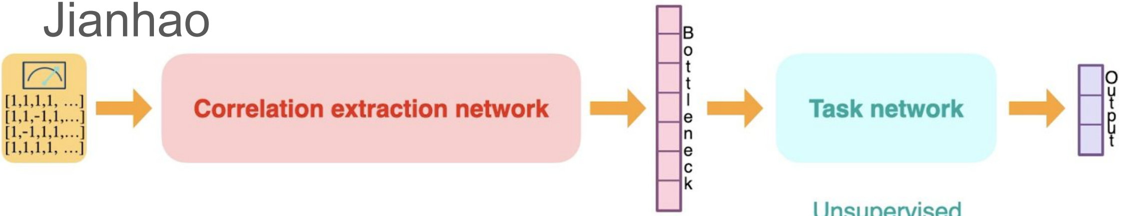
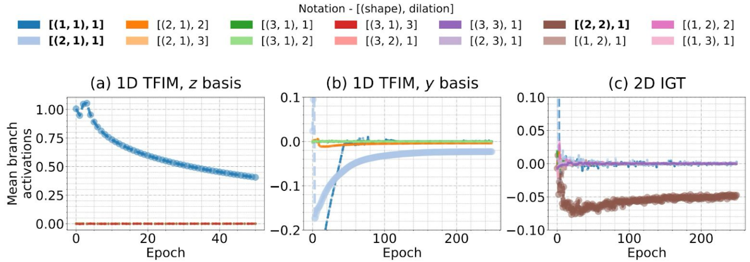
Add your 1-slide paper summary here:

https://tinyurl.com/805-sept-10

1-slide paper summaries

Jianhao

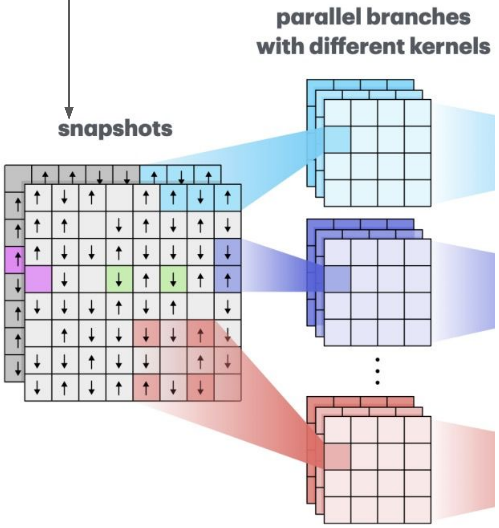




Unsupervised detection of phase transitions via prediction based method

Decomposing the input snapshot to multiple branches of different correlations

i.e. the loss function is simply the MSE between predicted g and real g!



Task In this work, TetrisCNN is used in combination with the prediction-based method [43, 9]. We train TetrisCNN using PyTorch [47] by minimizing the mean squared error between the network output and the tuning parameter. For 1D TFIM, the tuning parameter is the transverse field value, g; for 2D IGT, the tuning parameter is an inverse temperature, β. The training hyperparameters are in Tab. 1.

It means: for those low contributed terms, their contribution is less than the additional cost in this loss term due to their existence!

activations averaged across channels and physical system size via the following loss term:

$$L_{\text{bottle}} = \lambda_k \sum_k |a_k| \,. \tag{10}$$

A machine learning approach to duality in statistical physics arXiv:2411.04838

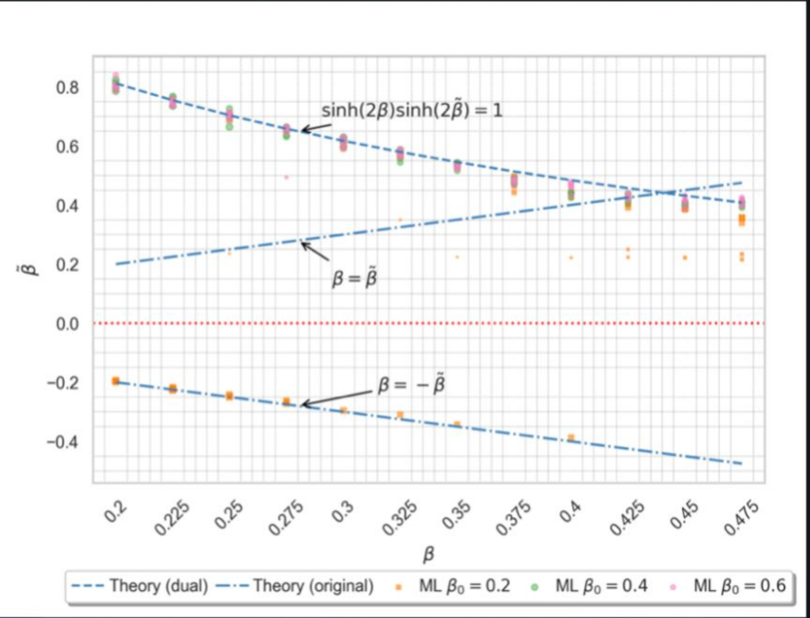
Gupta, Ferrari and Iqbal

$$H[\beta,\sigma] = -\beta\sum_{\langle ij\rangle}\sigma_i\sigma_j \leftrightarrow \tilde{H}[\tilde{\beta},\tilde{\sigma}] = -\tilde{\beta}\sum_{\langle ij\rangle}\tilde{\sigma}_i\tilde{\sigma}_j$$

$$\sinh 2\beta \sinh 2\tilde{\beta} = 1, \quad O_{ij} = \sigma_i\sigma_j \sim e^{-2\tilde{\beta}\tilde{\sigma}_i\tilde{\sigma}_j} = \tilde{O}_{ij}$$

Duality in 2d Ising model

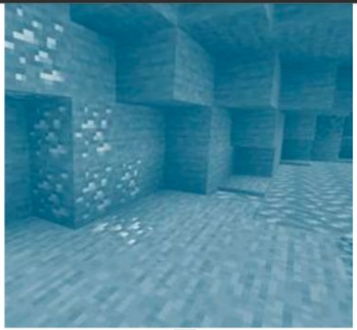
Learn these



Link variables $O_{ij} = \sigma_i\sigma_j$ are the input in the neural network

Loss function $\sim \sum (\langle O_{ij}\rangle_H - \langle \tilde{O}_{ij}\rangle_{\tilde{H}})^2$ + squared error in various moments

Posterior sampling
$P(Guide|D)$



Likelihood
$P(D|Guide)$

$P_{Now}(Guide)$

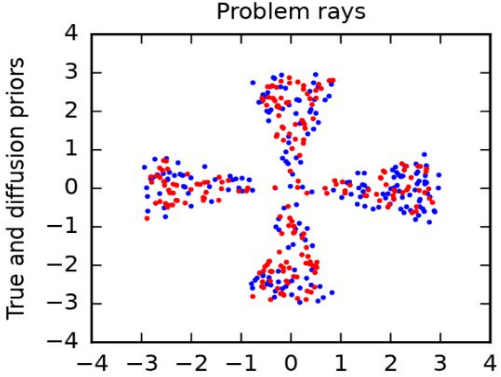
YouTube

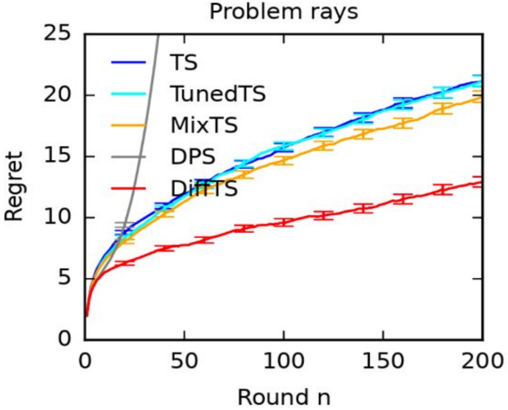
Meta

Diffusion prior
$P_{day3}(Guide)$

Tom's diary

Tom





- Diffusion model works well for multimodal prior
- At a cost of more expensive computation
- See 2410.03919 for more details

Fine-tuning Foundation Models for Molecular Dynamics: A Data-Efficient Approach with Random Features (252)

Problem

- Foundation models (MACE, OrbNet) pretrained on large QM datasets
- Downstream tasks often have very limited ab initio data
- Full fine-tuning is costly and unstable

Method

- Map pretrained embeddings → fixed **Random Feature (RF)** basis
- Learn only a linear readout (**convex optimization**)
- Efficient + stable under small-data regime

Results

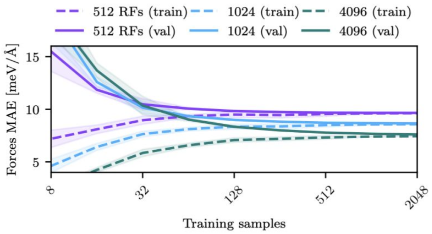
- RF fine-tuning converges faster and more robust than full model
- Superior performance in low-data regime

Table 1: Forces accuracy

Model	Forces MAE
MACE-MP0 [9] (zero-shot)	93.15 meV/Å
FLARE [24] (from scratch)	8.82 meV/Å
franken (fine-tuning)	**7.61 meV/Å**

Takeaway

- RF: engineering advancement that improves usability of chemical FMs
- Still purely data-driven; physics priors remain limited



Train time on `4xA100 GPUs`

# Samples	Train time [s]
8	5.8 ± 0.4
32	8.4 ± 0.5
128	16.6 ± 0.9
512	46.4 ± 0.9
2048	163.8 ± 0.6

Figure 1: **Sample complexity, forces prediction.** Training (dashed lines) and validation (solid lines) mean absolute errors corresponding to different numbers of RFs as a function of the training samples.

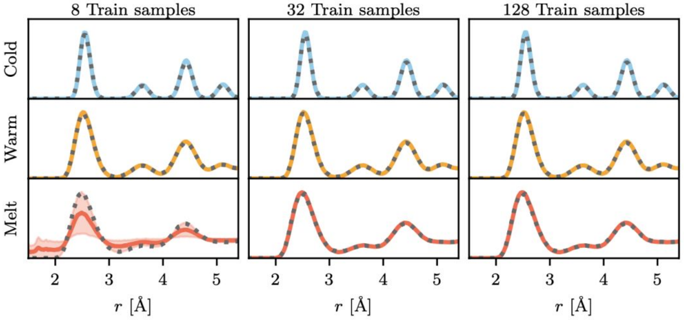


Figure 2: **Sample complexity, MD simulations**. Radial distribution functions generated from MD simulations with *franken* potentials at different temperatures (rows) and with 4096 RFs. Different columns correspond to different numbers of training samples. Each panel shows the mean (solid line) and standard deviation (shaded area) over 5 models trained on independent sub-splits, together with the reference calculated from the TM23 dataset (dotted line).

Interpreting Transformers for Jet Tagging

A. Wang[1], A. Gandrakota[2], J. Ngadiuba[2], V. Sahu[3], P. Bhatnaga[3], E. Koda[3], J. Duarte[3]

Background

- Jet tagging: Process of identifying which sort of particle originated a jet (collimated shower of particles).
- Particle transformer: attention-based transformer for jet tagging that uses particle-level attention [1].
- Particle inputs
  - Particle kinematics (4-vector)
  - Particle ID
  - Trajectory displacement (impact parameters)
- Interaction inputs

$$\Delta = \sqrt{(y_a - y_b)^2 + (\phi_a - \phi_b)^2},$$
$$k_{\mathrm{T}} = \min(p_{\mathrm{T},a}, p_{\mathrm{T},b})\Delta,$$
$$z = \min(p_{\mathrm{T},a}, p_{\mathrm{T},b})/(p_{\mathrm{T},a} + p_{\mathrm{T},b}),$$
$$m^2 = (E_a + E_b)^2 - \|\mathbf{p}_a + \mathbf{p}_b\|^2,$$

- Trained and evaluated using a co-introduced *JetClass* dataset. It has 10 classes (9 signal, 1 background).
- This paper [2] looks at the interpretability of ParT through attention scores.

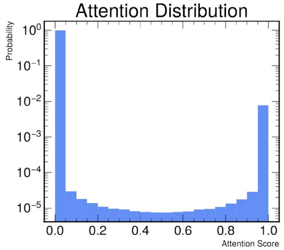
$$\mathrm{score}(a,b) = \frac{(x_a W^Q)(x_b W^K)^T}{\sqrt{d_k}} + U_{ab}$$

  - How important particle *b* is for updating particle *a*'s representation. I.e., it encodes whether the model sees a meaningful relation between them.
  - Tells us which particle-particle interactions are most relevant for classification.

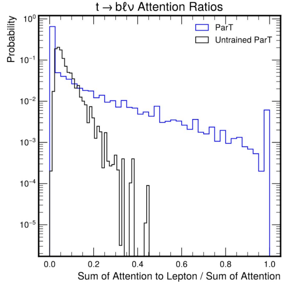
[1]University of Illinois Chicago, [2]Fermilab, [3]University of California San Diego

Results

- Distribution of attention scores
  - Attention scores mostly close to 0 or 1. It means that particles attend mostly to just one other particle.
  - Shows opportunity for improved computational efficiency of the attention mechanism of the transformer.



- Particle attention graphs
  - Jets reclustered (e.g. $t \to bl\nu$ into 2 subjets, $t \to bqq'$ to 3, $H \to 4q$ to 4, etc.)
  - $t \to bl\nu$: Tendency to attend to only leptons or to no leptons. Model recognizes importance of these in the classification.
  - Authors argue that is evidence that ParT is learning some underlying physics from jet data.



- Optimization study on the attention mechanism
  - Contrain the number of particles used in the attention mechanism, keeping only the k highest attention particles. The other particles are set to 0.

Max particles	Accuracy	AUC
1	0.770	0.9754
2	0.798	0.9795
3	0.814	0.9817
4	0.825	0.9831
6	0.838	0.9849
10	0.851	0.9865
20	0.859	0.9875
30	0.860	0.9877
128	0.861	0.9877

[1] arXiv:2202.03772
[2] arXiv:2412.03673v2

The paper a ... tatistical

mechanics ... nap. In

particular, t

$\langle O_\alpha ($

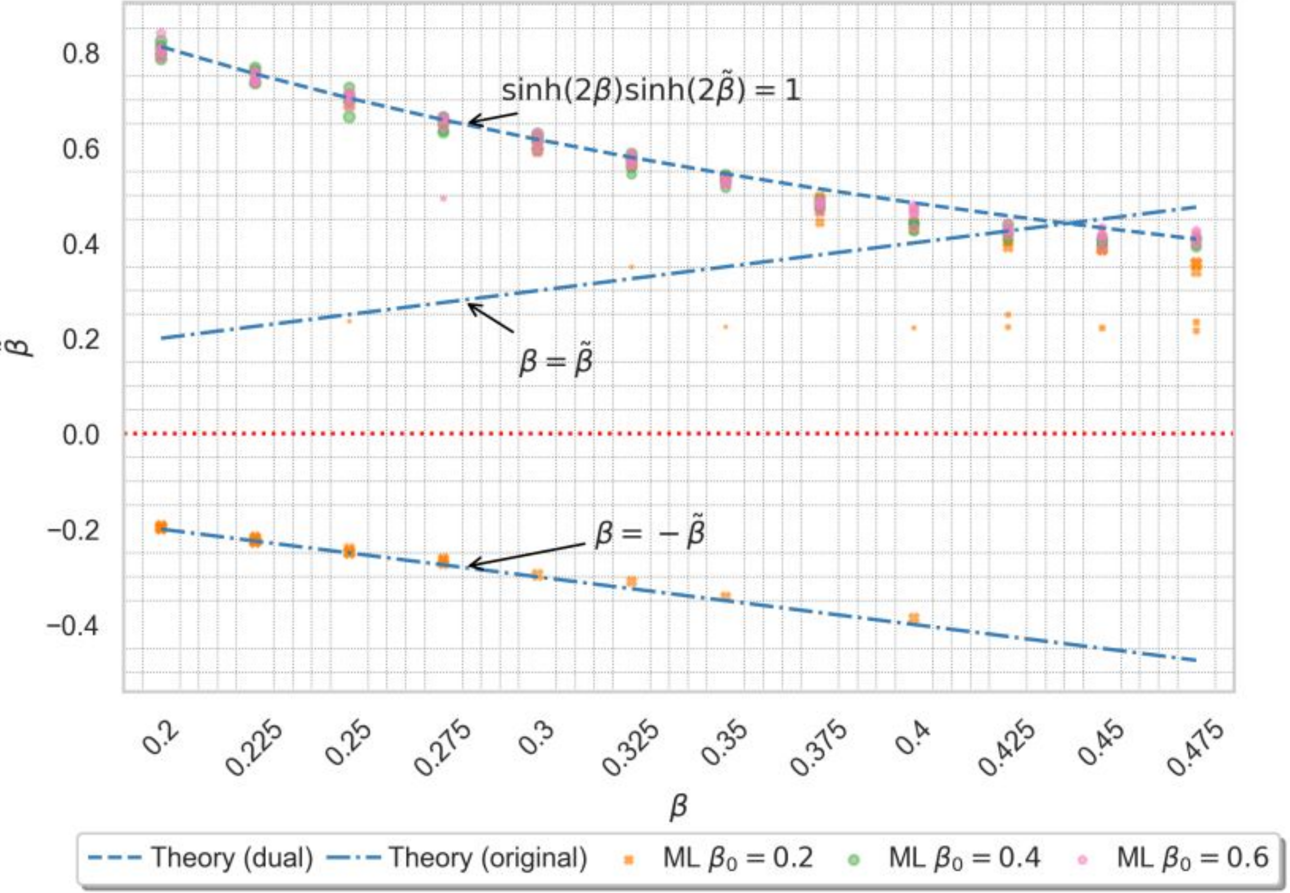
In 2d Ising ... $= e^{-2\tilde{\beta}\tilde{\sigma}_{i*}\tilde{\sigma}_{j*}}$

Consider G ... $\tilde{\sigma}_k\tilde{\sigma}_l\})$

The optimi ... $\langle\phi^a[G(\tilde{\sigma}_i)]\rangle_{\tilde{H}}$

$G^*, \tilde{H}^*$

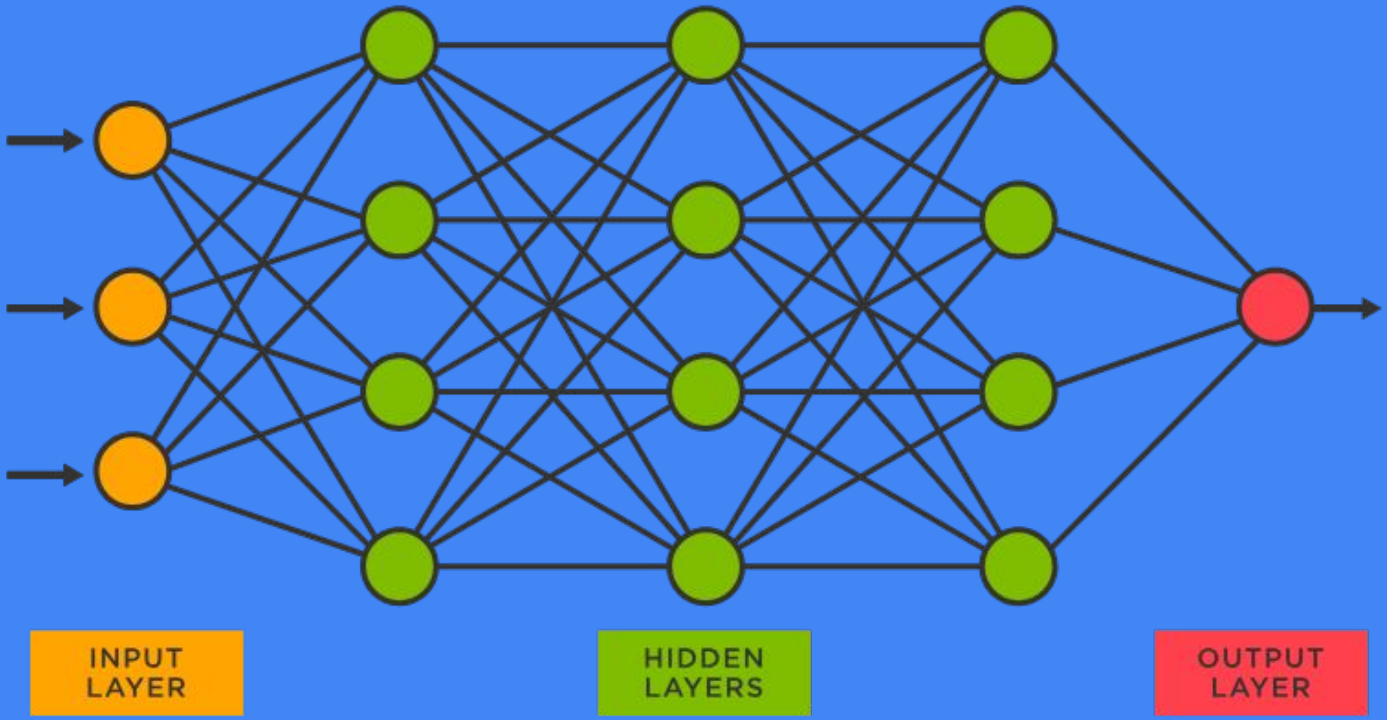
$G_\theta(\mathbf{f}_{\langle ij\rangle}) =$



The results

Components of a neural network

Components of a neural network



Components of a neural network

- **Trainable parameters:** weights & biases

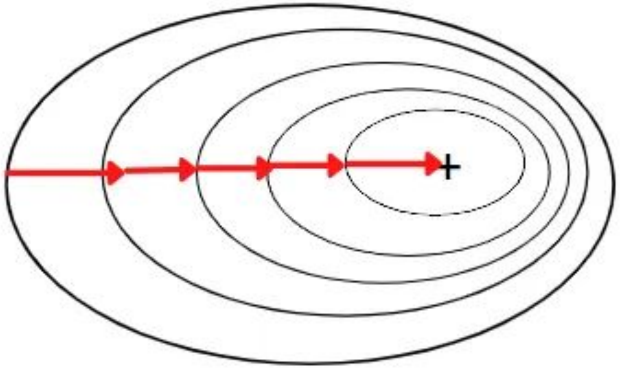
- **Data representations:** neurons & layers
  - Neurons & activations
  - Layers:
    - Input, hidden, output

- **Nonlinearities:** activation functions

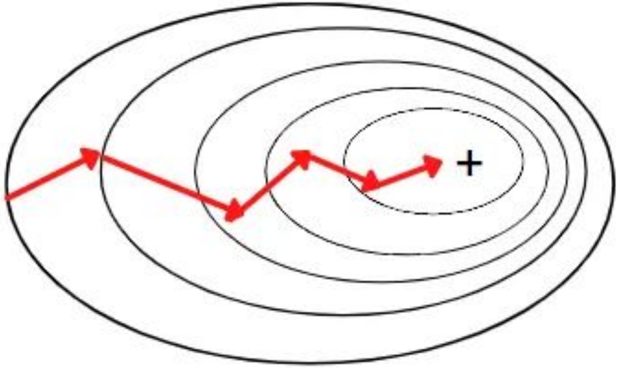
- **Organizing your training data:** (mini-)batches & DataLoaders

- **Training configuration:** loss function, # of epochs, optimizer

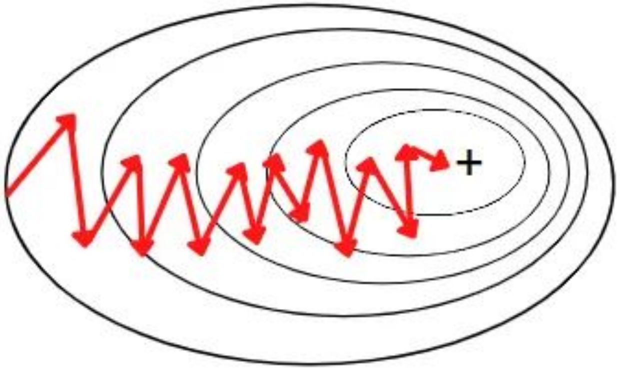
Batch Gradient Descent



Mini-Batch Gradient Descent



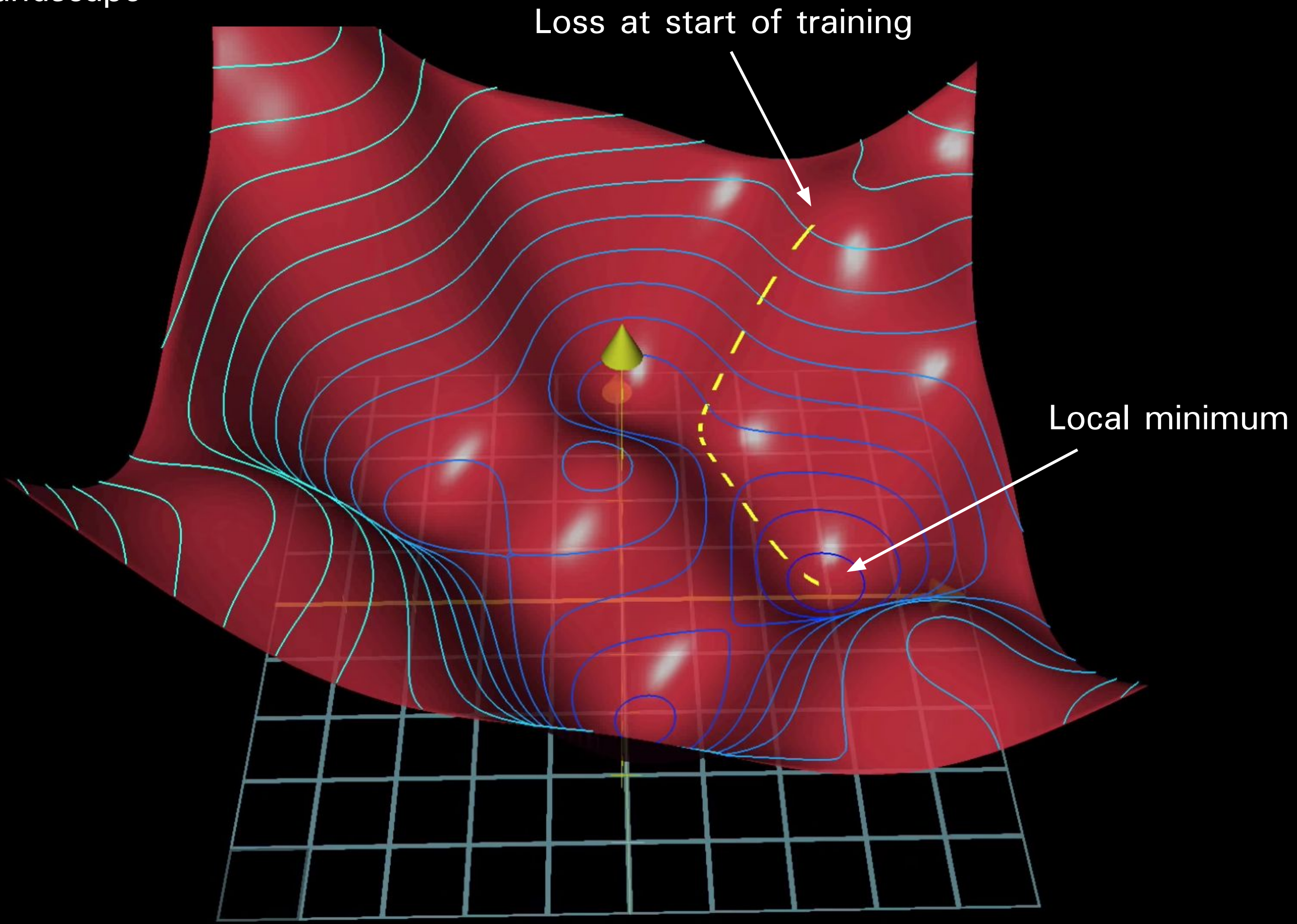
Stochastic Gradient Descent

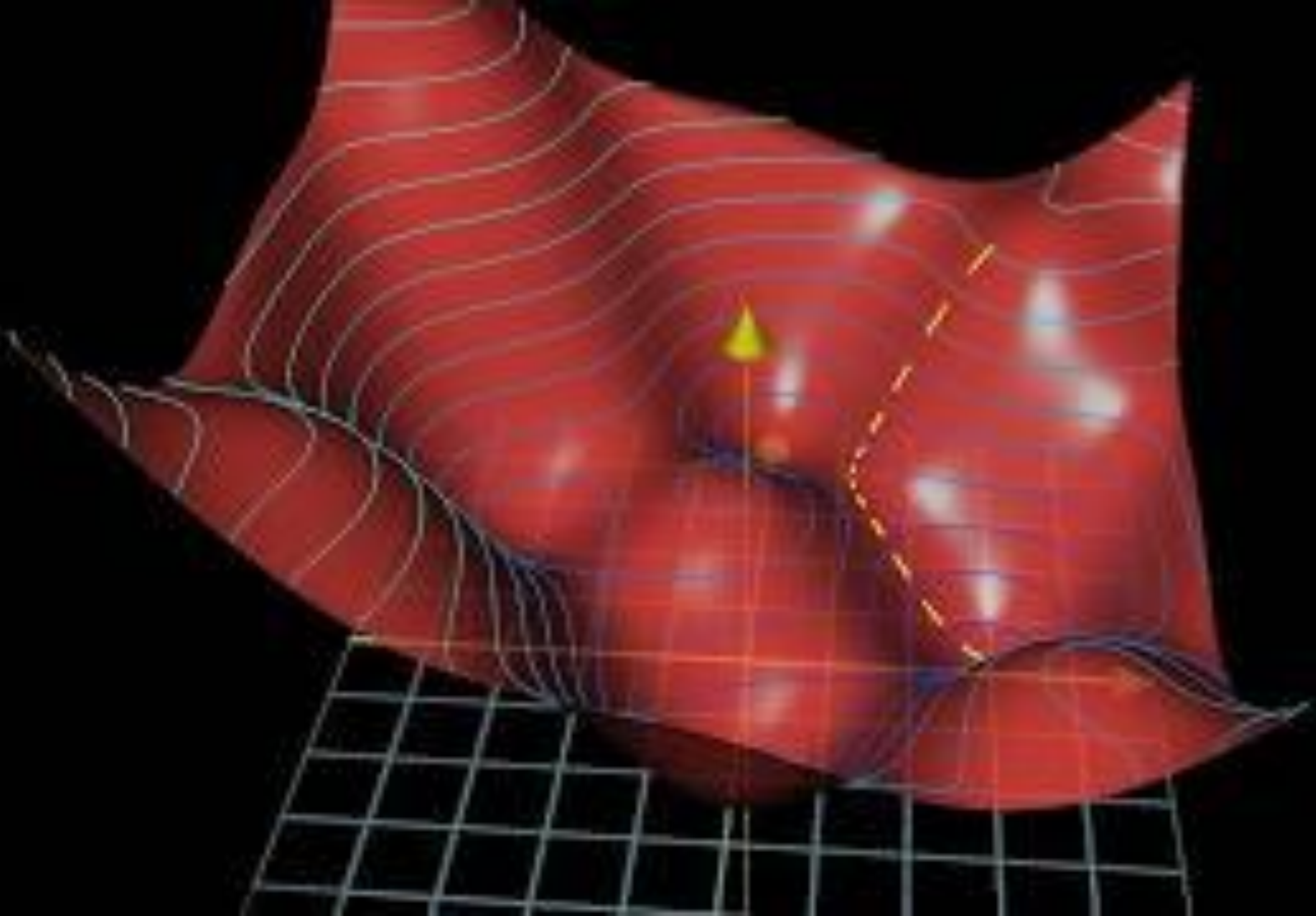


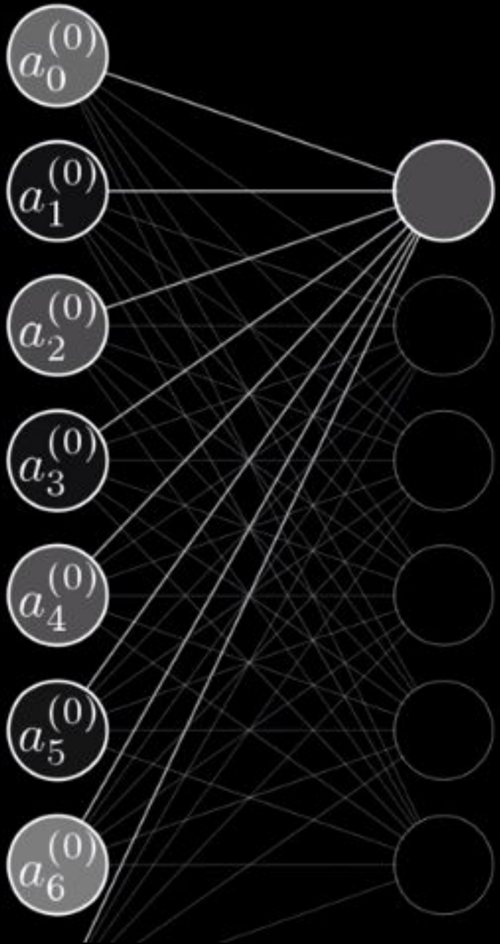
Common optimizers

- Gradient Descent
- Stochastic Gradient Descent
- RMSProp
- Adam
- Momentum
- Adagrad
- ...

Loss Landscape







Sigmoid ⟵ Activation function

$$a_0^{(1)} = \sigma\left(w_{0,0}\, a_0^{(0)} + w_{0,1}\, a_1^{(0)} + \cdots + w_{0,n}\, a_n^{(0)} + b_0\right)$$

Bias

Weights | Previous layer's activations | Biases

$$\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \cdots & w_{k,n} \end{bmatrix} \begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix}$$

Partner work: Training a Neural Network

- Navigate to the repository:
  - `https://github.com/mariel-pettee/phys_805_fall_2025`

- Work through Notebook 1
  - Don't just read in silence! Have both partners look at one screen and discuss as you go.

Coming up:

- Problem Set 1 will be posted this Friday and due 1 week later
- Then you'll have 2 weeks to work on Project 1
- And so on...