Machine Learning

Prof. Sergei Gleyzer

Lecture

PH451, PH551

April 15, 2025

Announcements

- **Outline and Demo due Thursday**
- **XC due next Tue, Apr 22**

Group Presentations

15 minutes total (12 minutes talk + 3 min Q/A)

Scoring Rubric:

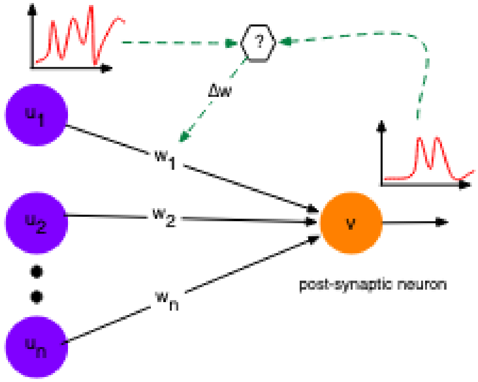
- Introduce the topic (10 pts)
  - What is the question you are trying to answer
  - Why is it important
  - Previous approaches/Why ML would help
- Machine Learning (10 pts)
  - Which technique and why
  - Model, training, hyperparameters
  - Results and conclusions
- Overall Impression (5pts) + Time Management (5pts)

Last Time: Energy Models

Key idea: minimize **energy** instead of **error**

- Compute **energy gradients**
- Learning = adjust **weights** to reduce energy

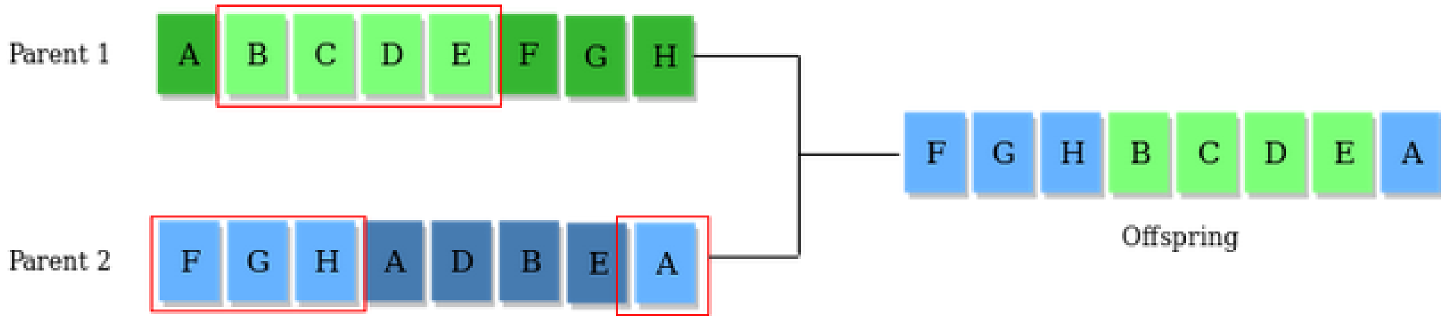
- "**Energy**" can take many forms
  - One common idea - learning is **Hebbian**
  - **Hebb Rule:** "Fire together, wire together"



Genetic Algorithms

Central idea:

- Adaptation, J.H. Holand, 1975
- Inspired by evolutionary biology concepts:

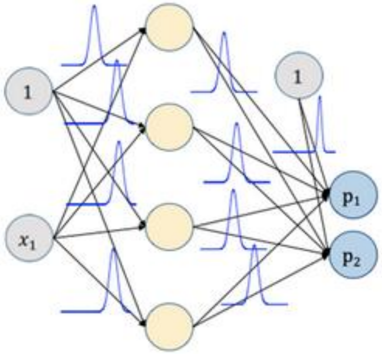


Genetic Algorithms

Begin with a large population of random solutions

- Evaluate each one
- **Fitness** function (some form of $S/\sqrt{B}$)
- Keep the **best subset**
  - Use it to build new solutions
  - Allow **mutation, cross-over**
  - Optimize over number of epochs/cycles

Bayesian Neural Networks



Bayesian Neural Networks

- Stochastic neural network with Bayesian inference
  - **Introduce stochasticity** into the network
    - Stochastic activations or weights
    - i.e. probability distributions (prior distribution over possible model parametrization)
- Trained as an **ensemble**
  - Aggregate the predictions
  - Natural uncertainty estimate

Bayesian Neural Networks

Given: $p(w \mid T) = p(T \mid w)\, p(w) / p(T)$

over the parameter space of the functions

$$n(\boldsymbol{x}, w) = 1 / [1+\exp(-f(\boldsymbol{x}, w))]$$

can estimate $p(s \mid \boldsymbol{x})$ as follows

$$p(s \mid \boldsymbol{x}) \sim n(\boldsymbol{x}) = \int n(\boldsymbol{x}, w)\, p(w \mid T)\, dw$$

- $n(\boldsymbol{x})$ is called a **Bayesian Neural Network** (BNN)

Bayesian Neural Networks

Computing Bayesian posterior is usually intractable

Approximate instead:

- with **Laplace Method**
  - low accuracy
- with **Markov Chain Monte Carlo (MCMC)**
  - long convergence
- with **variational approach**
  - approximates the exact posterior

Bayesian Neural Networks

MCMC approach: typically generate sample

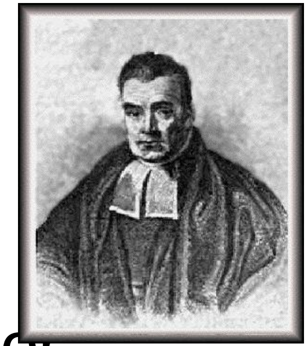
- **N** points {**w**} from p(**w** | T) using a Markov chain Monte Carlo (MCMC) technique

- average over the last **M** points

$$n(\mathbf{x}) = \int n(\mathbf{x}, w)\, p(w \mid T)\, dw$$

$$\sim \sum n(\mathbf{x}, w_i) / \mathbf{M}$$

Bayesian Neural Networks

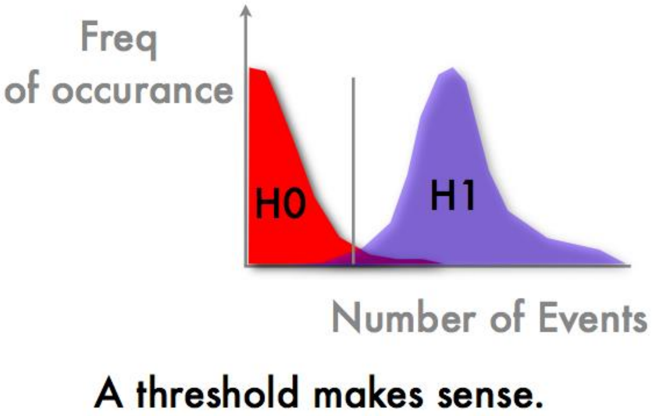


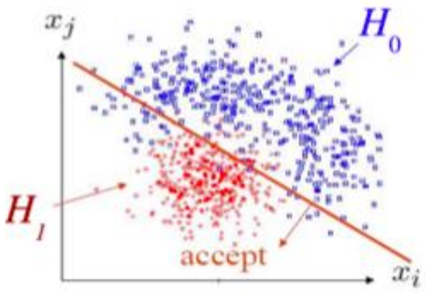
Variational method:

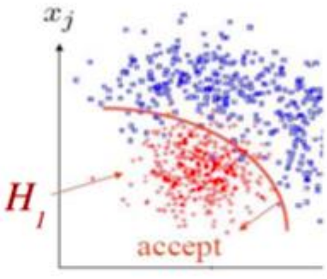
- **Scales** better
- Not exact
  - instead of sampling from exact posterior – **parametrized** by **variational distribution**
- Learn and perform **inference** s.t. the variational distribution is as close as possible to the posterior
  - Use K-L divergence and SGD
- Efficient way to approximate **Bayesian inference**

Uncertainties

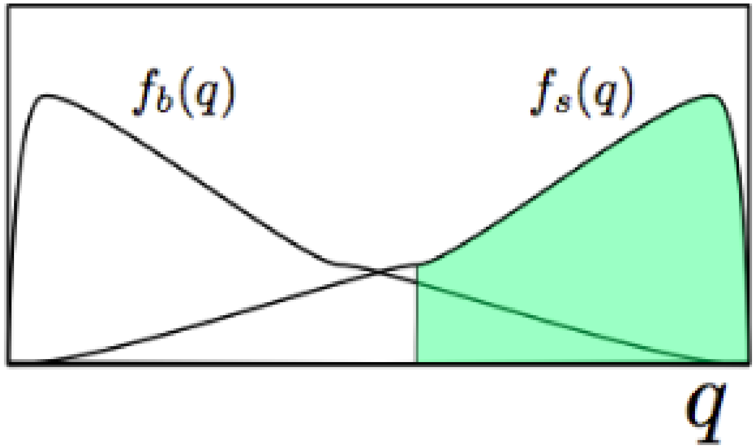
Uncertainties



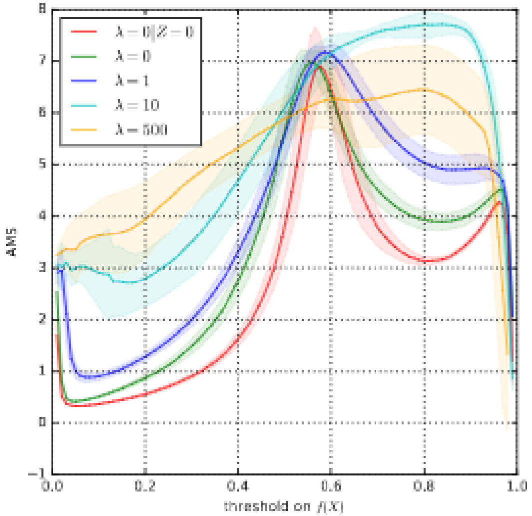




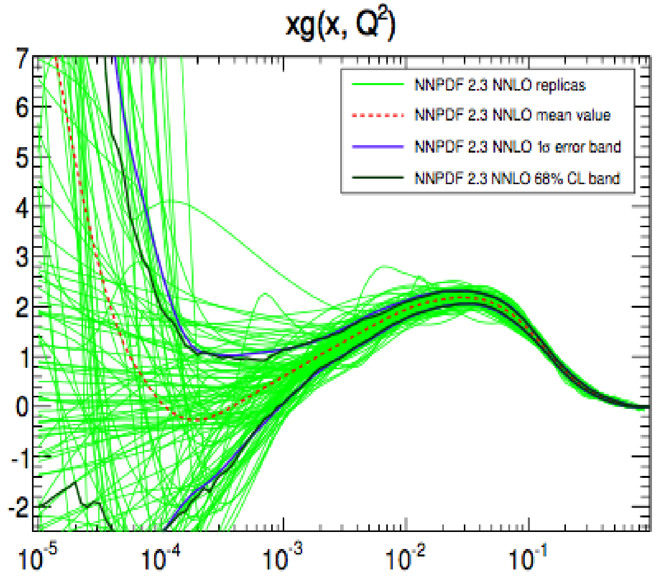
- **Impact on decision making**



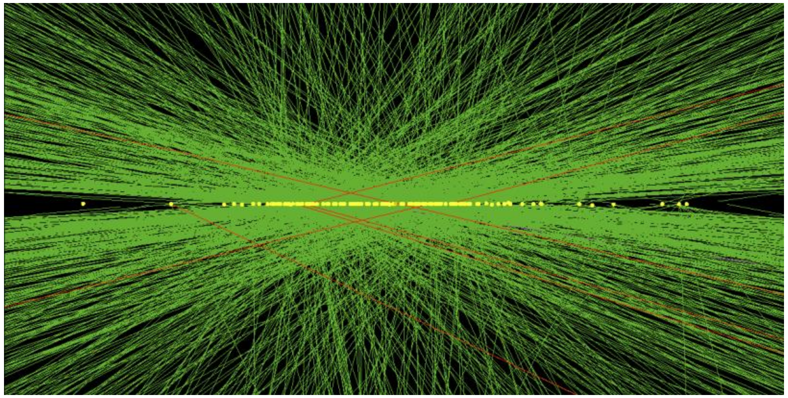
Uncertainties



Uncertainties matter



Bayesian connection: Deep neural networks with drop-out approximate variational inference of Bayesian NNs: *Gal and Ghahramani, 2016*

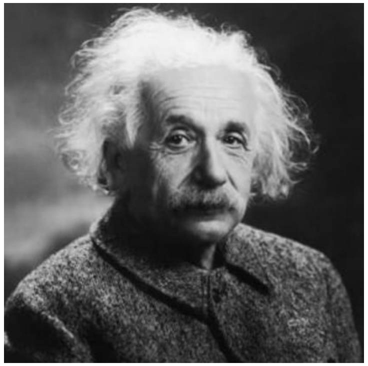
 

Physics of Deep Learning

Covariance in Physics

“The general laws of nature are to be expressed by equations which hold good for all systems of coordinates, that is, are covariant with respect to any substitutions whatever (generally covariant)”

A. Einstein,1916



Symmetries

- **Conservation laws/principles:**
  - **Mass, Energy**
- **Symmetries:**
  - **Rotation, Translation, Time, Reflection, Boost (relativity)**
  - **Many laws are equivariant to these symmetries**
- **Can think of this as prior knowledge**

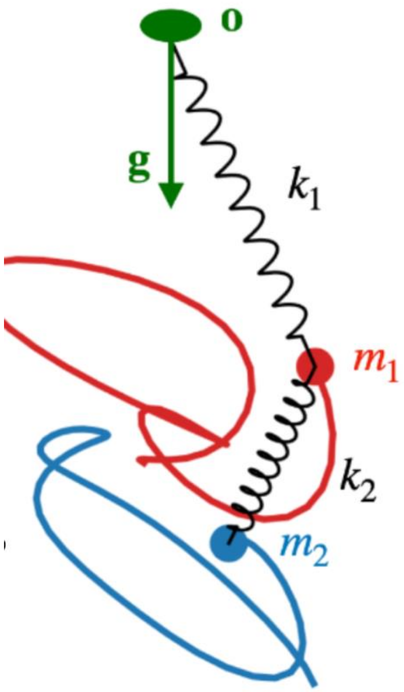
Covariance in Deep Learning

- **Apply concept of covariance - gauge invariance to design networks obeying certain symmetries**
  - **"Invariant scalars" 2110.03761**

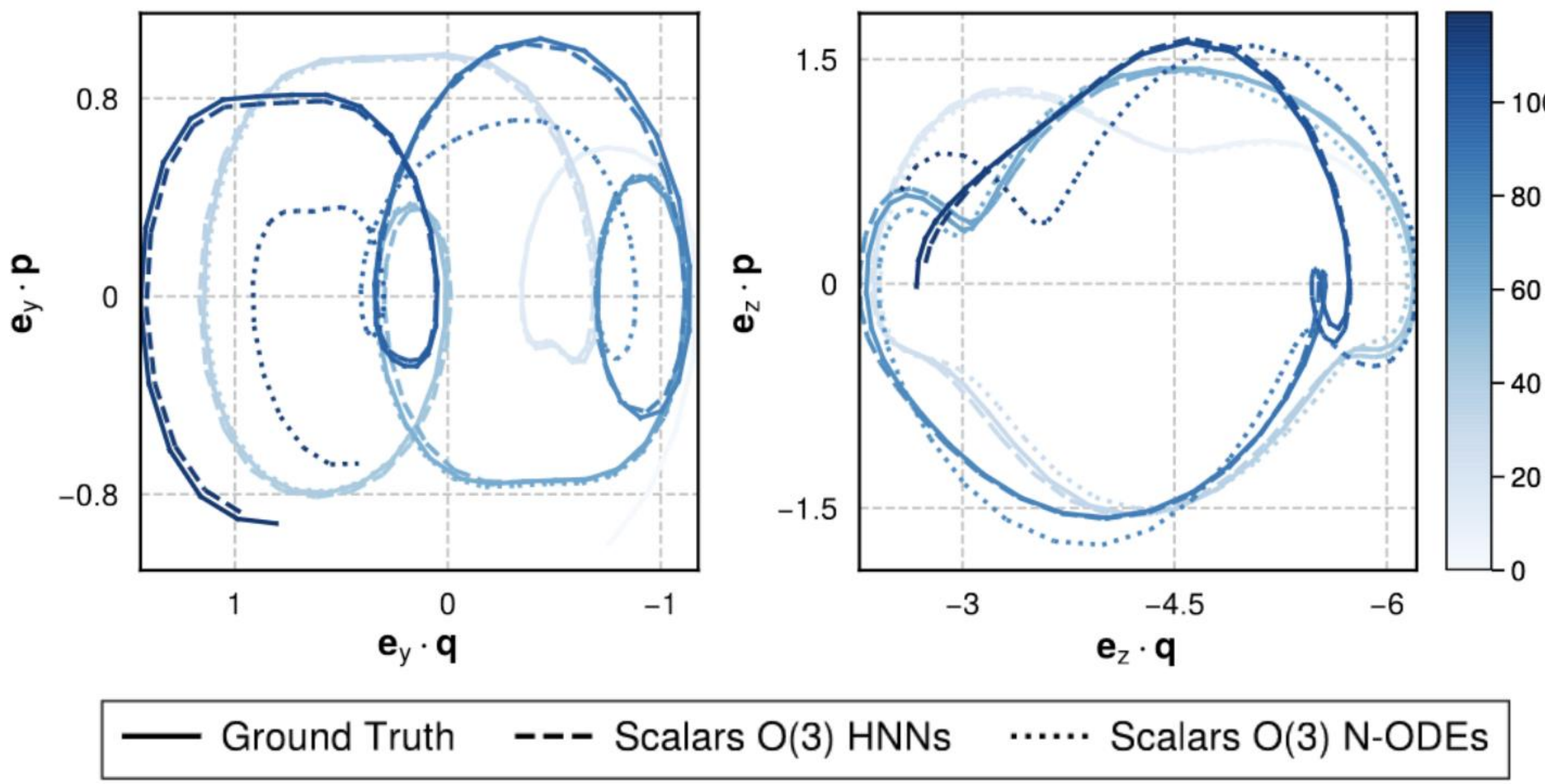
- **Equivariant CNN with known symmetry**
  - **Weiler and Cesa arXiv:1902.04615**
    - **Normal CNNs only have translational equivariance**
    - **Eg. can be any isometries of E(2): translations, rotations, reflections**

Invariant Scalars

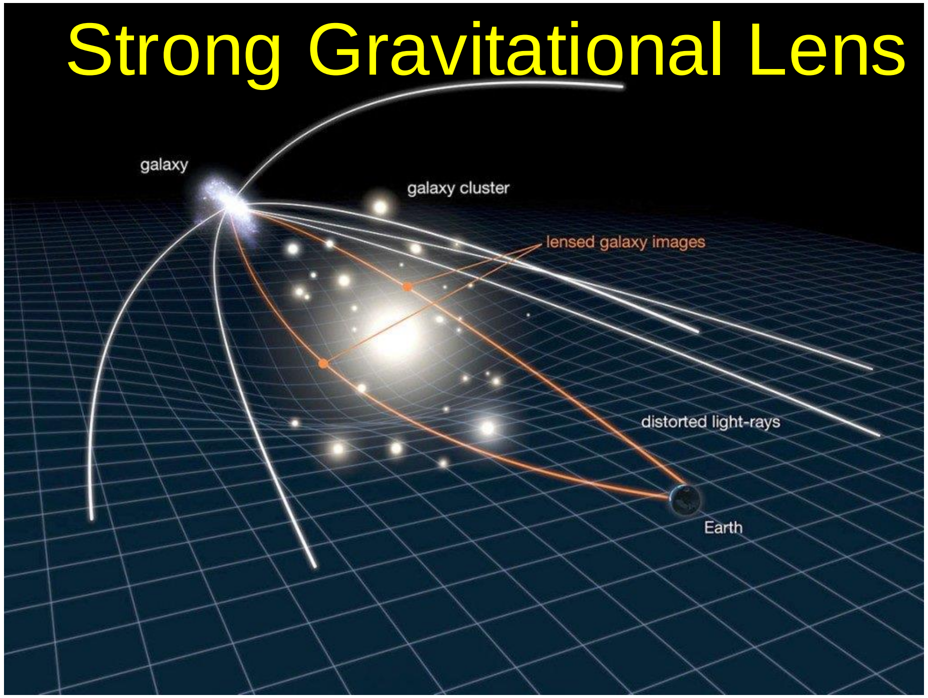
- Construct products of vectors for inputs (rotation/translation invariant)
  - Incorporate fundamental symmetry
- Learn the dynamical system with neural networks
  - Double pendulum example
  - Yao et al. Arxiv: 2110.03761

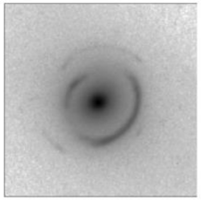


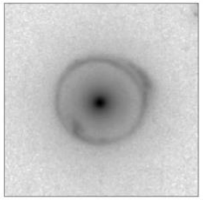
Invariant Scalars



Dark Matter and Deep Learning



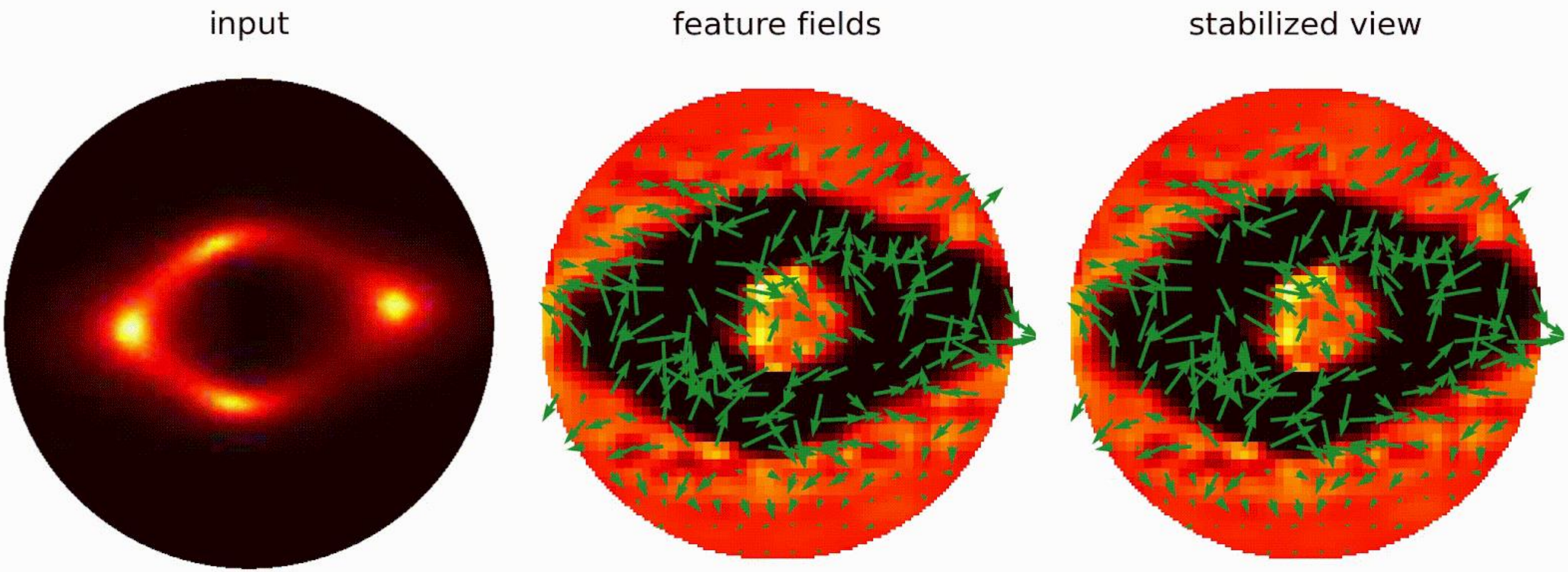




Can be used to infer dark matter physics

LSST/others will find ~10^4 lenses

Dark Matter and Deep Learning

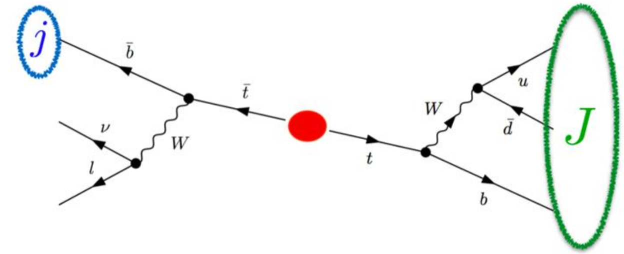


ResNet-18	ENN
0.963	**0.991**

Inductive Bias

- **We can also embed certain symmetries directly into the network architecture**
  - **By directly introducing constraints into the construction of a network layer (e.g invariance to a given symmetry)**
  - **typically achieved with special layers**
    - **Mattheakis et al. arXiv:1904.08991**

LoLA



1. CoLa* - learns the jet clustering history

$$k_{\mu,i} \xrightarrow{\text{CoLa}} \widetilde{k}_{\mu,j} = k_{\mu,i}\, C_{ij}$$

- Test on-shell conditions

$$\tilde{k}^2_{\mu,1} = (k_{\mu,1} + k_{\mu,2} + k_{\mu,3})^2 = m_t^2$$
$$\tilde{k}^2_{\mu,2} = (k_{\mu,1} + k_{\mu,2})^2 = m_W^2 \ .$$

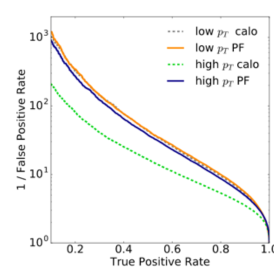
$$C = \begin{pmatrix} 1 & 0 & \cdots & 0 & C_{1,N+2} & \cdots & C_{1,M} \\ 0 & 1 & & \vdots & C_{2,N+2} & \cdots & C_{2,M} \\ \vdots & \vdots & \ddots & 0 & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & C_{N,N+2} & \cdots & C_{N,M} \end{pmatrix}$$

2. LoLa** - learns the kinematics

$$\tilde{k}_j \xrightarrow{\text{LoLa}} \hat{k}_j = \begin{pmatrix} m^2(\tilde{k}_j) \\ p_T(\tilde{k}_j) \\ w^{(E)}_{jm}\, E(\tilde{k}_m) \\ w^{(d)}_{jm}\, d^2_{jm} \end{pmatrix}$$

transform 4-vectors into: invariant mass, pT, energy and Minkowski distance
effectively a rotation in observable space

arXiv:1707.08966



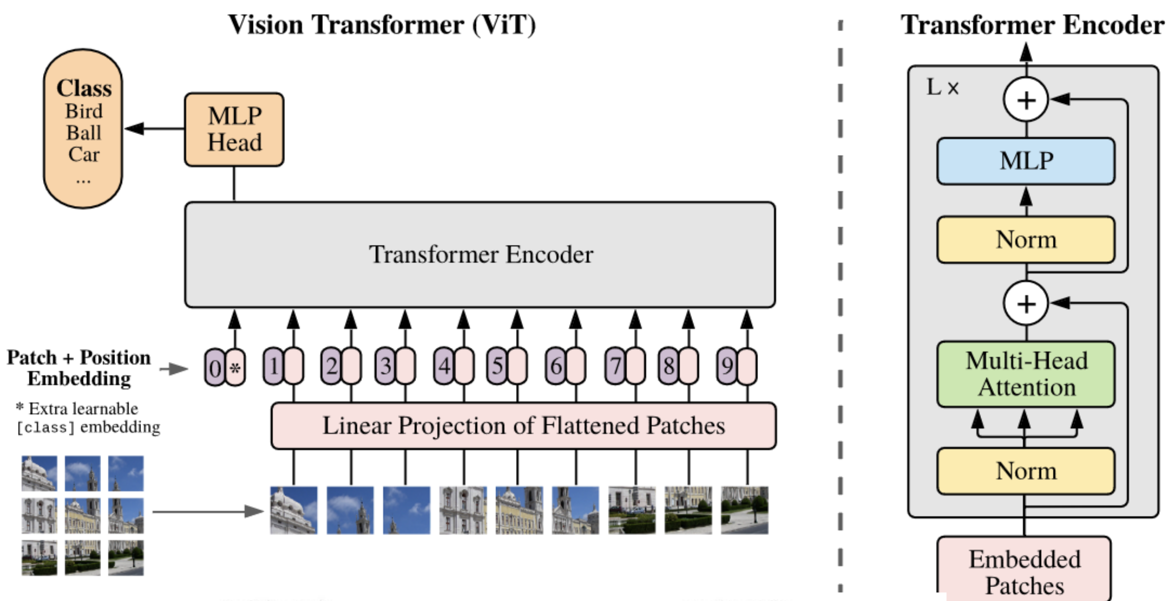
** CoLa = Combination Layer*

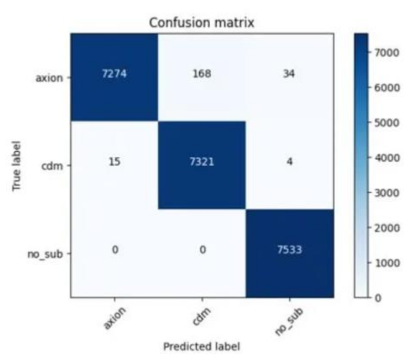
*** LoLa = Lorentz Layer*

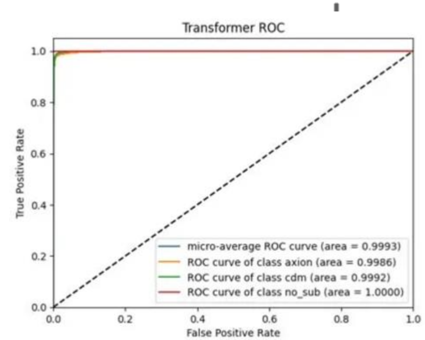
More examples

- **Lagrangian Deep Learning**
  - **Dai and Sejlak arXiv:2010.02926**
  - **Specialized layers that follow a Lagrangian Formulation to encapsulate the symmetry**
- **Learning bias with Loss Functions**
  - **Constrain the neural network**
- **Physics-informed learning**
  - **Example: Encode Navier-Stokes into the network (Raissi et al. Science 2020)**
  - **Karniadakis et al. Nature Reviews (2021)**

Vision Transformer





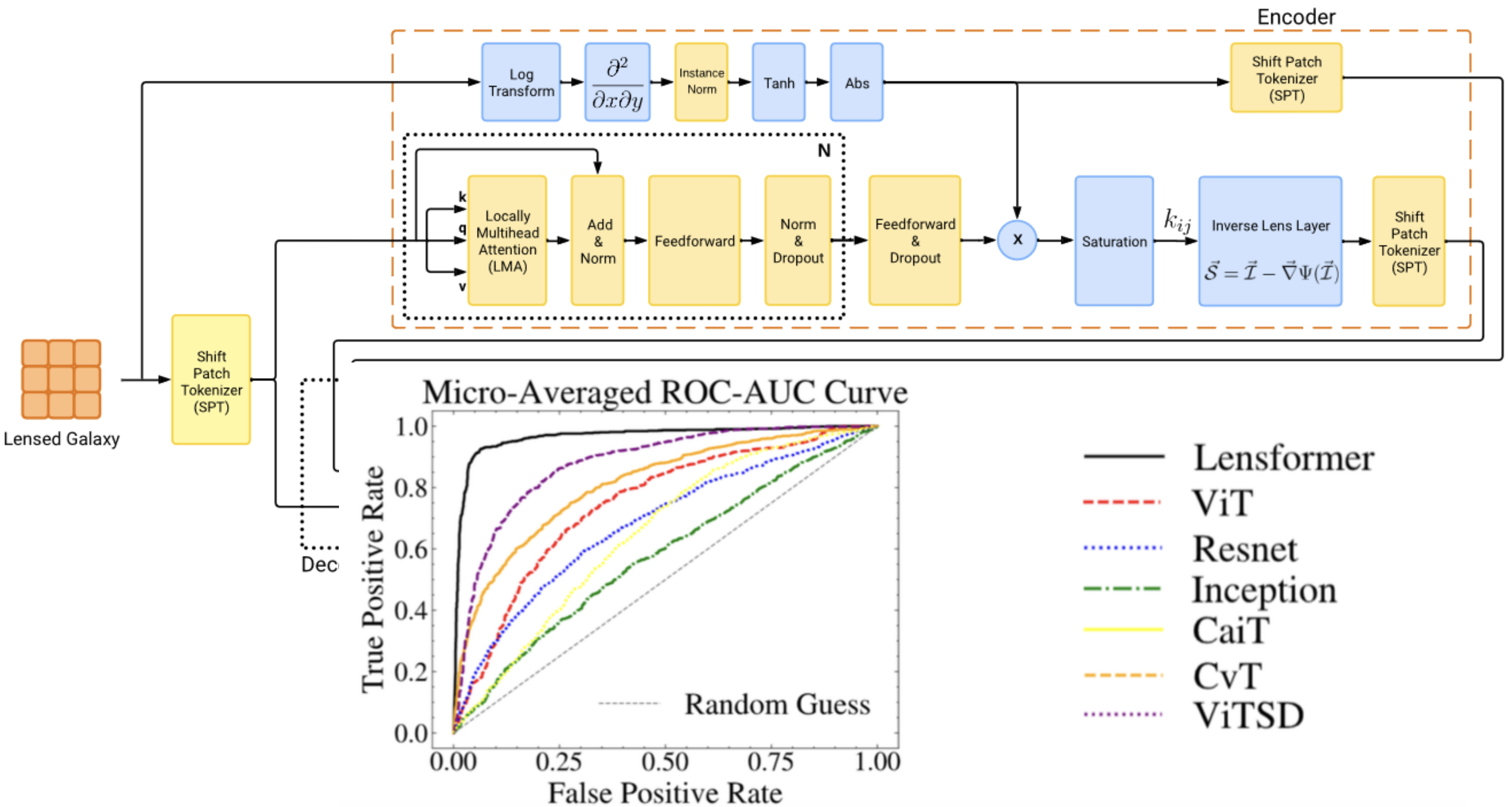


Physics Informed AI

- Incorporate *Physics* into the *Machine Learning* model directly
- Example: *LensFormer*
  - physics-informed transformer for strong gravitational lensing
  - Incorporates the lens equation into the model

Veloso et al. (2023)

LensFormer



Veloso et al. (2023)