable
  - **Integrated**: most problems are implementable using built-in tools

Time is right

- Current machine learning approaches are reaching limits
- Industry solves problems by increasing compute; no one is revisiting foundations, despite the existence of rich theory
- What comes after convnets, LLMs and diffusion?

Machine learning is expensive

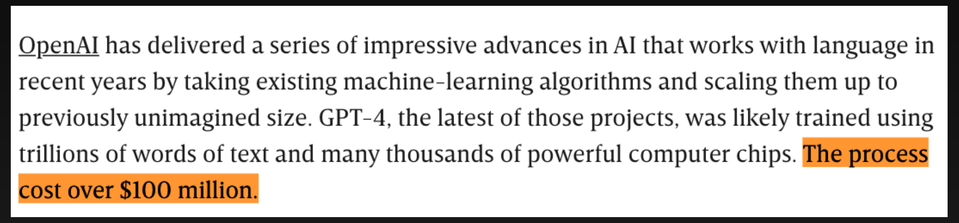
- Startups raise billions...
  - Anthropic raised $4B

AI Race Gets Hotter As Amazon To Invest Up To $4B In Startup Anthropic

Chris Metinko September 25, 2023

https://news.crunchbase.com/ai/google-anthropic-openai-funding-wars/

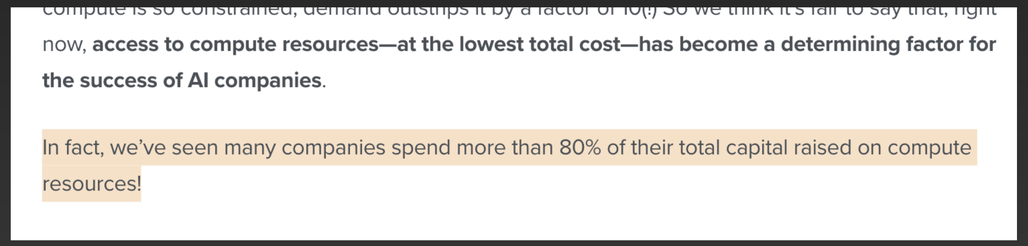
- ...to spend it on compute
  - ChatGPT-4 training cost over $100M



OpenAI has delivered a series of impressive advances in AI that works with language in recent years by taking existing machine-learning algorithms and scaling them up to previously unimagined size. GPT-4, the latest of those projects, was likely trained using trillions of words of text and many thousands of powerful computer chips. The process cost over $100 million.

https://www.wired.com/story/openai-ceo-sam-altman-the-age-of-giant-ai-models-is-already-over/

- Startups spend up to 80% of raised capital on compute



compute is so constrained, demand outstrips it by a factor of 10(!) So we think it's fair to say that, right now, **access to compute resources—at the lowest total cost—has become a determining factor for the success of AI companies**.

In fact, we've seen many companies spend more than 80% of their total capital raised on compute resources!

https://a16z.com/navigating-the-high-cost-of-ai-compute/

Developing models is painful

- Different problems require different architectures
  - Currently, each architecture is treated separately
- This limits the applicability of machine learning
  - What about problems for which it is unclear which architecture to use or for which architectures are nonexistent?
- Developing new architectures and writing custom GPU kernels is nontrivial

Models are black boxes

- Currently, models lack:
  - **Interpretability**
    - Can we understand the behavior of the models?
  - **Debuggability**
    - Can we fix models if they go wrong?
  - **Verifiability**
    - Can we guarantee behavior of the models?
- As a result, models cannot be deployed in dependable contexts

MLOps is complicated

- No “deploy and forget” solution
  - Scaling is not automatic
  - Monitoring of models to prevent degradation is not automatic
- Managing training pipelines is complicated

Solution: Traceoid

- ML platform (framework + hosting) based on these insights
  - **Unified**
    - All machine learning architectures, including convnets, diffusion models, transformers, are modeled as convolution of a Hopf algebra
  - **Geometric**
    - Enables interpretability and verifiability
  - **Integrated**
    - Traceoid provides all functionality the developer might need

Insights: All models are convolutions

- Convolution
- Transformers

time series etc.). We present here a unified framework which aims at capturing the essence of these diverse models, al
a systematic analysis of their properties and their mutual enrichment. We also show that attention models naturally fi
same framework: attention is convolution in which the structure itself is adaptive, and learnt, instead of being given a

Andreoli 2020: Convolution, attention and structure embedding (https://arxiv.org/abs/1905.01289)

- Diffusion

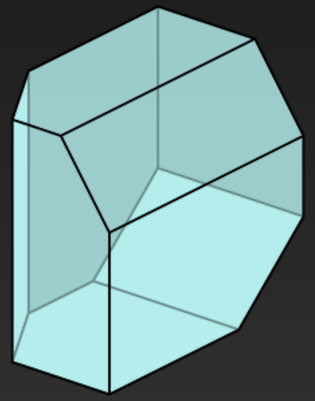
It is well-known that convolutions, differential operators and diffusion processes are interconnected subjects: the ordinary convolution commutes with the Laplacian, and the law of Brownian motion has a convolution semigroup property with respect to the ordinary convolution. If we seek to generalize this useful connection so as to cover other differential operators and diffusion processes, we are naturally led to the notion of a convolution-like operator—i.e. a bilinear operator with respect to which

Sousa 2022: Convolution-like Structures, Differential Operators and Diffusion Processes (https://link.springer.com/book/10.1007/978-3-031-05296-5)

Insights: ML is geometry discovery

- Training is about finding decision boundaries (symmetries)
- These form geometric structures called polytopes (observed in the wild)
- Traceoid makes these geometries explicit

Polytope



https://en.wikipedia.org/wiki/Simple_polytope

Emergence of polytopes has been observed in the wild

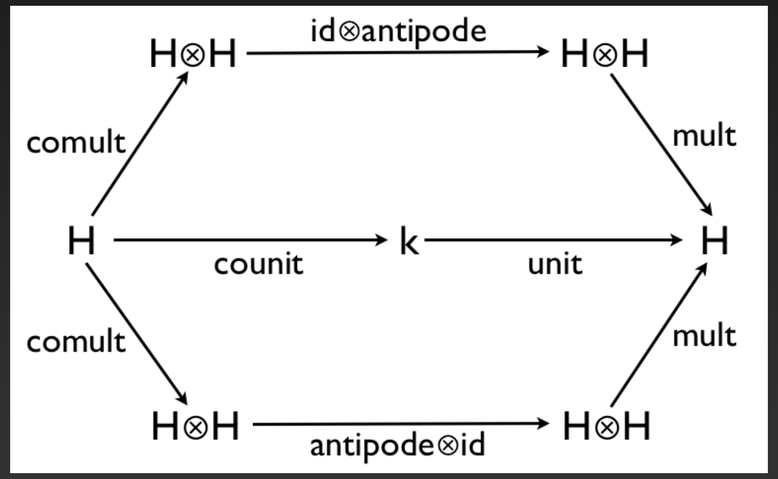
> There are several reasons why these models seem like a particularly interesting case for studying learning dynamics. Firstly, unlike most neural networks, the fully trained models converge to a simple but non-trivial structure that rhymes with an emerging thread of evidence that neural network learning dynamics might have geometric weight structure that we can understand. One might hope that understanding the final structure would make it easier for us to understand the evolution over training. **Secondly, superposition hints at surprisingly discrete structure (regular polytopes of all things!).** We'll find that the underlying learning dynamics are also surprisingly discrete, continuing an emerging trend of evidence that neural network learning might be less continuous than it seems. Finally, since

https://transformer-circuits.pub/2022/toy_model/index.html

Insights: Hopf algebra

- Hopf algebra convolution allows for detection of these geometries
- Hopf algebra is a rich algebraic structure, generalization of tensors
- Hopf algebras were there all along, RNN -> Ising model -> renormalization -> Hopf algebra

Hopf algebra



http://haskellformaths.blogspot.com/2012/03/what-is-hopf-algebra.html

Competitors

- Traceoid: framework and a hosting platform
- **Frameworks**:
  - PyTorch, Mojo: effort required to overcome shortcomings of Python, no predictability, architecture questions remain
- **Hosting**:
  - AWS, Google Cloud: undifferentiated, not integrated
- **Framework + hosting**:
  - Julia & JuliaHub: Julia is an unsuitable language for large scale projects, no predictability, fragmented ecosystem, architecture questions remain

Differentiators

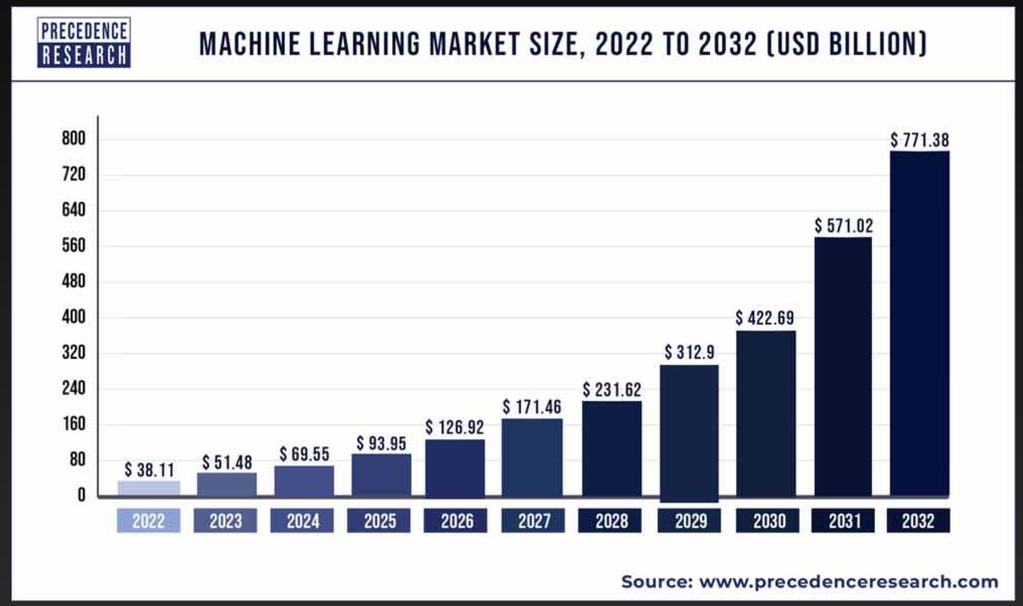
- **Neural architecture search**
  - Both weights and architectures are discovered during training
- **Explainability, verifiability**
  - Model behavior is understandable & can be guaranteed statically
- **Cheaper training and inference**
  - More structure is extracted from data => both training and inference cheaper
- **Easy deployment**
  - Develop & deploy ML based applications the same way you'd deploy apps on Heroku

Business

- Open-core product like Databricks
- Main platform is open source, enterprise offering is paid, add-on libraries/ functionalities

Market

- $ 50B in 2023, projected $750B in 10 years



https://www.precedenceresearch.com/machine-learning-market

- Framework/hosting combination can capture 30% of the market

How will the money be spent?

- CPU based prototype in Rust
- Benchmark against a comparable CPU based approach
- Eventually, all will run on GPU
- Raising $1.5M, most of which will be spend on salaries
  - 4 devs, ~$120K average salary

Team

- **Adam Nemecek (Founder)**
  - Harvard CS
  - Implemented macOS recording for loom.com (~$1B Atlassian acquisition)
  - Worked at MSFT

- **Ammar Husain**
  - Berkeley PhD in Topological Quantum Computation
  - Quantum compiler developer

- **Iago Leal de Freitas**
  - M.Sc. from URFJ, top 🇧🇷 university
  - Programmer / mathematician for an energy consulting company

Why us?

- Non-standard approach, infinite upside
- Familiarity with the relevant theories
- Not a research project - theory is all there, implementations are non-existent or fragmented *in code*