MINI-WORKSHOP ON MACHINE LEARNING AND ALGEBRAIC GEOMETRY

NOTES BY MAKSYM ZUBKOV

Contents

1. Speaker 1: Tropical Algebraic Approaches to Deep Learning

Ruriko Yoshida

This talk was meant as an introduction, overview, and discussion of ideas and open problems in tropical algebraic geometry for deep learning.

First, let us start with a definition of a feedforward neural network. A feedforward neural network is a function

$$f : \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$$

Date: June 4, 2026.

given by a composition of affine maps and activation functions:

$$f = \sigma_L \circ A_L \circ \cdots \circ \sigma_1 \circ A_1.$$

Here the affine maps contain the weights and biases, and the activation functions are applied coordinatewise.

Problem 1.1. We wish to characterize the function f.

Why do we care about understanding the geometry associated with a function f, i.e. with a neural network?

Remark 1.2. Describing f gives:

(1) better understanding of decision boundaries;
(2) possible ways to speed up the optimization process;
(3) possible ways to improve the performance and robustness of deep learning models.

We have different activation functions:

sigmoid, tanh, ReLU, ELU, Maxout, Leaky ReLU, ...

Depending on the activation function, the surface or piecewise-linear structure defined by f changes.

The neuromanifold is the set of functions defined by neural networks with parameters varying over a fixed architecture.

Theorem 1.3 (Zhang–Naitzat–Lim, 2018)**.** *Feedforward ReLU neural networks are equivalent to tropical rational maps.*

Remark 1.4. In the work of Brandenburg, Loho, and Montúfar, a binary classifier is studied as the sign of a tropical rational function, i.e. as the difference of two convex piecewise-linear functions. They study subdivisions of the parameter space and the combinatorics of decision boundaries.

Remark 1.5. Another related work by Balakin, Cox, Loho, and Sturmfels studies Maxout polytopes. These are polytopes arising from feedforward neural networks with maxout activation functions and non-negative weights after the first layer.

1.1. **Why tropical algebraic geometry?** First, tropical algebra is related to optimization. Under suitable assumptions, optimization problems can be expressed using tropical polynomials and tropical powers.

There is also a connection to combinatorial optimization. Many dynamic programming algorithms can be written using tropical operations.

For shortest path problems, Floyd's algorithm for shortest paths in a weighted directed graph can be viewed through tropical matrix powers.

For assignment problems, the Hungarian method can be interpreted as a tropical analogue of Gaussian elimination for computing a tropical determinant.

1.2. **Tropical basics.** The max-plus tropical semiring is

$$\mathbb{T} = (\mathbb{R} \cup \{-\infty\}, \oplus, \odot),$$

where

$$a \oplus b = \max(a, b), \qquad a \odot b = a + b.$$

The tropical projective space is obtained by identifying vectors that differ by adding the same scalar to all coordinates.

The tropical projective torus is

$$\mathbb{R}^d / \mathbb{R}\mathbf{1},$$

where

$$\mathbf{1} = (1, \ldots, 1).$$

Remark 1.6.

$$\mathbb{R}^d/\mathbb{R}\mathbf{1} \cong \mathbb{R}^{d-1}.$$

The tropical distance between two points $x, y \in \mathbb{R}^d/\mathbb{R}\mathbf{1}$ is

$$d_{\mathrm{tr}}(x, y) = \max_i(x_i - y_i) - \min_i(x_i - y_i).$$

A tropical power is defined by

$$x^{\odot k} = kx.$$

A tropical monomial has the form

$$c \odot x_1^{\odot a_1} \odot \cdots \odot x_d^{\odot a_d} = c + a_1 x_1 + \cdots + a_d x_d.$$

A tropical polynomial is a tropical sum of tropical monomials:

$$f(x) = \bigoplus_j \left(c_j \odot x_1^{\odot a_{j1}} \odot \cdots \odot x_d^{\odot a_{jd}} \right).$$

Equivalently,

$$f(x) = \max_j \left(c_j + a_{j1} x_1 + \cdots + a_{jd} x_d \right).$$

For example,

$$f(x) = x^{\odot 3} \oplus 2 \odot x^{\odot 2} \oplus 2$$

means

$$f(x) = \max(3x, 2x + 2, 2).$$

1.3. **Tropical neural networks.** A tropical neural network is a neural network using tropical operations or tropical geometry in its architecture.

One construction is to use a tropical embedding layer first, followed by classical neural network layers.

A tropical embedding layer takes

$$x \in \mathbb{R}^d/\mathbb{R}\mathbf{1}$$

and maps it to Euclidean coordinates using tropical distances. In one common form, the output of the j-th neuron is

$$z_j = \max_i(x_i + w_{ji}) - \min_i(x_i + w_{ji}).$$

This embeds points from the tropical projective torus into a Euclidean space, where standard neural network layers can then be applied.

1.4. **Tropical convolutional neural networks.** The talk also discussed tropical convolutional neural networks. The idea is to exploit the fact that many neural networks with piecewise-linear activations already have tropical or polyhedral structure.

Instead of only analyzing neural networks after the fact using tropical geometry, one can build tropical layers directly into the architecture.

1.5. **Tropical attention.** The talk also mentioned tropical attention mechanisms. Attention mechanisms are central in modern AI models, especially transformers and reasoning models.

The tropical idea is to replace or reinterpret parts of attention using max-plus operations, so that the resulting mechanism preserves sharper polyhedral and piecewise-linear structure.

This leads to open questions about whether tropical attention can improve robustness, interpretability, or out-of-distribution generalization.

1.6. **Takeaway.** The main takeaway is that tropical algebraic geometry gives a natural language for studying deep learning models with piecewise-linear structure.

The rough dictionary is:

$$\text{ReLU / Maxout neural networks} \longleftrightarrow \text{tropical rational functions / tropical polynomials} \longleftrightarrow \text{polyhedral geometry.}$$

For my own interests, the most relevant point is that tropical geometry can help describe decision boundaries, parameter spaces, and neuromanifolds of neural networks in a combinatorial and polyhedral way.

2. Speaker 2: Flow-Based Extremal Mathematical Structure Discovery

Baran Hashemi, MPI-MIS Leipzig

The talk started with the AI for mathematics landscape. One point mentioned was the Leiden Declaration on Artificial Intelligence and Mathematics. The first recommendation is to disclose the tools used, including large language models, machine learning systems, proof assistants, and other mathematical software.

2.1. Extremal mathematics.

What is extremal mathematics? The word extremal comes from the nature of the problems this field deals with:

> If a collection of finite objects satisfies certain restrictions, how large or how small can it be?

One example is Ramsey-type problems. Once the system size N crosses a critical threshold T, a specific structural property P becomes unavoidable.

Another type is Turán-type problems. Instead of forcing a property, here we forbid one under a given local restriction R.

Problem 2.1 (Discrete version)**.** Given a real-valued graph parameter P and a class of graphs $\mathcal{C}$, how large or small can P be for an element of $\mathcal{C}$?

Problem 2.2 (Continuous version)**.** Given a function

$$f : \mathbb{R}^n \to \mathbb{R}$$

and a subset

$$S \subset \mathbb{R}^n,$$

how large or small can $f(x)$ be for an element $x \in S$?

2.2. Discovering extremal configurations.

The general problem is to find rare, complex geometric or combinatorial solutions in vast, non-convex spaces with hard constraints.

Some challenges:

- there is no atlas of the search space;
- the spaces are huge and non-convex;
- many problems have hard geometric or algebraic constraints;
- brute-force search is usually intractable.

Problems discussed include:

- sphere packing in unit hypercubes, including dimensions 3 and 12;
- circle packing problems;
- the Heilbronn problem;
- the star discrepancy problem.

The art is finding the extreme: the densest, the most uniform, or the hardest to avoid.

2.3. **FlowBoost.** The proposed solution is simulation-based optimization.

The idea is to use evaluators as optimization oracles and learn amortized samplers that concentrate on high-reward solutions.

Ingredients:

- a configuration space $\mathcal{X}$;
- an objective or reward function
$$J : \mathcal{X} \to \mathbb{R};$$
- the goal
$$x^* = \arg\max_{x \in \mathcal{X}} J(x).$$

The framework introduced in the paper is called FlowBoost. It combines:

- geometry-aware conditional flow matching;
- reward-guided policy optimization;
- stochastic local search.

The key point is that the sampler is not only trained to imitate good examples. It receives reward feedback and is pushed toward better configurations while respecting geometric feasibility.

2.4. **Takeaway.** The main takeaway is that extremal mathematical discovery can be viewed as a search problem over difficult configuration spaces.

The pipeline is roughly:

$$\text{configuration space} \longrightarrow \text{objective/evaluator} \longrightarrow \text{learned sampler} \longrightarrow \text{high-reward extremal structures.}$$

For my own understanding, the most relevant point is that this gives a way to use machine learning not just for prediction, but for discovering rare mathematical objects under hard constraints.

3. Speaker 3: Aspects of Optimal Transport

Samir Bhatt, University of Copenhagen and Imperial College London
Neural optimal transport on Riemannian manifolds

Samir Bhatt is a professor of machine learning, statistics, and public health. The talk was about recent work on neural optimal transport on Riemannian manifolds, especially Riemannian Neural Optimal Transport and its entropic version.

3.1. **History of optimal transport.** A rough historical path of optimal transport is:

$$\text{Monge} \longrightarrow \text{Kantorovich} \longrightarrow \text{Brenier} \longrightarrow \text{McCann} \longrightarrow \text{Cuturi} \longrightarrow \text{Neural OT} \longrightarrow \text{Riemannian Neural OT.}$$

Monge's formulation asks for a deterministic transport map. Kantorovich relaxed the problem to a linear programming problem over couplings. Brenier gave the classical Euclidean structure theorem for quadratic cost. McCann generalized optimal transport maps to Riemannian manifolds. Cuturi introduced entropic regularization as a computationally efficient approach to OT.

The recent work discussed in the talk combines:

$$\text{entropic OT} + \text{neural parameterizations} + \text{Riemannian geometry.}$$

3.2. **The shipping problem.** The classical intuition is the shipping problem.

Factories are distributed according to a source measure

$$\mu,$$

and stores are distributed according to a target measure

$$\nu.$$

Now suppose the factories and stores lie on a Riemannian manifold M. The shipping cost is a function

$$c : M \times M \to \mathbb{R}.$$

A common choice is the squared geodesic distance:

$$c(x, y) = \frac{1}{2} d_M(x, y)^2.$$

3.3. **The Kantorovich primal.** A shipping plan is a coupling

$$\pi(x, y),$$

that is, a joint distribution over factory–store pairs.

The marginal constraints say:

$$\pi_X = \mu, \qquad \pi_Y = \nu.$$

In words, supply equals demand.

The Kantorovich primal problem is

$$\inf_{\pi \in \Pi(\mu,\nu)} \int_{M \times M} c(x, y)\, d\pi(x, y),$$

where

$$\Pi(\mu, \nu)$$

is the set of all couplings with marginals μ and ν.

3.4. **The dual: pricing the routes.** Instead of searching over transport plans π, the dual problem searches over prices.

We introduce:

- a factory pick-up fee $\phi(x)$;
- a store delivery fee $\psi(y)$.

The participation constraint is:

$$\phi(x) + \psi(y) \le c(x, y).$$

The dual problem is

$$\sup_{\phi,\psi} \left\{ \int_M \phi(x)\, d\mu(x) + \int_M \psi(y)\, d\nu(y) \right\}$$

subject to

$$\phi(x) + \psi(y) \le c(x, y).$$

Strong duality says that the value of this dual problem equals the value of the primal problem.

3.5. **From strong duality to the c-transform.** The dual has two potentials tied together by one constraint:

$$\phi(x) + \psi(y) \le c(x, y).$$

Fix the store price ψ. Then every feasible pick-up fee must satisfy

$$\phi(x) \le c(x, y) - \psi(y)$$

for all $y \in M$.

So the largest admissible choice is the tightest one:

$$\phi(x) = \inf_{y \in M} \big(c(x, y) - \psi(y) \big).$$

This is the c-transform.

3.6. **From potentials to maps: Euclidean case.** In $\mathbb{R}^d$, with quadratic cost

$$c(x, y) = \frac{1}{2}\|x - y\|^2,$$

optimal transport maps can be described using potentials.

In the Euclidean Brenier setting, the optimal map has the form

$$T(x) = x - \nabla\phi(x),$$

up to the convention used for the Kantorovich potential.

Equivalently, if the minimizer in the c-transform is y^*, then first-order optimality gives a gradient relation between x, y^*, and the potential.

3.7. **McCann's map: Riemannian generalization.** On a Riemannian manifold, the phrase "step from x in the direction v" is expressed using the exponential map:

$$\exp_x(v).$$

Thus, the Riemannian analogue of the Euclidean transport map is

$$T(x) = \exp_x\big(-\nabla\phi(x)\big).$$

This is the McCann map.

3.8. **The curse of dimensionality barrier.** A key problem is the curse of dimensionality.

On compact Riemannian manifolds, many discretization-based OT methods produce maps whose pushforward

$$T_{\#}\mu$$

is supported on at most m points.

This includes point-cloud, mesh-based, and related discrete approximation methods.

The recent RNOT work proves a lower bound showing that such discrete methods necessarily suffer from a dimension-dependent approximation barrier. In the notation of the talk, for a compact p-dimensional manifold M with

$$\mu, \nu \ll \mathrm{vol}_M,$$

one has a lower bound of the form

$$\inf \mathrm{RMSE}(T, T_*) \geq Cm^{-1/p}.$$

Thus, to achieve accuracy δ, one needs roughly

$$m \gtrsim \delta^{-p}.$$

This is exponential in the dimension p.

3.9. **RNOT: universality and breaking the curse of dimensionality.** RNOT stands for Riemannian Neural Optimal Transport.

The idea is to parameterize a continuous c-concave potential and define the transport map by

$$T(x) = \exp_x(-\nabla\phi(x)).$$

Unlike discretization-based methods, this gives a map with unrestricted output support.

The RNOT paper proves that, under suitable regularity assumptions, such neural parameterizations can approximate Riemannian OT maps with sub-exponential complexity in the dimension.

In short:

$$T_k \to T_*$$

for a sequence of neural RNOT maps T_k, with better dimension scaling than discrete methods.

3.10. **Curse of dimensionality in practice: scaling on S^p.** The experiments include scaling on spheres

$$S^p.$$

The point is to compare discretization-based methods with RNOT-type methods as the dimension p grows.

The expectation is that discrete methods degrade quickly with dimension, while the neural Riemannian parameterization scales better.

3.11. **Why entropic? Geometry and computation.** The entropic version combines intrinsic entropic optimal transport with amortized out-of-sample evaluation on Riemannian manifolds.

Entropic regularization is useful computationally because it smooths the OT problem and connects to Sinkhorn-style algorithms.

The entropic RNOT framework learns a target-side Schrödinger potential and recovers the corresponding Gibbs coupling. This produces intrinsic transport surrogates on the manifold.

3.12. **Clean geometry: Cartan–Hadamard manifolds.** A particularly clean geometric setting is given by Cartan–Hadamard manifolds.

These are manifolds that are:

- complete;
- simply connected;
- non-positively curved.

In this setting, geodesics are unique, and geometric averages such as Fréchet means behave more stably.

This makes the geometry especially suitable for defining intrinsic transport surrogates.

3.13. **Takeaway.** The main takeaway is that neural optimal transport can be extended from Euclidean spaces to Riemannian manifolds by using the geometry of the manifold directly.

The key dictionary is:

$$\text{Euclidean step} \longrightarrow \text{exponential map},$$
$$\text{convex potential} \longrightarrow c\text{-concave potential},$$
$$\text{discrete OT approximation} \longrightarrow \text{continuous neural Riemannian map}.$$

For my own understanding, the most relevant point is that RNOT avoids discretizing the manifold and therefore can escape the curse of dimensionality barrier faced by many mesh-based or point-cloud OT methods.

Notes by Maksym Zubkov