Title: Quantum hypernetworks

Speakers: Juan Felipe Carrasquilla Ã•lvarez

Collection: New Frontiers in Machine Learning and Quantum

Date: November 22, 2022 - 3:45 PM

URL: https://pirsa.org/22110089

Deep learning

- Computer vision, natural language processing, machine translation, self driving cars, game playing, physics, chemistry, finance, healthcare, demographics, entertainment, music, art, robotics.
- Availability of datasets, specialized hardware, and algorithmic developments have ushered a new generation of large models displaying unprecedented accuracy across a wide array of technologically and scientifically relevant tasks in artificial intelligence.
- Example: Diffusion models
- Impressive results
- Art will change dramatically

https://strikingloo.github.io/stable-diffusion-vs-dalle-2

Prompt: "Cute and adorable ferret wizard, wearing coat and suit, steampunk, lantern, anthromorphic, Jean paptiste monge, oil painting"

Deep learning

- Computer vision, natural language processing, machine translation, self driving cars, game playing, physics, chemistry, finance, healthcare, demographics, entertainment, music, art, robotics.

- Availability of datasets, specialized hardware, and algorithmic developments have ushered a new generation of large models displaying unprecedented accuracy across a wide array of technologically and scientifically relevant tasks in artificial intelligence.

- Example: Diffusion models

- Impressive results

- Art will change dramatically

https://strikingloo.github.io/stable-diffusion-vs-dalle-2

Prompt: “Cute and adorable ferret wizard, wearing coat and suit, steampunk, lantern, anthromorphic, Jean paptiste monge, oil painting”

Deep learning

- Example: "Galactica: A Large Language Model for Science" https://arxiv.org/abs/2211.09085
- Prompt a scientific topic and the language model writes a manuscript for you. Surprising results.
- Meta shuts down public test of Galactica, its 'AI for Science' because it produced pseudoscientific papers

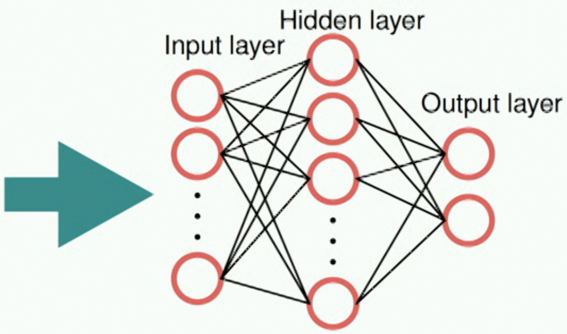
Neural Language models

- A neural network language model is a language model based on neural networks
- Neural networks are powerful universal function approximators and can in principle compute any function
- We can ask whether these models represent complex quantum states and use them for reconstruction or simulation

$$P(\text{output text}|\text{input text})$$

$$P(\text{English}|\text{Spanish})$$

$$P(\text{Spanish})$$



Deep learning

Estelle: “It's really insane!”

× not verified

Recurrent Neural Network wavefunctions

We can use a Recurrent Neural Network (RNN) to model the quantum wavefunction of a many-body system.

In this notebook we show how to implement this model and train it on the ground state of the Ising model.

For this notebook we will use the TensorFlow Keras API.

The model

The RNN wavefunction has the following structure:

Each site is represented by a spin variable s_i which can take two values, ± 1.

The wavefunction is parametrized by an RNN, where each spin s_i is treated as a time step in the RNN. The hidden state h_i at each time step i is updated according to the following rule:

$$h_i = f(s_i W^{(1)} + h_{i-1} W^{(2)})$$

where f is an activation function, $W^{(1)}$ and $W^{(2)}$ are matrices which parametrize the RNN. The output at each time step is given by:

$$\psi(s_1, \ldots, s_N) = f(h_N W^{(3)})$$

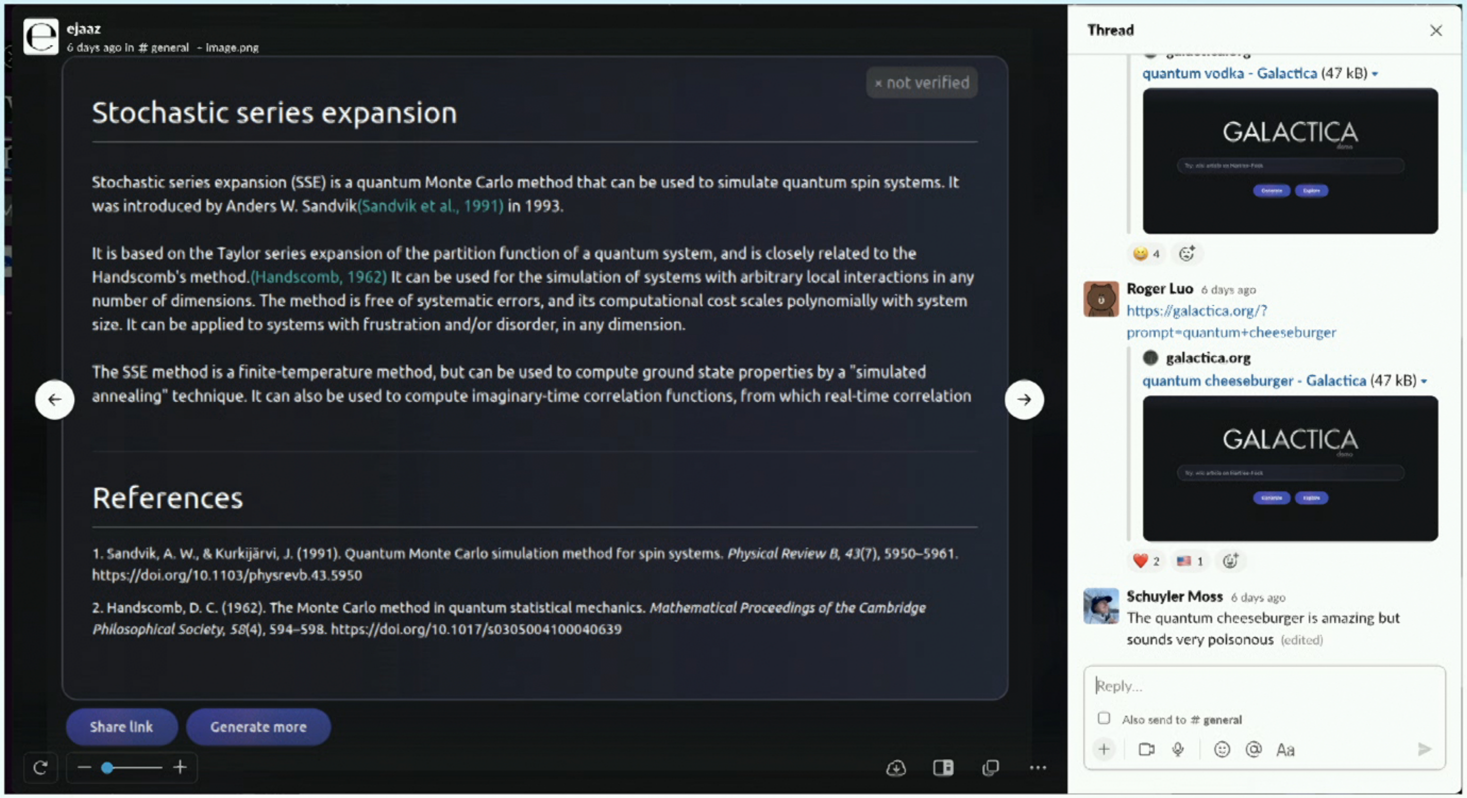
The matrices $W^{(1)}$, $W^{(2)}$ and $W^{(3)}$ are trained using Variational Monte Carlo (VMC).

Data

We train the RNN on the ground state of the one-dimensional transverse field Ising model:

Deep learning

Ejaaz: "pretty good honestly"



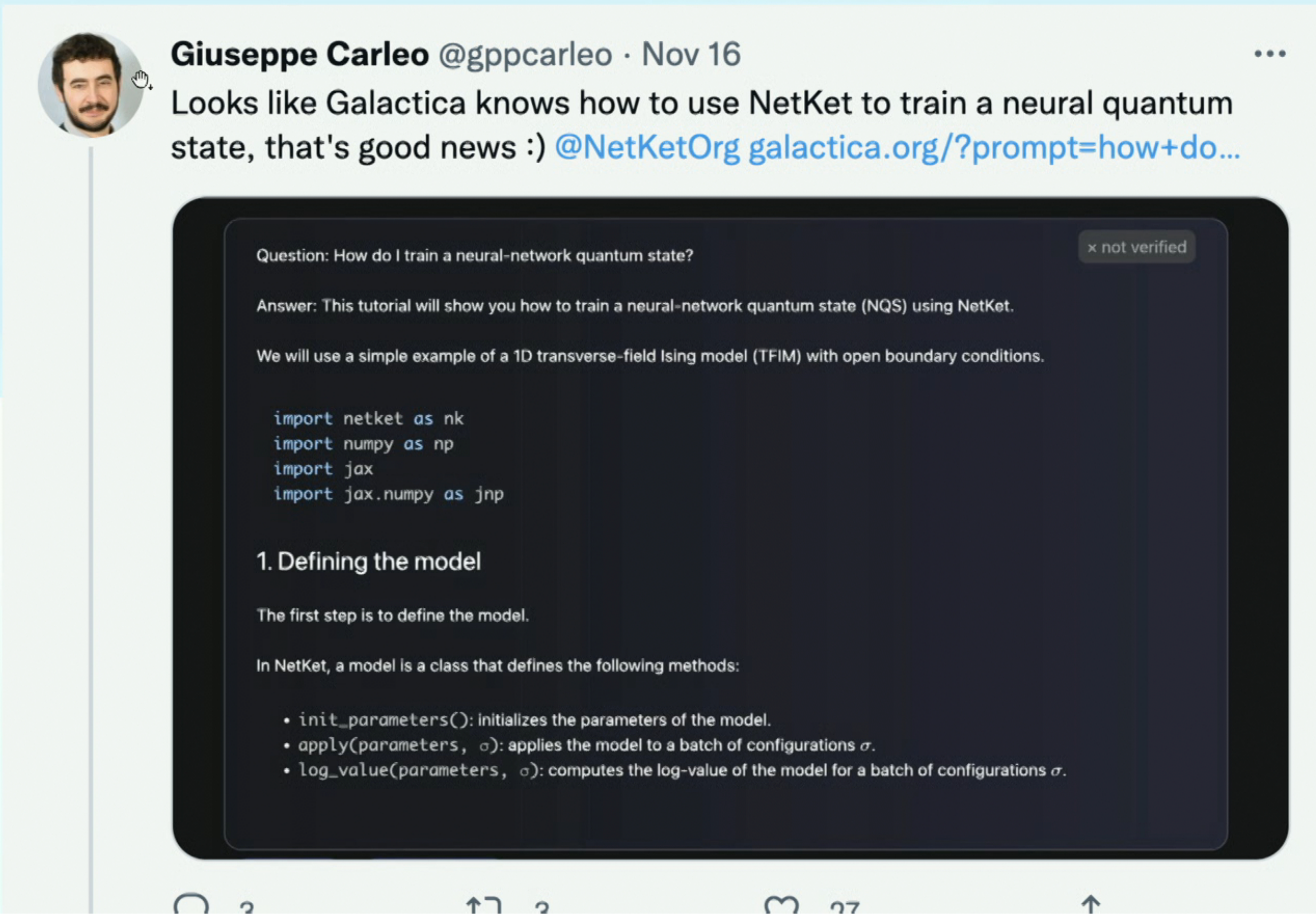
Deep learning

Schuyler Moss 6 days ago
typed in "review of variational ansatze for the anti-ferromagnetic heisenberg model on the kagome lattice"

Schuyler Moss 6 days ago
"The Heisenberg antiferromagnet on the kagome lattice is a paradigmatic example of a frustrated quantum magnet. Despite a long history of research, the nature of its ground state remains elusive. Recent years have seen an explosion of numerical studies, based on a variety of techniques, which have unveiled several unexpected properties. Among the most striking findings is the absence of magnetic order, which is not in line with the usual semi-classical picture. Other features include a large density of low-lying singlets below the first triplet excitation, and a very rich phenomenology under applied magnetic field. While these findings challenge our understanding of the model, they also raise new questions, and in particular it is not clear whether the ground state of the model is a spin liquid with fractionalized excitations, or a more conventional state with broken lattice symmetries."

7

Deep learning



Giuseppe Carleo @gppcarleo · Nov 16

Looks like Galactica knows how to use NetKet to train a neural quantum state, that's good news :) @NetKetOrg galactica.org/?prompt=how+do…

× not verified

Question: How do I train a neural-network quantum state?

Answer: This tutorial will show you how to train a neural-network quantum state (NQS) using NetKet.

We will use a simple example of a 1D transverse-field Ising model (TFIM) with open boundary conditions.

```
import netket as nk
import numpy as np
import jax
import jax.numpy as jnp
```

1. Defining the model

The first step is to define the model.

In NetKet, a model is a class that defines the following methods:

- `init_parameters()`: initializes the parameters of the model.
- `apply(parameters, σ)`: applies the model to a batch of configurations σ.
- `log_value(parameters, σ)`: computes the log-value of the model for a batch of configurations σ.

3 3 27

But doing this is expensive

- These advances crucially depend on the availability of specialized computational resources such as graphics and tensor processing units, which demand a high electricity consumption.
- In particular, a set of key but computationally expensive elements in the modern machine learning (ML) workflow include hyperparameter optimization and neural architecture search.
- GPT-3's training costs to be up to $27.6 million

The GPT-3 economy: https://bdtechtalks.com/2020/09/21/gpt-3-economy-business-model/

Binary Neural networks

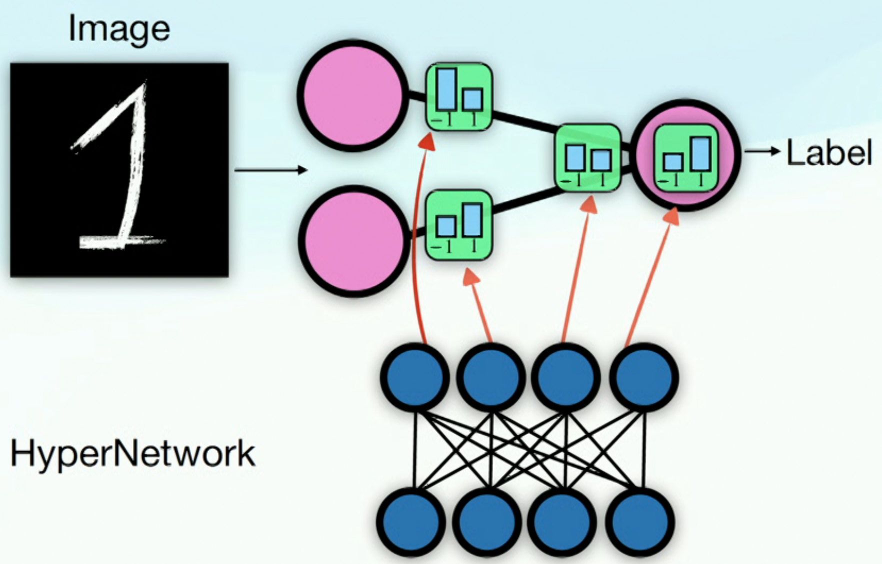
- ✅ Neural networks with binary weights and activations (BiNNs) partially alleviate these issues as they are computationally efficient, hardware-friendly, and energy efficient.
- ✅ 32-fold reduction in memory.
- ✅ Robust to adversarial attacks.
- ✅ Specialized hardware implementations that simultaneously increase computational speed and improve their energy efficiency.
- ❌ Parameter, hyperparameter, and architectural searches remains computationally expensive— multiple nested combinatorial optimization problems (training parameters on training set+ outerloop on hyper parameters and architectural search on a validation set)

Binary Neural Networks

- Traditionally, there are two loops: outer optimization loop which searches through the hyperparameter and architectural state spaces guided by the model's performance on a validation set, and an inner optimization which adjusts the weights of the neural network on a training set.
- Such a nested optimization process remains the most computationally demanding task in the modern ML workflow and entails an unsustainable carbon footprint, which calls for computationally efficient hardware and algorithms to train and search for neural architectures

HyperNetworks

- HyperNetworks: an approach of using a one network, also known as a hypernetwork, to generate the weights for another network.



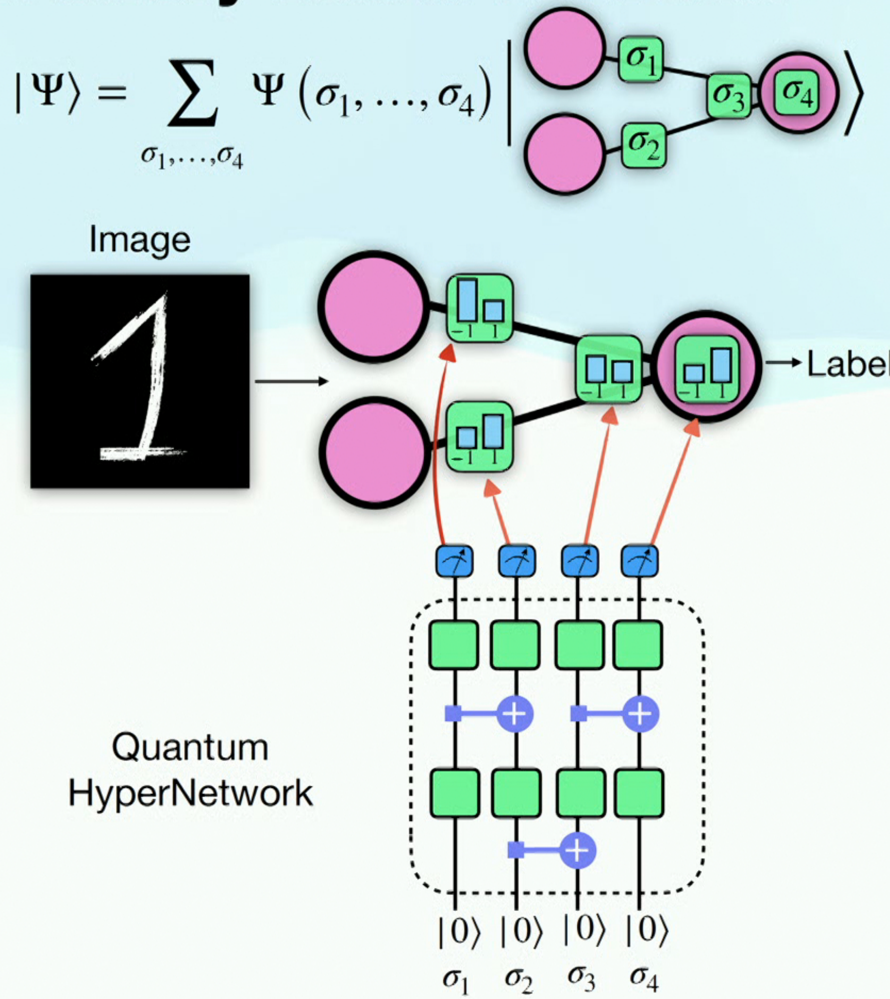
- Used in natural language processing, computer vision, hyperparmeter tuning, neural architectural search, meta-learning.

- HyperNetworks. https://arxiv.org/abs/1609.09106

Quantum HyperNetworks

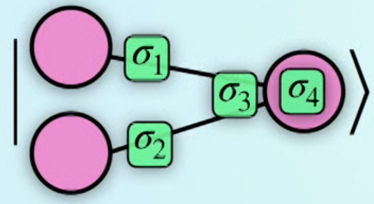
Quantum HyperNetworks to train binary neural networks

- We define Quantum HyperNetworks and use them to unify parameter, hyperparameter, and architectural search for binary neural networks in just **one** optimization loop
- Can be understood as training binary neural networks in quantum superposition
- Superpositions contain exponentially many binary neural networks with different parameters, architectural choices, and hyperparameters

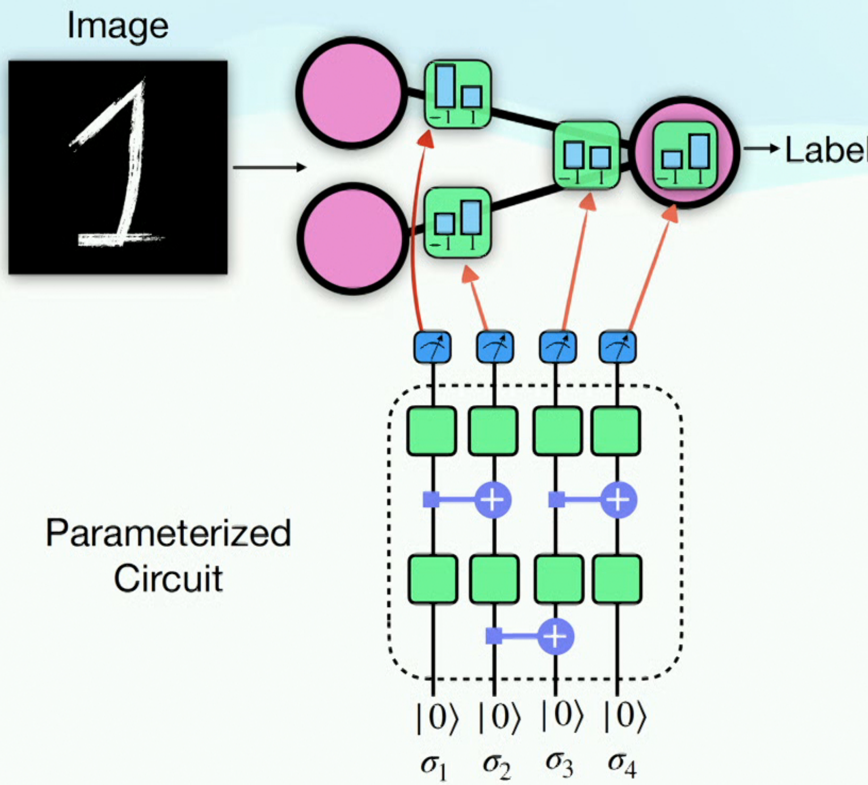


Encoding BiNNs in a quantum state

$$|\Psi\rangle = \sum_{\sigma_1,\ldots,\sigma_4} \Psi\left(\sigma_1,\ldots,\sigma_4\right)\Big|$$

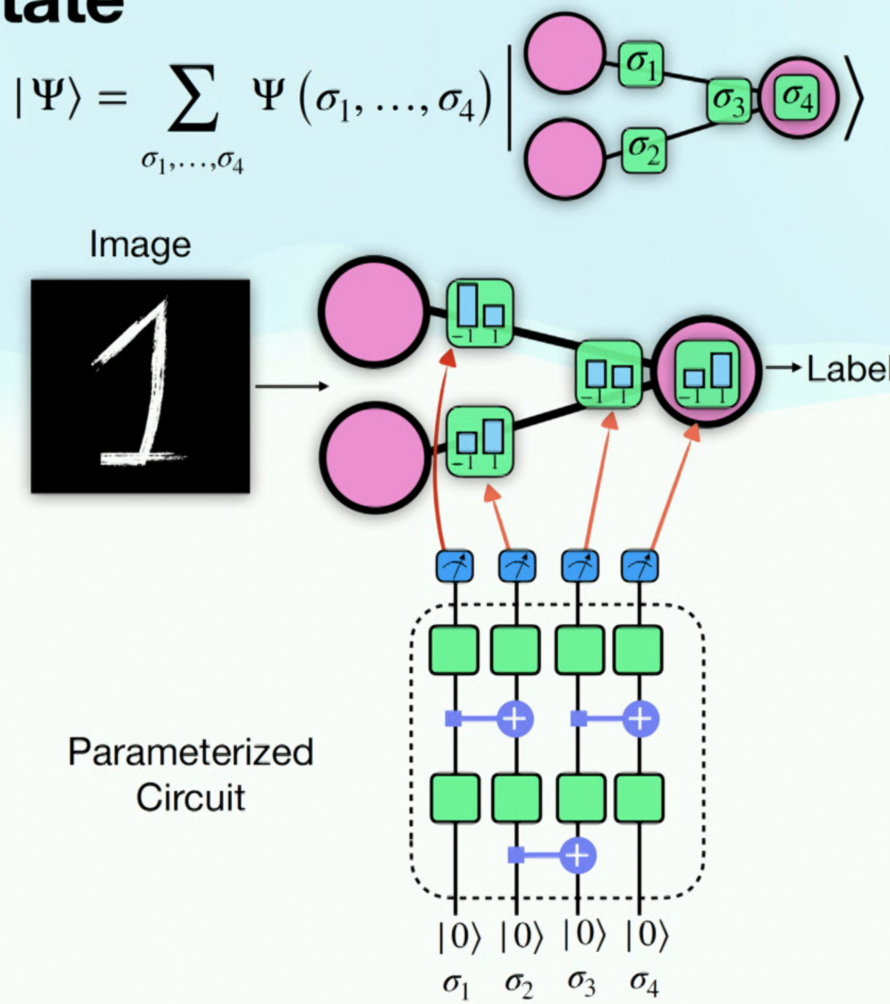


- Consider a quantum state
- $$|\Psi\rangle = \sum_{\sigma_1,\ldots,\sigma_N} \Psi(\sigma_1,\ldots,\sigma_N)\,|\,\sigma_1,\ldots,\sigma_N\rangle$$
- To each basis element $|\boldsymbol{\sigma}\rangle = |\,\sigma_1,\ldots,\sigma_N\rangle$ we associate a specific configuration of an augmented model comprising the weights of a BiNN, its hyperparameters, and any desired architectural choices to be encoded in the VQA search.
- Characterized by 2 weights (qubits σ_1 and σ_2), a bias (qubit σ_3), and an activation function (architectural choice). The selection of activation function from two possibilities f_1 or f_2, we make the activation function qubit dependent (qubit σ_4). $f(x) \rightarrow f(x, \sigma_4)$



Encoding BiNNs in a quantum state

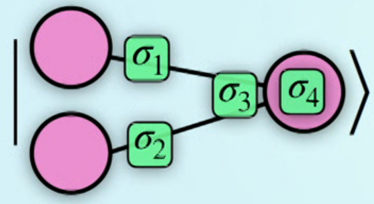
- $f(\boldsymbol{x};\sigma) = \begin{cases} f_1(\boldsymbol{x}) & \text{if } \sigma = 0 \\ f_2(\boldsymbol{x}) & \text{if } \sigma = 1. \end{cases}$
- Other architectural choices can be encoded (skip connections, dimension of the hidden layer, # of layers, etc), just add more qubits.
- How can we "nudge" the state so that when we measure it in an experiment, it returns neural networks with good architectural choices, parameters, and hyperparameters?



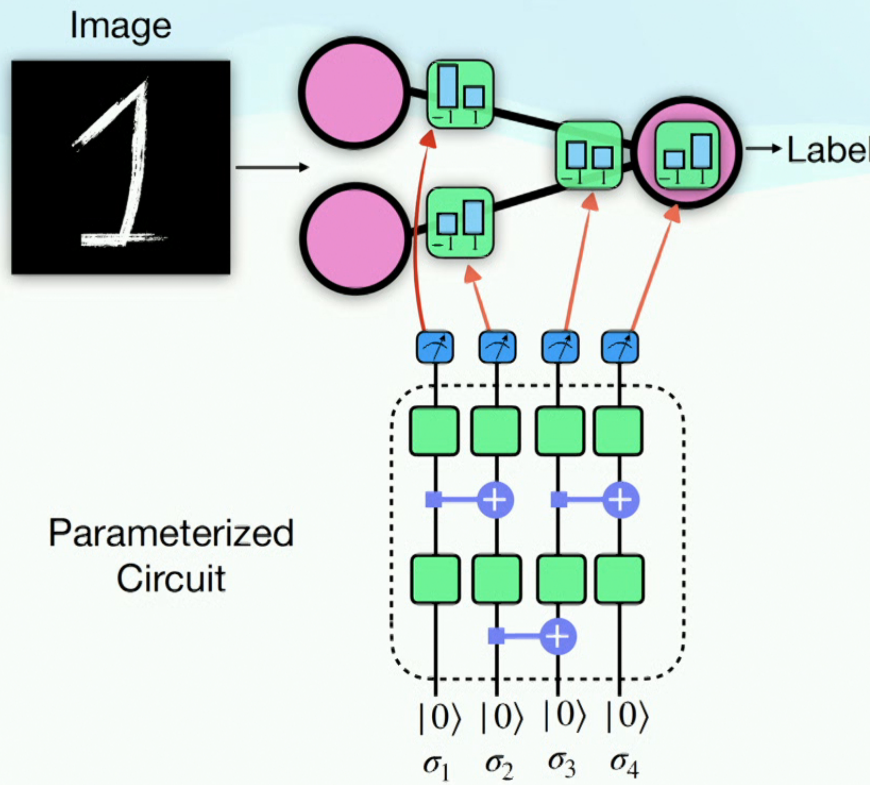
Variational quantum algorithm

Encode the problem in a form suitable to optimization by a variational quantum algorithm

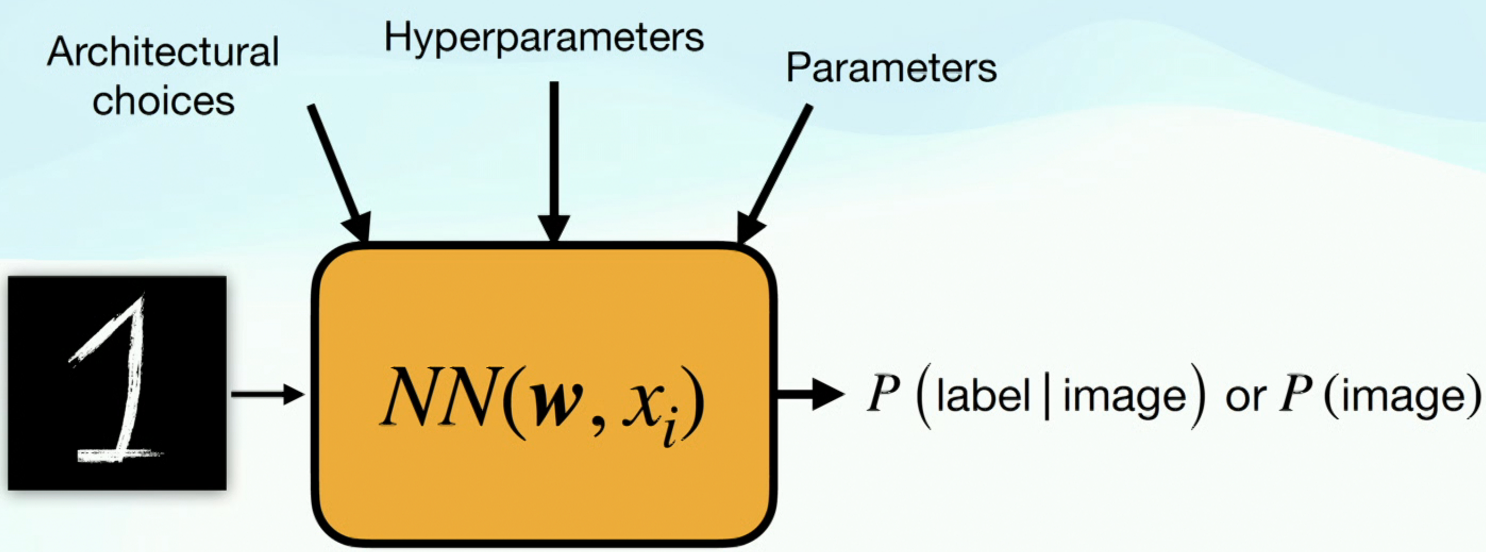
$$|\Psi\rangle = \sum_{\sigma_1,\ldots,\sigma_4} \Psi\left(\sigma_1,\ldots,\sigma_4\right)$$



- One idea: a variational quantum algorithm (VQA).
- A VQA employs a classical optimizer acting on a parameterized quantum circuit, with the purpose of finding solutions to a problem encoded in an objective function.
- Objective: $C(\boldsymbol{w}) = \frac{1}{N_t}\sum_{i=1}^{N_t} \mathscr{L}\left(\mathrm{NN}(\boldsymbol{x}_i; \{\boldsymbol{w}\}), \boldsymbol{y}_i\right)$.
- The augmented model parameters $\boldsymbol{w} = \{w_1, \ldots, w_N\}$, include the neural network weights, biases, hyperparameters, and architectural choices.

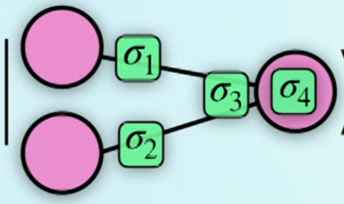


Augmented model

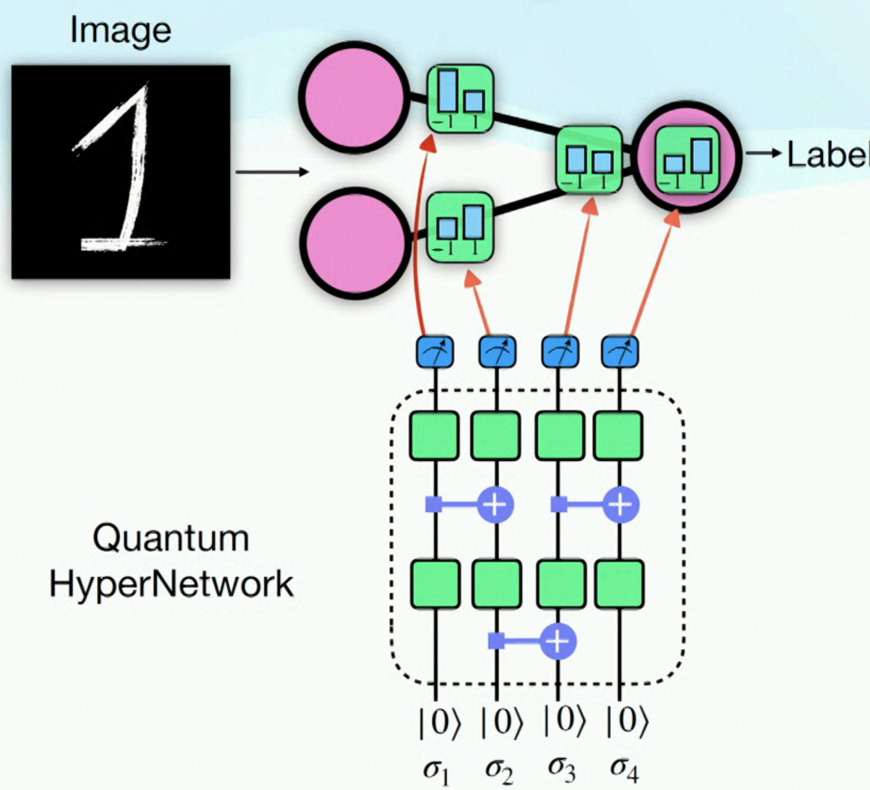


Variational quantum algorithm

Encode the problem in a form suitable to optimization by a variational quantum algorithm

$$|\Psi\rangle = \sum_{\sigma_1,\ldots,\sigma_4} \Psi(\sigma_1,\ldots,\sigma_4)\Big|$$  $$\Big\rangle$$

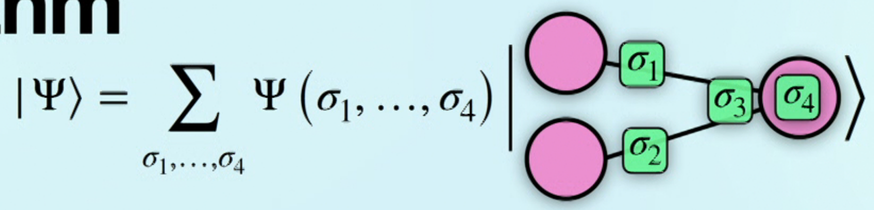
- Making the objetive function quantum
- Promote the parameters of the BiNN to a set of Pauli matrices
  $\boldsymbol{w} \to \hat{\boldsymbol{\sigma}}_z = \left(\hat{\sigma}_1^z, \hat{\sigma}_2^z, \ldots, \hat{\sigma}_N^z\right),$
- $C(\boldsymbol{w}) \to \hat{C}$ (i.e. go from a Boolean function to a big $2^N \times 2^N$diagonal matrix).
- This encoding is flexible — off-diagonal operators, multi-basis encoding



Variational Quantum Optimization with Multi-Basis Encodings. Taylor L. Patti, Jean Kossaifi, Anima Anandkumar, Susanne F. Yelin. https://arxiv.org/abs/2106.13304

Variational quantum algorithm

Encode the problem in a form suitable to optimization by a variational quantum algorithm



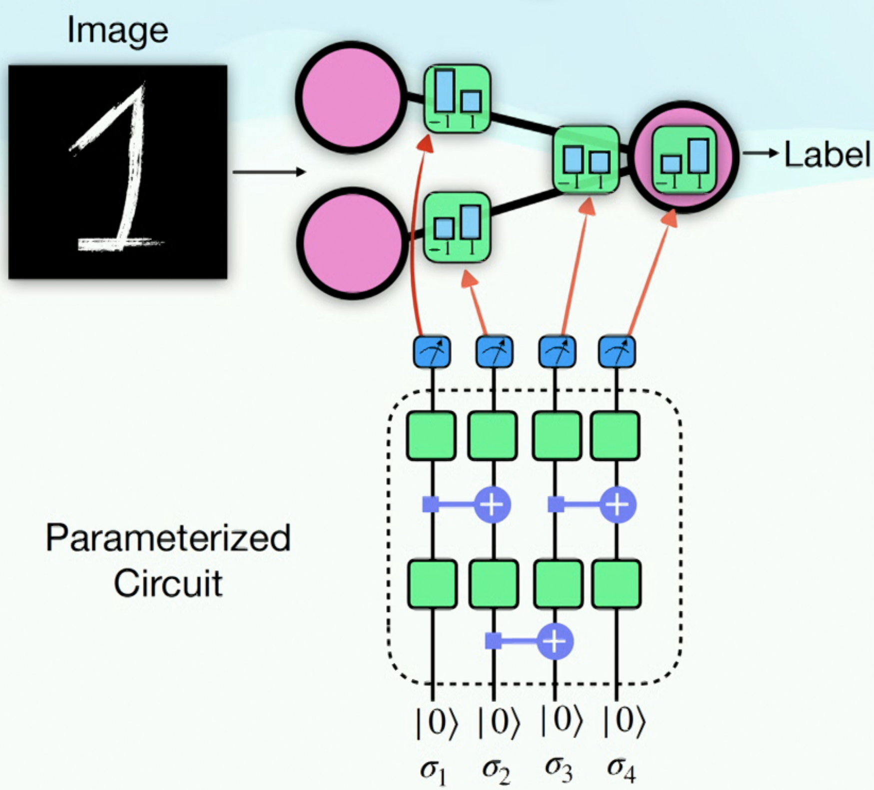
- We construct a quantum state $|\Psi\rangle$ through a parameterized quantum circuit $U(\boldsymbol{\theta})$ with continuous parameters $\boldsymbol{\theta}$ such that $|\Psi\rangle \rightarrow |\Psi_{\boldsymbol{\theta}}\rangle = U(\boldsymbol{\theta})|0\rangle^{\otimes n}$

- We aim at finding solutions to the training of the BiNN solving for

- $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} E(\boldsymbol{\theta}),$

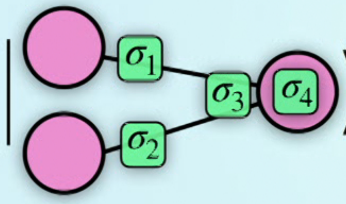
$$E(\boldsymbol{\theta}) = \langle\Psi_{\boldsymbol{\theta}}|\hat{C}|\Psi_{\boldsymbol{\theta}}\rangle \tag{6}$$
$$= \sum_{\sigma_1,\sigma_2,\ldots,\sigma_N} |\Psi_{\boldsymbol{\theta}}(\sigma_1,\sigma_2,\ldots,\sigma_N)|^2 C(\sigma_1,\sigma_2,\ldots,\sigma_N)$$
$$= \mathbb{E}_{\boldsymbol{\sigma}\sim|\Psi_{\boldsymbol{\theta}}|^2}[C(\boldsymbol{\sigma})] \approx \frac{1}{N_s}\sum_{i=1}^{N_s} C(\boldsymbol{\sigma}_i),$$

-



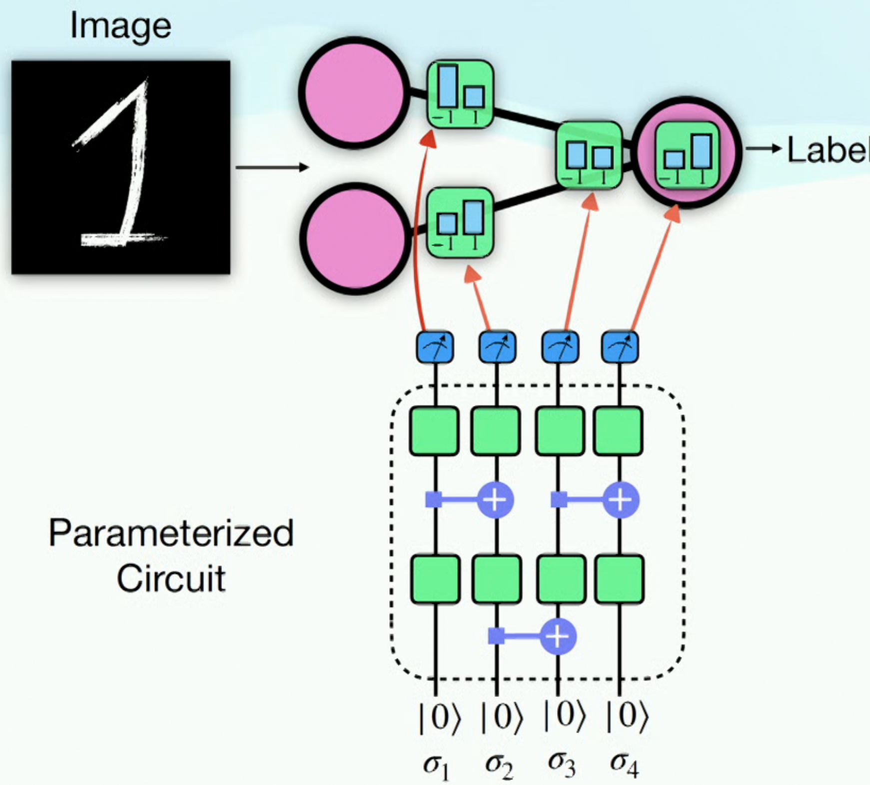
Variational quantum algorithm

Encode the problem in a form suitable to optimization by a variational quantum algorithm

$$|\Psi\rangle = \sum_{\sigma_1,\dots,\sigma_4} \Psi\left(\sigma_1,\dots,\sigma_4\right) \Big|$$  $$\Big\rangle$$

$$\begin{aligned} E(\boldsymbol{\theta}) &= \langle \Psi_{\boldsymbol{\theta}} | \hat{C} | \Psi_{\boldsymbol{\theta}} \rangle \qquad (6) \\ &= \sum_{\sigma_1,\sigma_2,\dots,\sigma_N} |\Psi_{\boldsymbol{\theta}}(\sigma_1,\sigma_2,\dots,\sigma_N)|^2 C(\sigma_1,\sigma_2,\dots,\sigma_N) \\ &= \mathbb{E}_{\boldsymbol{\sigma}\sim|\Psi_{\boldsymbol{\theta}}|^2}\left[C(\boldsymbol{\sigma})\right] \approx \frac{1}{N_s}\sum_{i=1}^{N_s} C(\boldsymbol{\sigma}_i), \end{aligned}$$

•



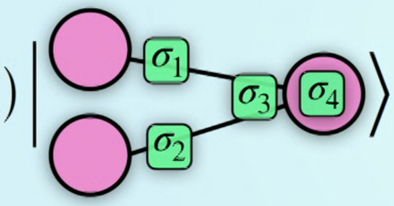
- From an ML perspective, this approach can be understood as a stochastic relaxation of the discrete optimization problem. This is close to a Bayesian BiNN with a “quantum” approximating posterior.
- Instead of optimizing binary variables, optimize continuous parameters $\boldsymbol{\theta}$.

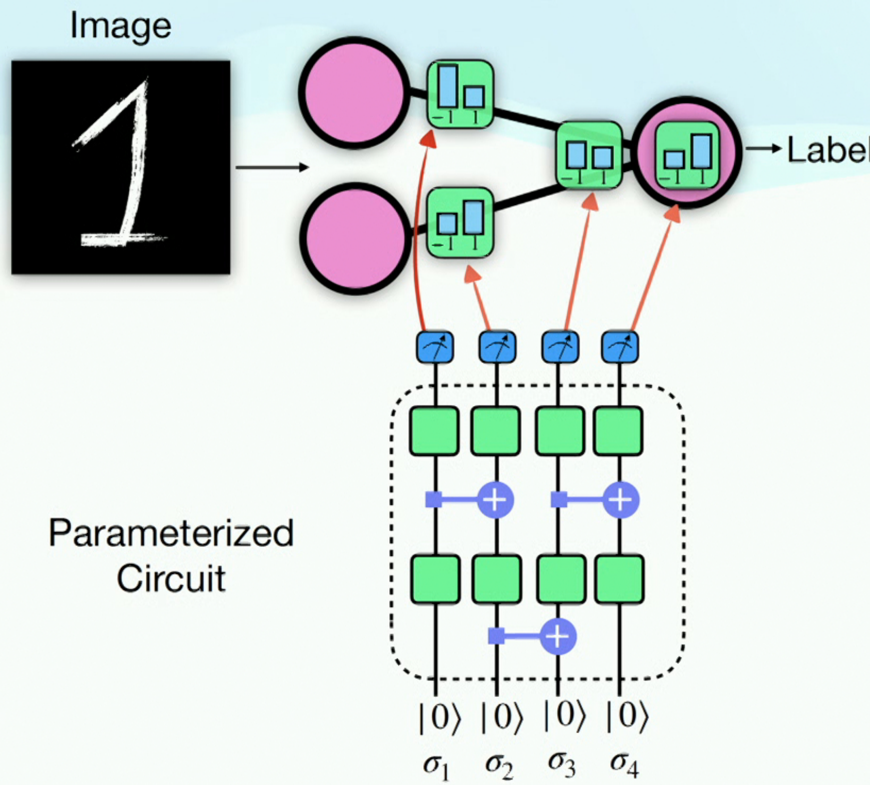
Variational quantum algorithm

Encode the problem in a form suitable to optimization by a variational quantum algorithm

- Design of the circuit is important. Depth, connectivity of the gates etc.
- As a boolean function, we don't know a whole lot about C
- We choose a circuit with linear connectivity and vary its depth.
- Most available quantum computers have this connectivity

$$|\Psi\rangle = \sum_{\sigma_1,\ldots,\sigma_4} \Psi\left(\sigma_1,\ldots,\sigma_4\right)$$

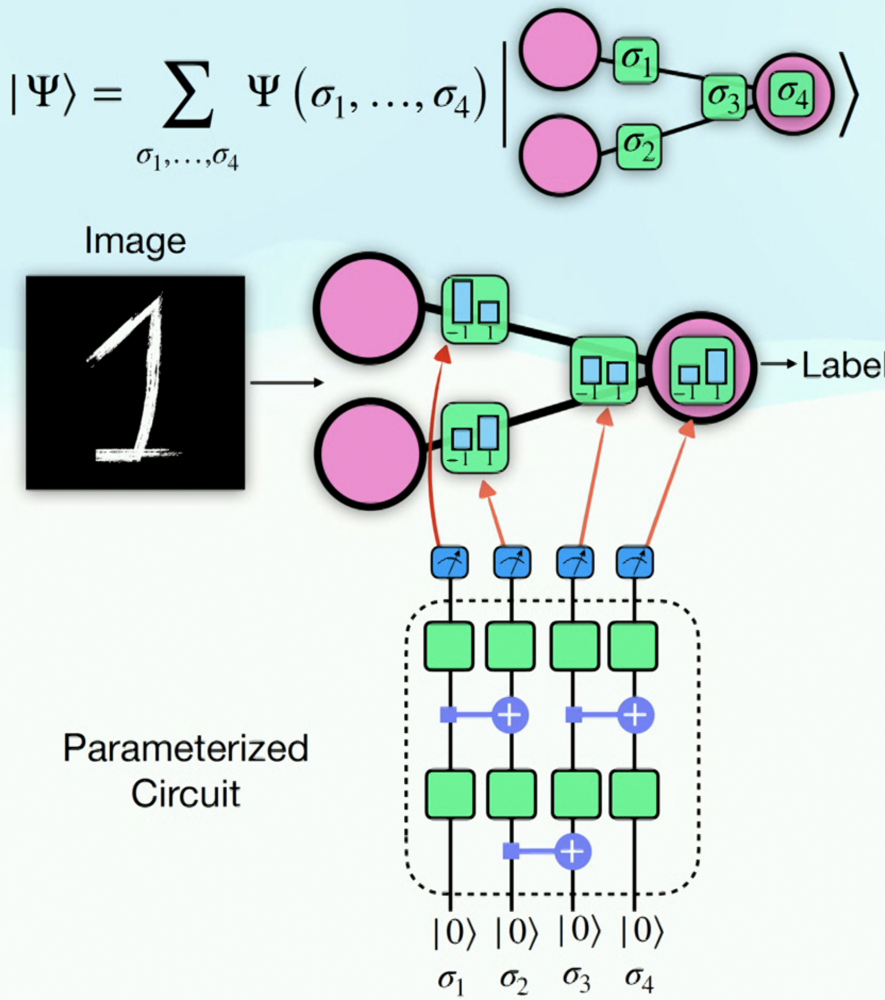




Optimization

Encode the problem in a form suitable to optimization by a variational quantum algorithm

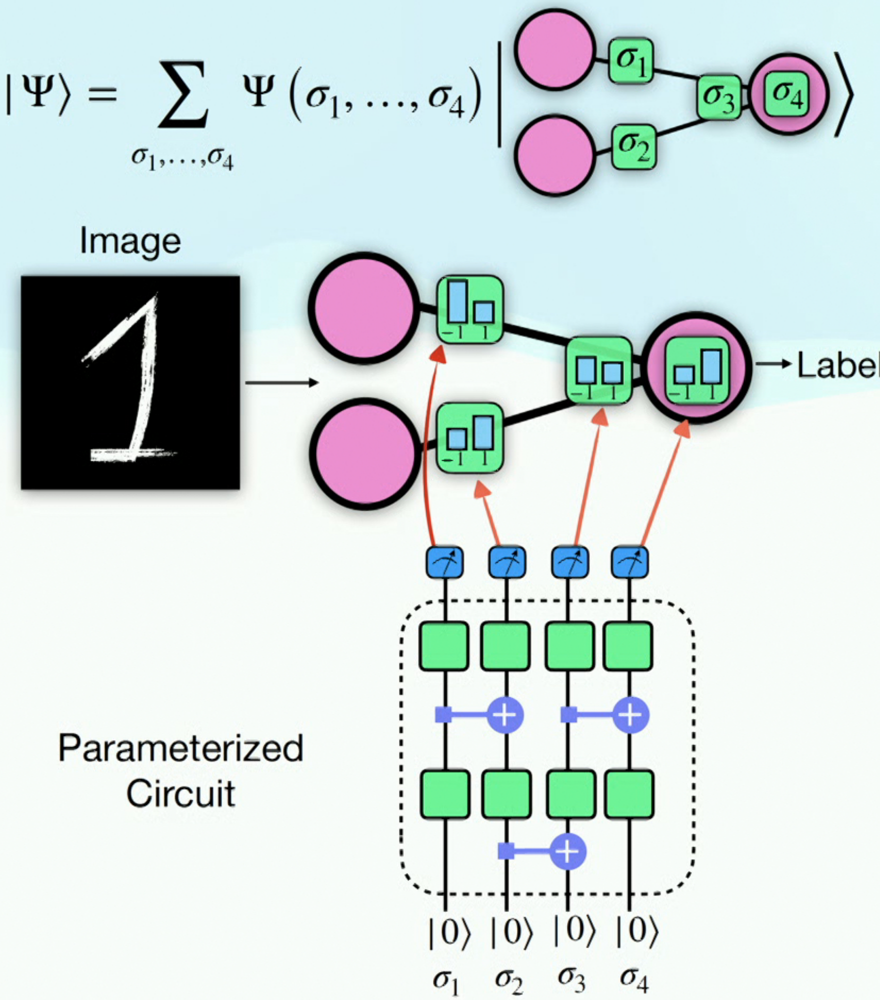
- Use gradient descent to optimize $E(\boldsymbol{\theta})$
- Gradients:
- $$\frac{\partial E\left(\boldsymbol{\theta}\right)}{\partial \theta_{\alpha,j,k}} = \frac{1}{2}\left[E(\boldsymbol{\theta}^{+}_{\alpha,j,k}) - E(\boldsymbol{\theta}^{-}_{\alpha,j,k})\right],$$
- The shifted parameter vector $\boldsymbol{\theta}^{\pm}_{\alpha jk}$ is such that $\theta^{\pm}_{\beta,i,l} = \theta_{\beta,i,l} \pm \frac{\pi}{2}\delta_{\alpha,\beta}\delta_{i,j}\delta_{k,l}$
- Thus, the calculation of the gradient corresponds to the evaluation of a shifted version of the objective function $E(\boldsymbol{\theta})$.



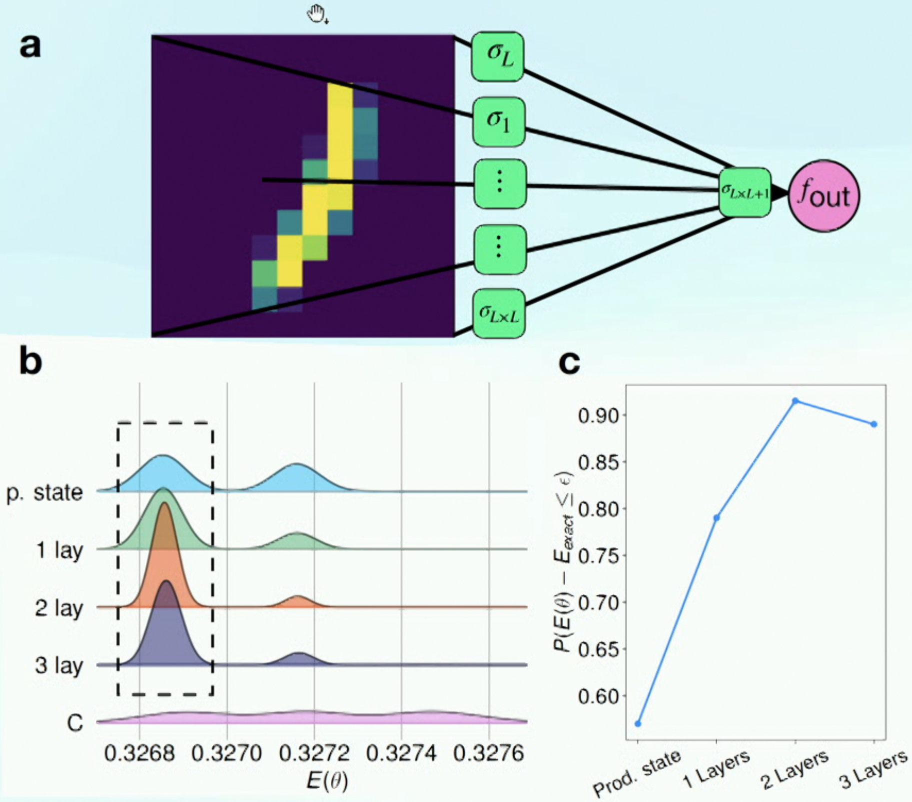
Optimization

Encode the problem in a form suitable to optimization by a variational quantum algorithm

- Use gradient descent to optimize $E(\boldsymbol{\theta})$
- Gradients:
- $$\frac{\partial E(\boldsymbol{\theta})}{\partial \theta_{\alpha,j,k}} = \frac{1}{2}\left[E(\boldsymbol{\theta}^{+}_{\alpha,j,k}) - E(\boldsymbol{\theta}^{-}_{\alpha,j,k})\right],$$
- The shifted parameter vector $\boldsymbol{\theta}^{\pm}_{\alpha jk}$ is such that $\theta^{\pm}_{\beta,i,l} = \theta_{\beta,i,l} \pm \frac{\pi}{2}\delta_{\alpha,\beta}\delta_{i,j}\delta_{k,l}$
- Thus, the calculation of the gradient corresponds to the evaluation of a shifted version of the objective function $E(\boldsymbol{\theta})$.

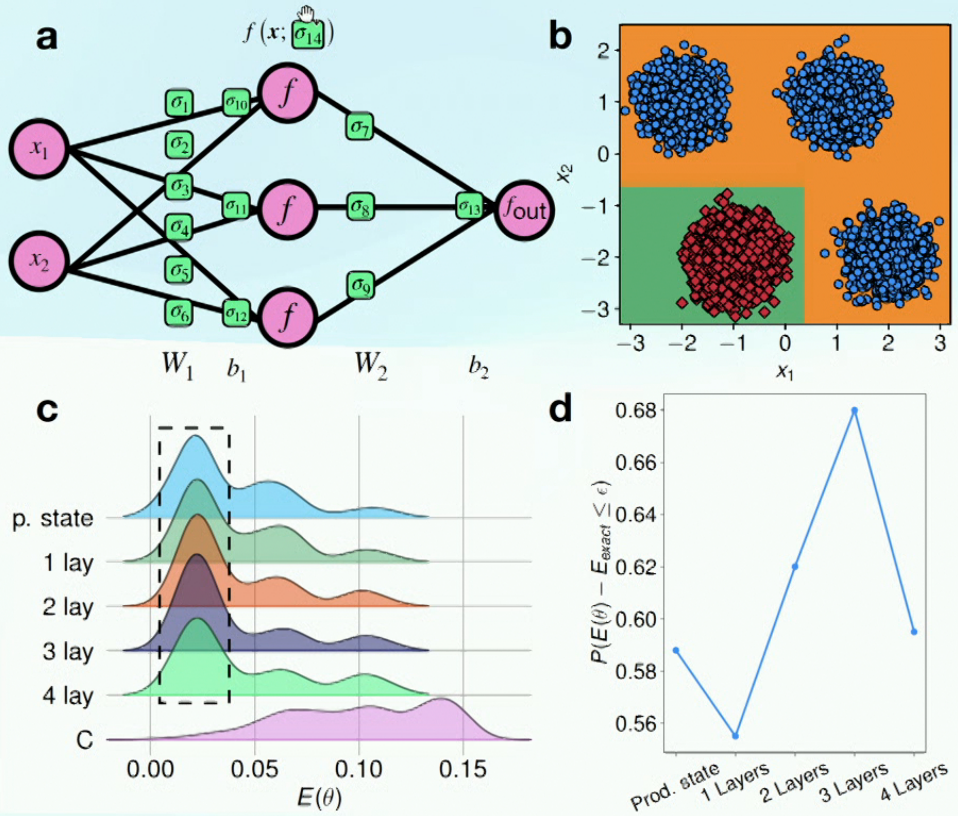


Results: MNIST, binary logistic regression

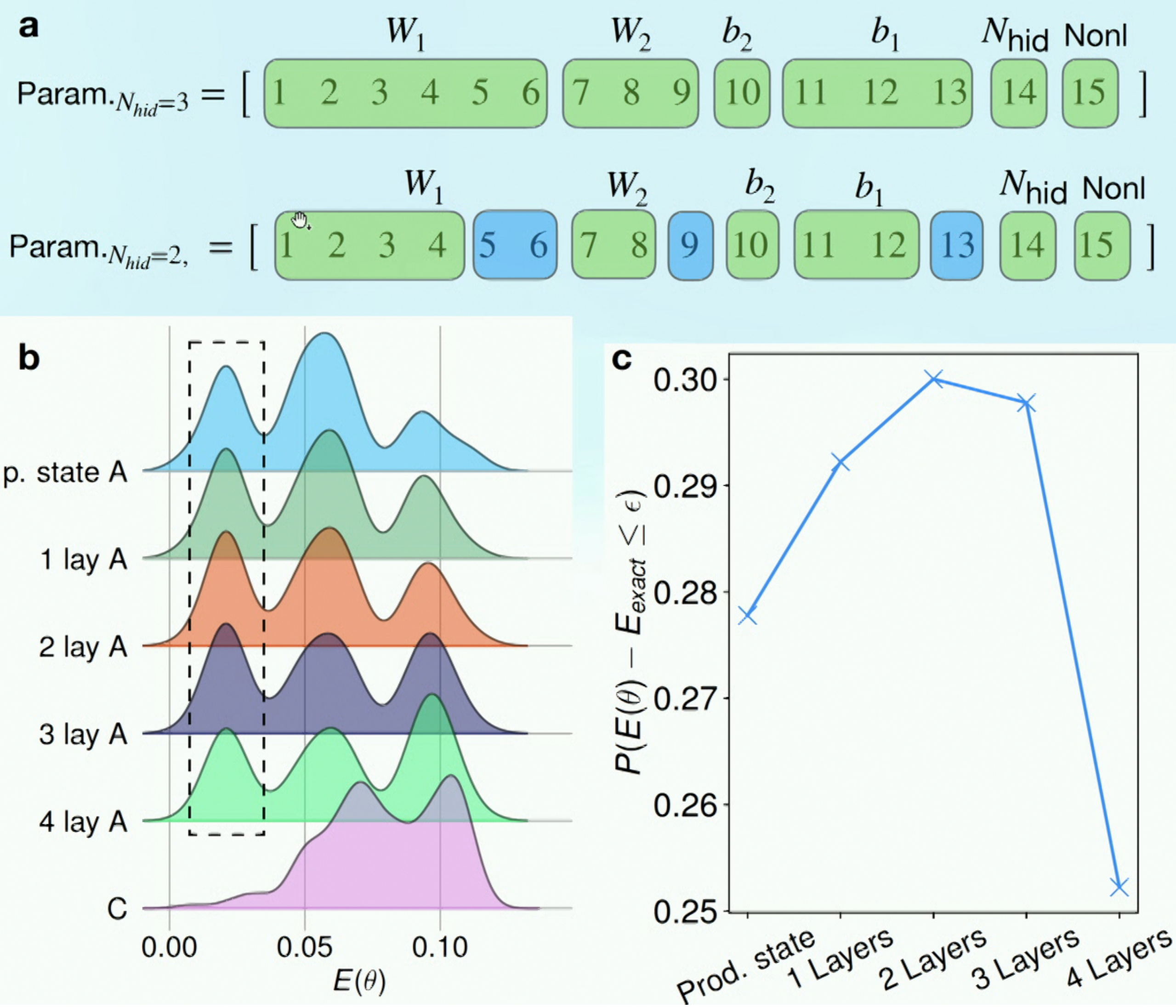


- Train weights and bias.
- Run optimization at least 200 times and evaluate the probabilities of finding an objective function with value $E(\boldsymbol{\theta})$
- Compute Probability that $E(\boldsymbol{\theta})$ is less than ϵ.
- Optimization is successful frequently
- Optimal circuit depth suggests an optimal use of entanglement

Results



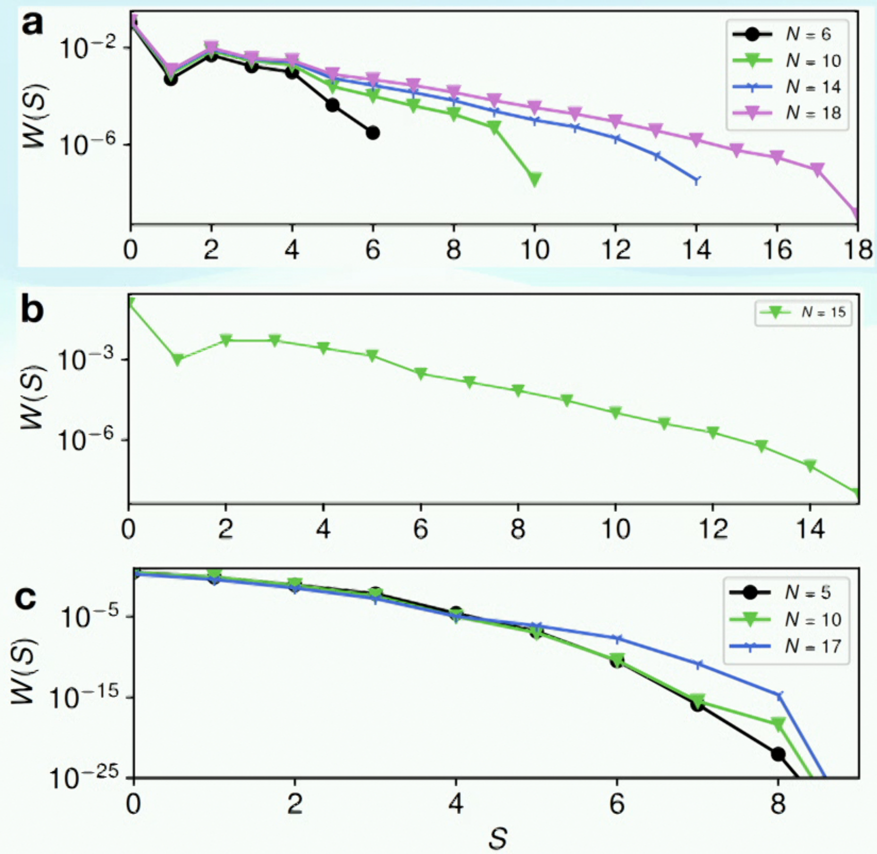
- Train weights + architectural choice of non-linearity.
- Run optimization at least 200 times and evaluate the probabilities of finding an objective function with value $E(\boldsymbol{\theta})$
- Compute Probability that $E(\boldsymbol{\theta})$ is less than
- Optimization is successful frequently
- Optimal circuit depth— optimal use of entanglement



- Train weights + architectural choice of non-linearity + hidden dimension (2 or 3, binary choice)
- Optimal circuit depth
- Success probability a bit smaller
- But overall successful optimization

Fourier Analysis

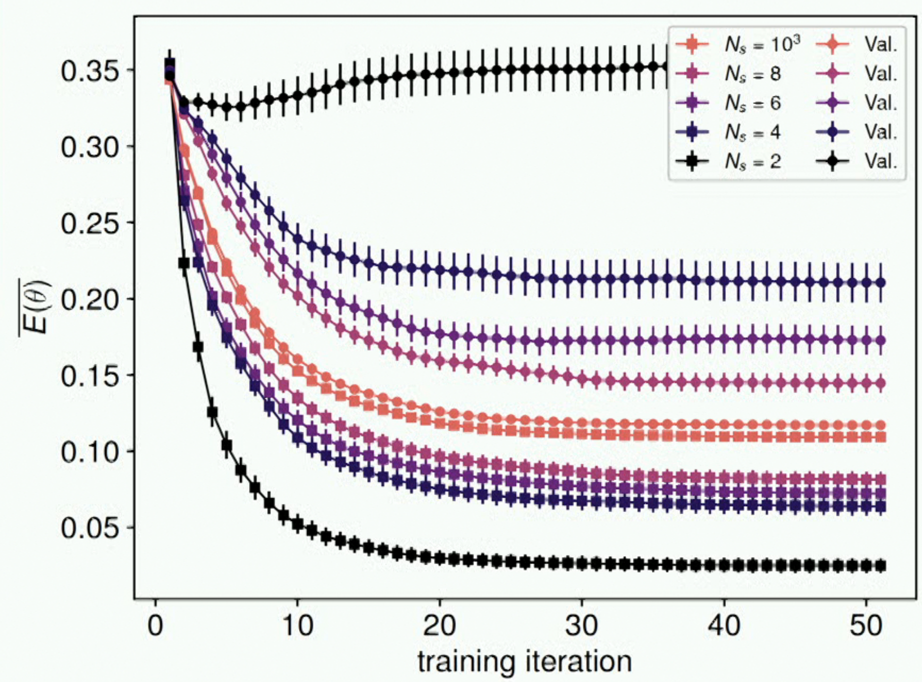
- $$\hat{C} = \sum_{\hat{\sigma}_1,\ldots\hat{\sigma}_N} f(\hat{\sigma}_1,\ldots\hat{\sigma}_N) \bigotimes_{i=1}^{N} \hat{\sigma}_i$$
- Effective Ising model with multi-variable all-to-all interactions
- Fourier coefficients are given by $f(\hat{\sigma}_1,\ldots\hat{\sigma}_N) = \frac{1}{2^N}\mathrm{Tr}\left[\hat{C}\otimes_{i=1}^{N}\hat{\sigma}_i\right] \in \mathbb{R}$
- $$W(S) = \sum_{\hat{\sigma}_1,\ldots\hat{\sigma}_N} |f(\hat{\sigma}_1,\ldots\hat{\sigma}_N)|^2 \delta_{S,S(\hat{\sigma}_1,\ldots\hat{\sigma}_N)}.$$



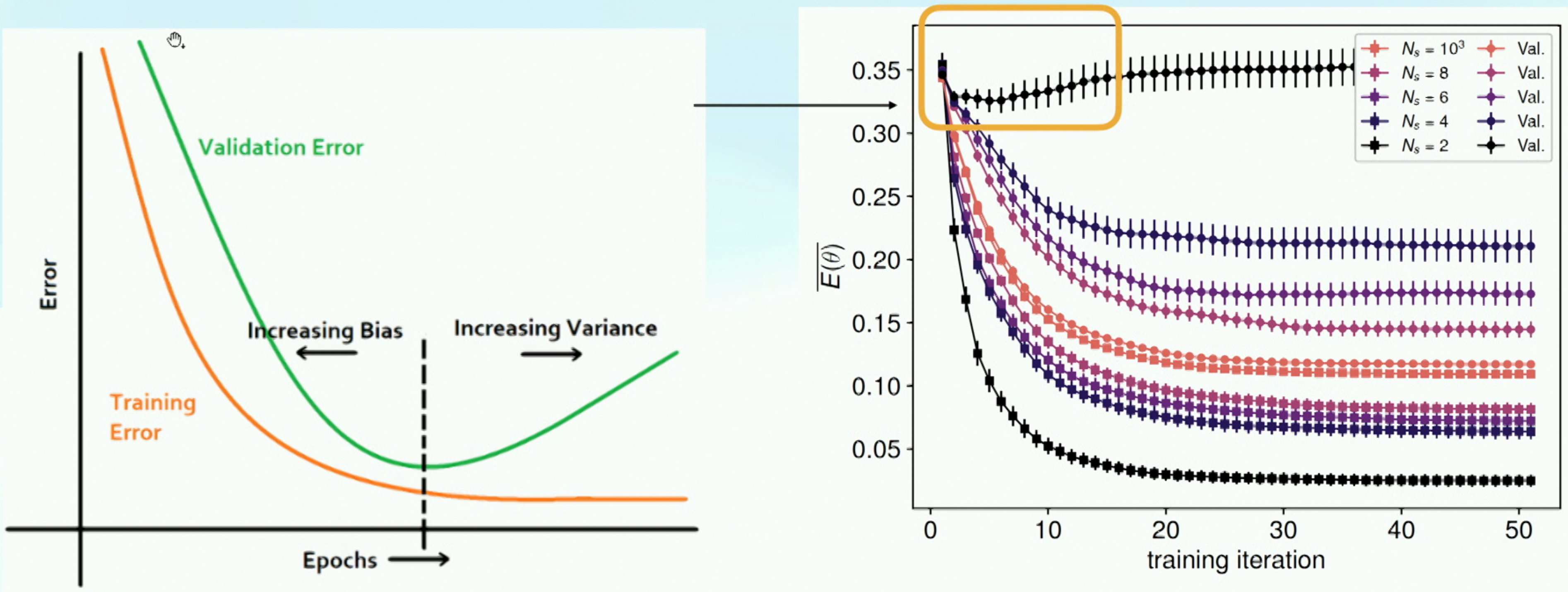
“Augmented” model selection

- Augmented model encapsulating the parameters, hyperparameters and architecture of a neural network which we jointly optimize on a training dataset. How to choose model using a validation set?

- The data suggests that these augmented models behave like traditional statistical models which follow the usual bias-variance decomposition.



“Augmented” model selection



https://medium.com/@rahuljain13101999/why-early-stopping-works-as-regularization-b9f0a6c2772

Conclusions

- We have introduced HyperNetworks which train binary neural networks in quantum superposition
- One optimization loop trains parameters, hyperparameters, and architectural choices in binary neural networks
- Quantum computers are currently reaching the ability to vastly outperform supercomputers' energy efficiency by many orders of magnitude over classical computers.
- Binary neural networks save energy at inference time. We are suggesting is that we can potentially save energy in training, architectural design and hyperparameter search.
- Neural networks perform best when they are large—need better encoding of the problem
- Quantum annealers.