Uncertainty Quantification and Deep Learning

Elise Jennings

ALCF Data Science group

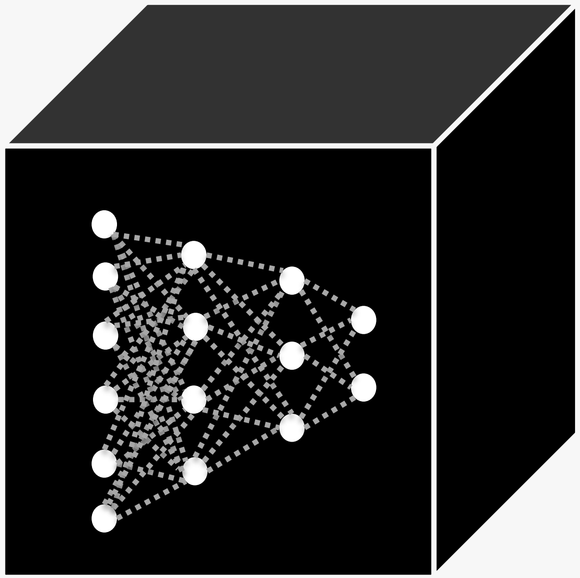
ejennings@anl.gov

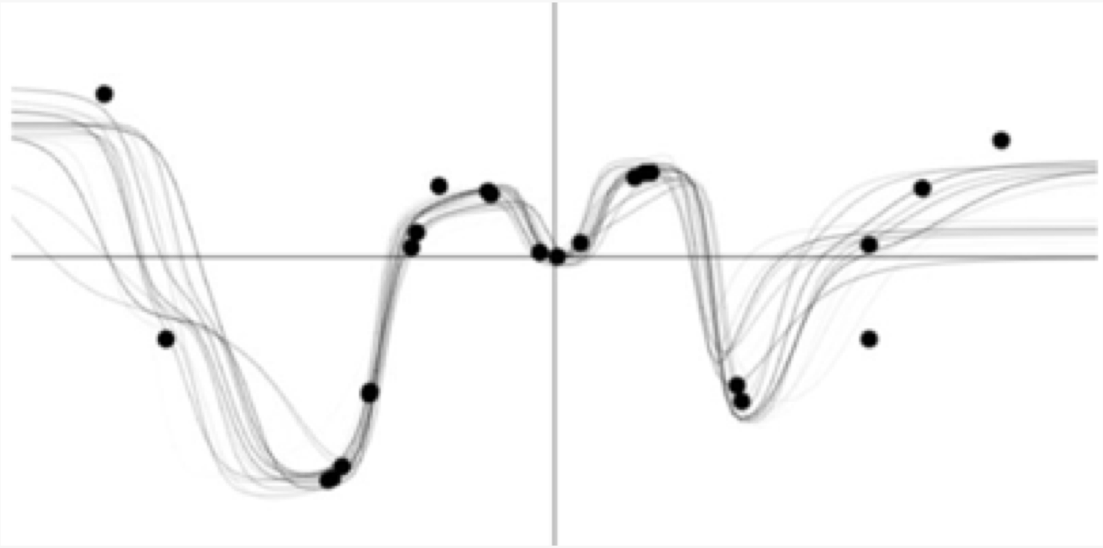
Argonne NATIONAL LABORATORY

Uncertainty Quantification

Neural Networks:

- Black Box
- How do we know if new model is making sensible predictions or guessing at random?
- Model or statistical errors help explain failure to generalize
- DI often criticized for lack of robustness, interpretability, reliability





Understanding what a model does not know is a critical part of any scientific analysis

Neural Networks

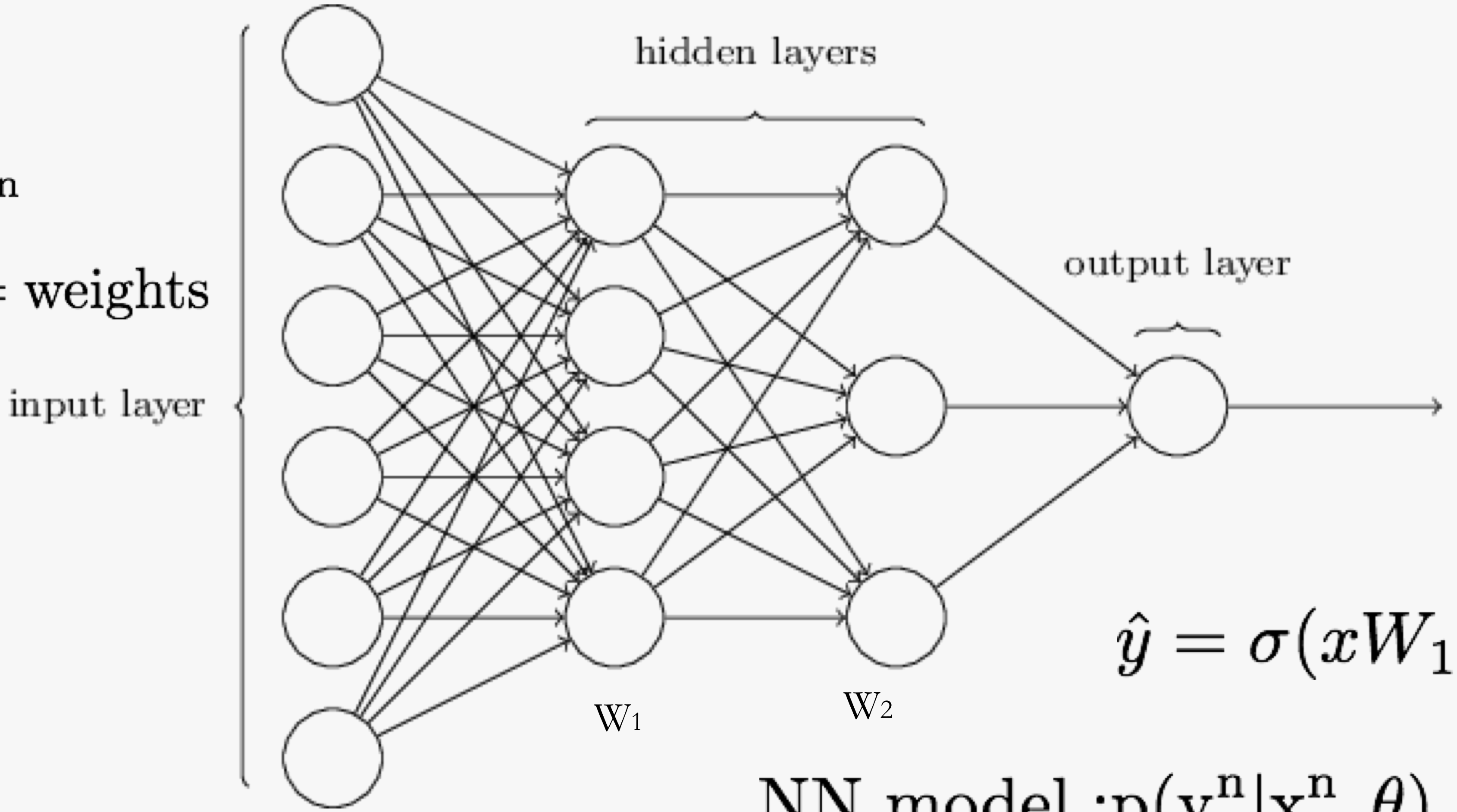
A Neural Network

represents a function with many parameters &

is recursive application of weighted linear functions followed by non-linear functions

$$\text{Data}: D = x^n, y^n$$

$$\text{Parameters}: \theta = \text{weights}$$



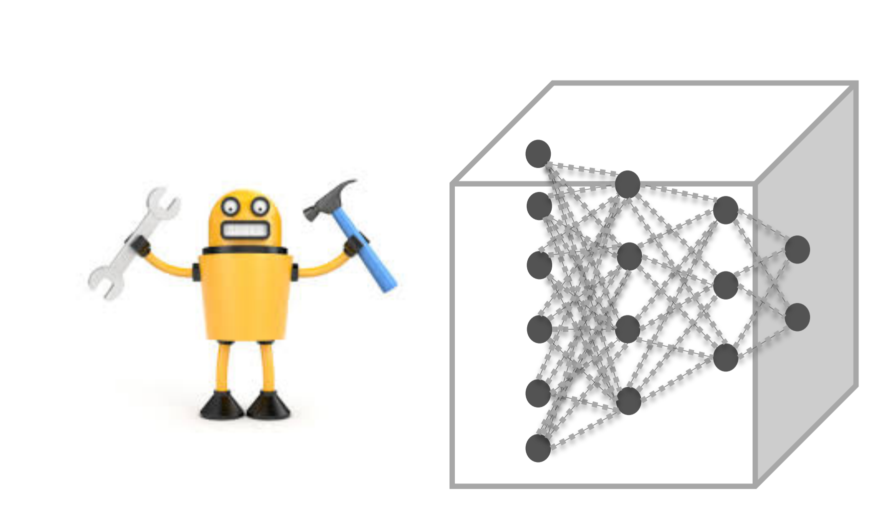
$$\hat{y} = \sigma(xW_1 + b)W_2$$

$$\text{NN model}: p(y^n | x^n, \theta)$$

Why use Bayesian methods in Deep Learning?

Drawback to DL:

- Many hyperparameters require specific tuning, with large datasets finding the optimal set can take a long time
- NN's trained with BP obtain point estimates of the weights in the network
- No uncertainty in these point estimates: very important for e.g. medical diagnosis, finance, self driving cars etc.
- Common to use large NN to fit data & use regularization to try to prevent overfitting
- Need efficient search algorithms/guess work to find best network architecture



Explaining why a model fails…

Softmax gives probabilities for each class but not the uncertainty in the model

Deep Neural Networks are Easily Fooled:
High Confidence Predictions for Unrecognizable Images

Anh Nguyen
University of Wyoming
anguyen8@uwyo.edu

Jason Yosinski
Cornell University
yosinski@cs.cornell.edu

Jeff Clune
University of Wyoming
jeffclune@uwyo.edu

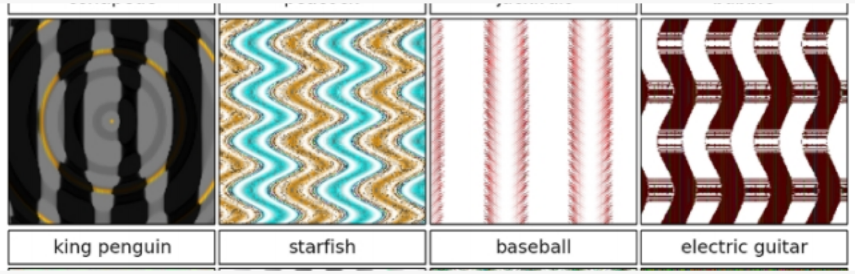


Figure 1. Evolved images that are unrecognizable to humans, but that state-of-the-art DNNs trained on ImageNet believe with $\geq$ 99.6% certainty to be a familiar object. This result highlights differences between how DNNs and humans recognize objects.

CIFAR-100's *apple* misclassified as CIFAR-10's *frog* class with $p > 0.9$.

https://hjweide.github.io/quantifying-uncertainty-in-neural-networks

What are Bayesian Neural Networks?

- Think of training the network as inference problem which we solve using Bayes' Thm.

$$p(\theta|\mathbf{D}) = \frac{\mathcal{L}(\mathbf{D}|\theta)\pi(\theta)}{\mathbf{p}(\mathbf{D})}$$

What are Bayesian Neural Networks?

- Think of training the network as inference problem which we solve using Bayes' Thm.

$$p(\theta|\mathbf{D}) = \frac{\mathcal{L}(\mathbf{D}|\theta)\pi(\theta)}{\mathbf{p}(\mathbf{D})}$$

- A Bayesian Neural Network is a Neural Network with distributions over weights and biases. The loss which we are trying to minimize is the Posterior Distribution.
- We find a weighted average over all parameters which can be thought of as an infinite ensemble of neural networks.

What are Bayesian Neural Networks?

- Think of training the network as inference problem which we solve using Bayes' Thm.

$$p(\theta|\mathbf{D}) = \frac{\mathcal{L}(\mathbf{D}|\theta)\pi(\theta)}{\mathbf{p}(\mathbf{D})}$$

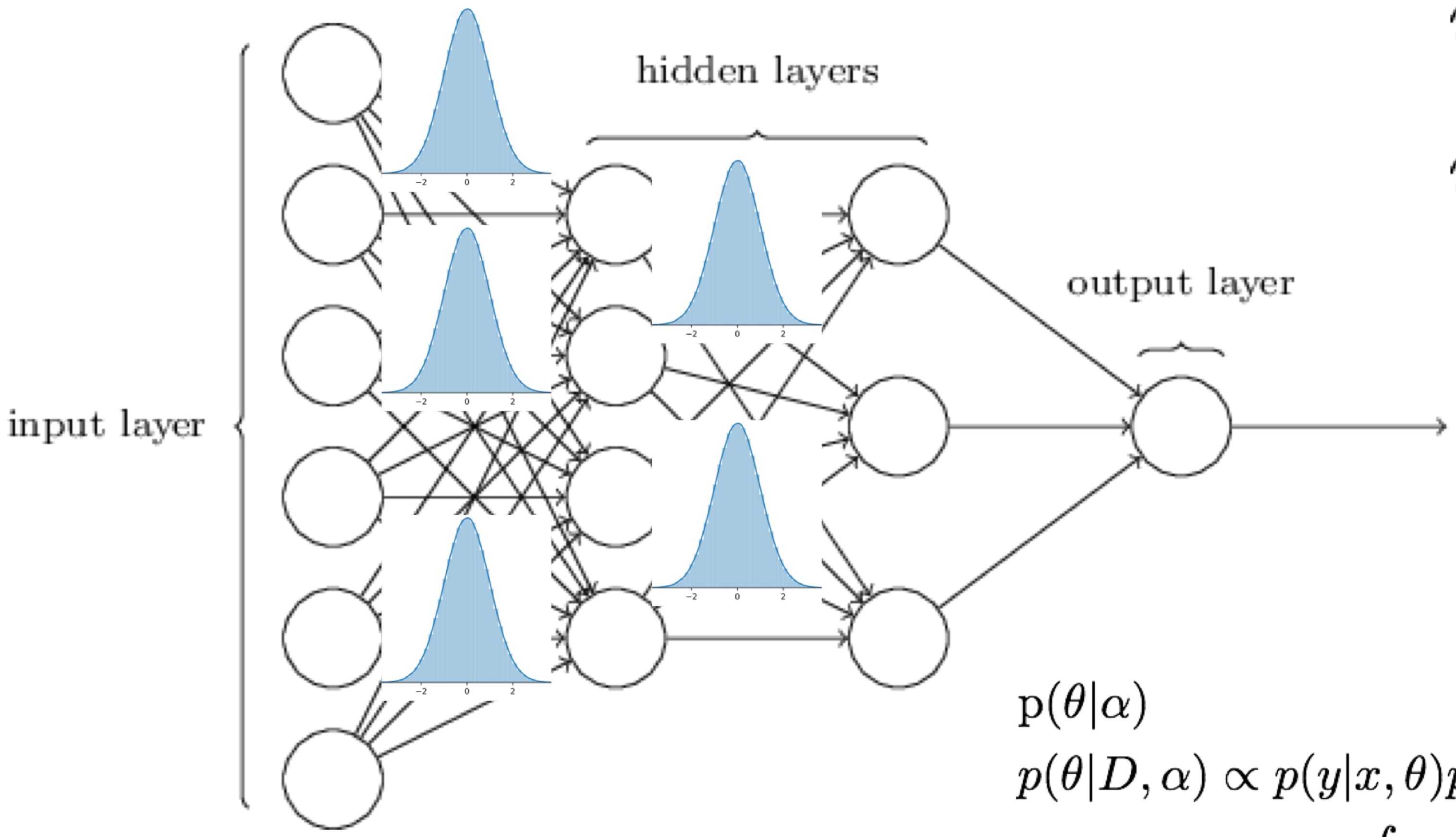
- A Bayesian Neural Network is a Neural Network with distributions over weights and biases. The loss which we are trying to minimize is the Posterior Distribution.

- We find a weighted average over all parameters which can be thought of as an infinite ensemble of neural networks.

- Neal 1995 (& Williams 1997, Lee et al 2018 Google Brain…)

A single layer infinitely wide nn with distributions over weights = A Gaussian process

Bayesian Neural Networks



$$w_i^1 \sim \mathcal{N}(0, \epsilon^1)$$
$$w_i^2 \sim \mathcal{N}(0, \epsilon^2)$$

$\mathrm{p}(\theta|\alpha)$ prior

$p(\theta|D,\alpha) \propto p(y|x,\theta)p(\theta|\alpha)$ posterior

$p(y'|D,x',\alpha) = \int p(y'|x',\theta)p(\theta|D,\alpha)\mathrm{d}\alpha$ prediction

Many inference methods to approximately solve for this posterior

Practical Variational Inference for Neural Networks

Alex Graves
Department of Computer Science
University of Toronto, Canada
graves@cs.toronto.edu

Abstract

Variational methods have been previously explored as a tractable approximation to Bayesian inference for neural networks. However the approaches proposed so far have only been applicable to a few simple network architectures. This paper introduces an easy-to-implement stochastic variational method (or equivalently, minimum description length loss function) that can be applied to most neural networks. Along the way it revisits several common regularisers from a variational perspective. It also provides a simple pruning heuristic that can both drastically reduce the number of network weights and lead to improved generalisation. Experimental results are provided for a hierarchical multidimensional recurrent neural network applied to the TIMIT speech corpus.

MCMC methods
Gibbs sampling
Hamiltonian MC
Variational Inference

Classes

`class BiGANInference` : Adversarially Learned Inference (Dumoulin et al., 2017) or

`class GANInference` : Parameter estimation with GAN-style training

`class Gibbs` : Gibbs sampling (Geman & Geman, 1984).

`class HMC` : Hamiltonian Monte Carlo, also known as hybrid Monte Carlo

`class ImplicitKLqp` : Variational inference with implicit probabilistic models

`class Inference` : Abstract base class for inference. All inference algorithms in

`class KLpq` : Variational inference with the KL divergence

`class KLqp` : Variational inference with the KL divergence

`class Laplace` : Laplace approximation (Laplace, 1986).

`class MAP` : Maximum a posteriori.

`class MetropolisHastings` : Metropolis-Hastings (Hastings, 1970; Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953).

`class MonteCarlo` : Abstract base class for Monte Carlo. Specific Monte Carlo methods

`class ReparameterizationEntropyKLqp` : Variational inference with the KL divergence

`class ReparameterizationKLKLqp` : Variational inference with the KL divergence

`class ReparameterizationKLqp` : Variational inference with the KL divergence

`class SGHMC` : Stochastic gradient Hamiltonian Monte Carlo (Chen, Fox, & Guestrin, 2014).

`class SGLD` : Stochastic gradient Langevin dynamics (Welling & Teh, 2011).

`class ScoreEntropyKLqp` : Variational inference with the KL divergence

`class ScoreKLKLqp` : Variational inference with the KL divergence

Bayesian approach

- Marginalization over hyperparameter
- Naturally account for uncertainty
- More robust to overfitting as average rather than point estimate used
- L1/L2 regularization = choice of prior for weights
- Model comparison via Bayesian Evidence

A Practical Bayesian Framework for Backprop Networks

David J.C. MacKay
Computation and Neural Systems*
California Institute of Technology 139–74
Pasadena CA 91125
mackay@hope.caltech.edu

Abstract

A quantitative and practical Bayesian framework is described for learning of mappings in feedforward networks. The framework makes possible: (1) objective comparisons between solutions using alternative network architectures; (2) objective stopping rules for network pruning or growing procedures; (3) objective choice of magnitude and type of weight decay terms or additive regularisers (for penalising large weights, etc.); (4) a measure of the effective number of well–determined parameters in a model; (5) quantified estimates of the error bars on network parameters and on network output; (6) objective comparisons with alternative learning and interpolation models such as splines and radial basis functions. The Bayesian 'evidence' automatically embodies 'Occam's razor,' penalising over–flexible and over–complex models. The Bayesian approach helps detect poor underlying assumptions in learning models. For learning models well matched to a problem, a good correlation between generalisation ability and the Bayesian evidence is obtained.

Bayesian approach

- Marginalization over hyperparameter
- Naturally account for uncertainty
- More robust to overfitting as average rather than point estimate used
- L1/L2 regularization = choice of prior for weights
- Model comparison via Bayesian Evidence

A Practical Bayesian Framework for Backprop Networks

David J.C. MacKay
Computation and Neural Systems*
California Institute of Technology 139–74
Pasadena CA 91125
mackay@hope.caltech.edu

Abstract

A quantitative and practical Bayesian framework is described for learning of mappings in feedforward networks. The framework makes possible: (1) objective comparisons between solutions using alternative network architectures; (2) objective stopping rules for network pruning or growing procedures; (3) objective choice of magnitude and type of weight decay terms or additive regularisers (for penalising large weights, etc.); (4) a measure of the effective number of well–determined parameters in a model; (5) quantified estimates of the error bars on network parameters and on network output; (6) objective comparisons with alternative learning and interpolation models such as splines and radial basis functions. The Bayesian 'evidence' automatically embodies 'Occam's razor,' penalising over–flexible and over–complex models. The Bayesian approach helps detect poor underlying assumptions in learning models. For learning models well matched to a problem, a good correlation between generalisation ability and the Bayesian evidence is obtained.

But how well do they scale… ??

PyMC3

Stan

- Slow in high dim
- Approximate solution to exact posterior

Tensorflow Probability & Edward (Tran et al 2016)

- Variational inference: finds exact solution to approx. posterior

ZhuSuan (Shi et al 2017)

SKPro machine learning toolbox (Gressman et al 2018)

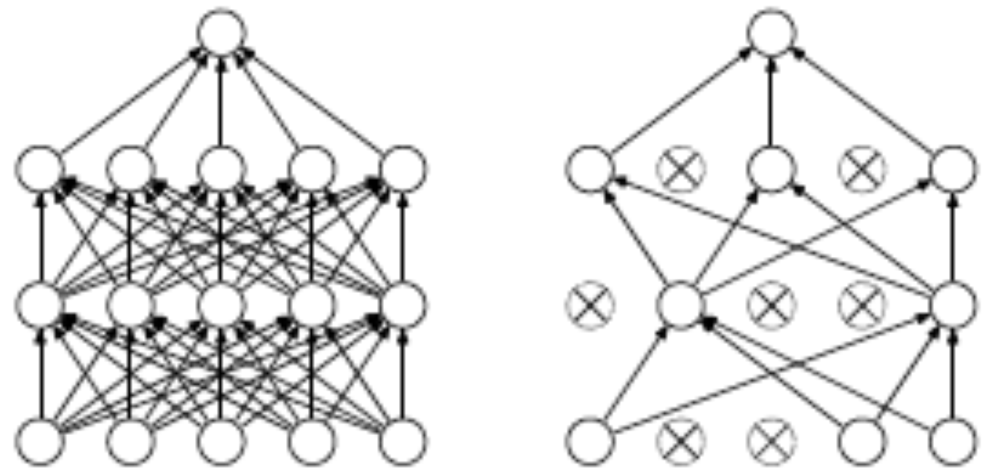
Pomegranate (Schreiber 2017)

Oracle Labs Augur (Tristan et al 2014) - 1,000 GPUs

Uncertainty Quantification – no extra cost

Dropout

- Prob p to drop weights from network at training time
- Avoids overfitting as it prevents units co-adapting



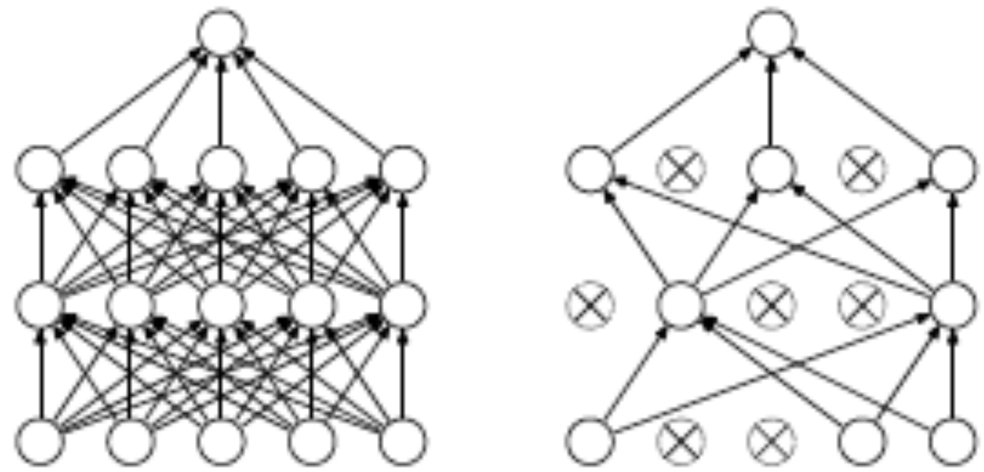
Uncertainty Quantification – no extra cost

$$\hat{y} = \sigma(xb_1W_1 + b)b_2W_2$$

$$b_i \sim \mathrm{Bernoulli}(p_i)$$

Dropout

- Prob p to drop weights from network at training time
- Avoids overfitting as it prevents units co-adapting

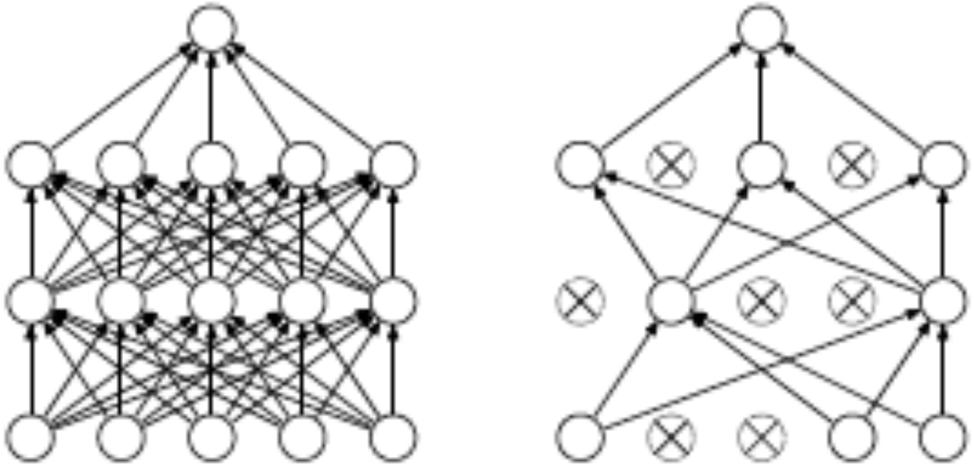


Uncertainty Quantification – no extra cost

$$\hat{y} = \sigma(xb_1W_1 + b)b_2W_2$$
$$b_i \sim \text{Bernoulli}(p_i)$$

Dropout

- Prob p to drop weights from network at training time
- Avoids overfitting as it prevents units co-adapting
- A dropout network is simply a Gaussian process approximation
- Srivastava et al 2014: Optimal p=0.8 input layers, 0.5 hidden layers

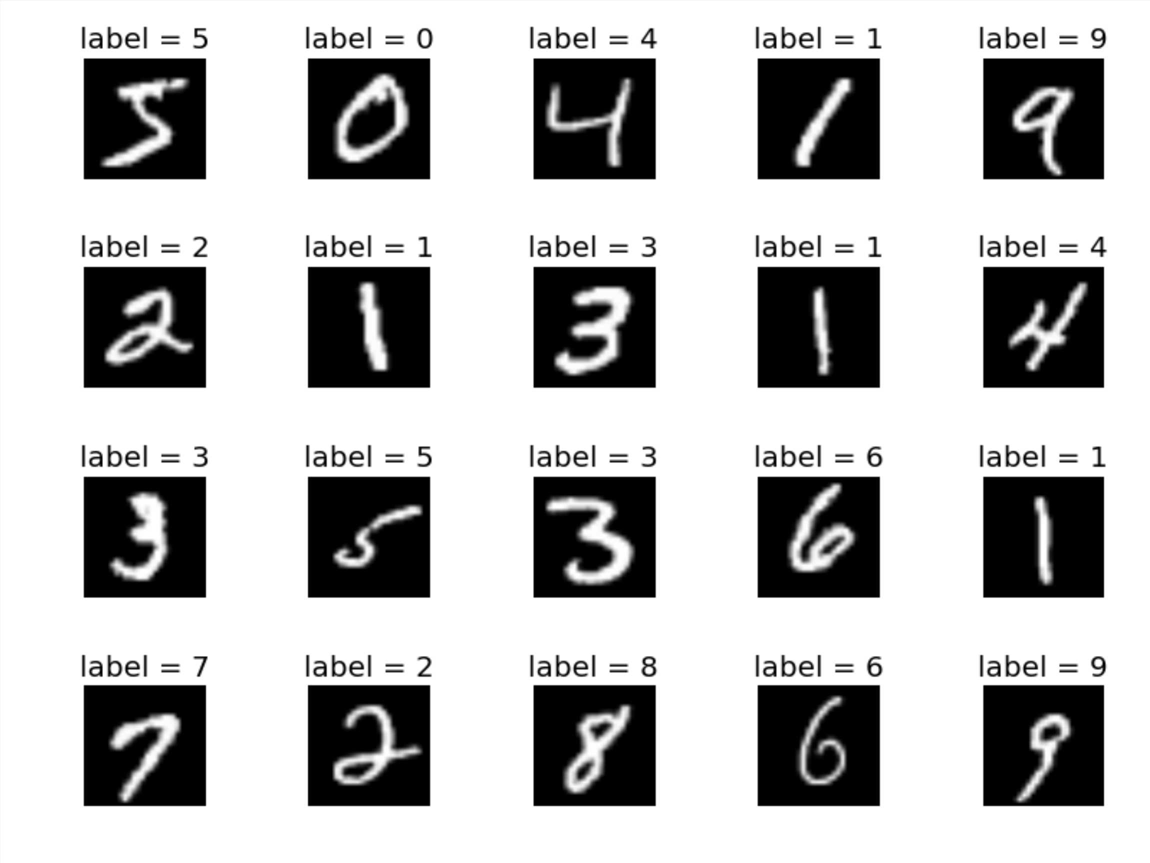


How does dropout compare to Bayesian Neural Networks?

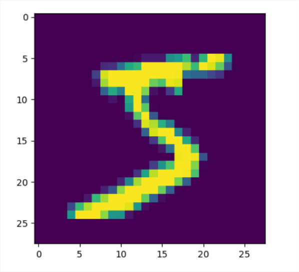
- Dropout can be interpreted as averaging exponentially many models with shared weights
- Each model is equally weighted
- Faster to use at train and test time
- Tune hyper parameters

- Bayesian nn is the proper way of averaging over the space of nn structures and parameters
- Each model is weighted taking into account priors and how well model fits data
- Can be slow to train, difficult to scale
- Marginalize over hyperparameters

Example: MNIST database of handwritten digits

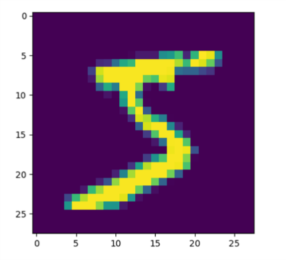


Example: MNIST database of handwritten digits

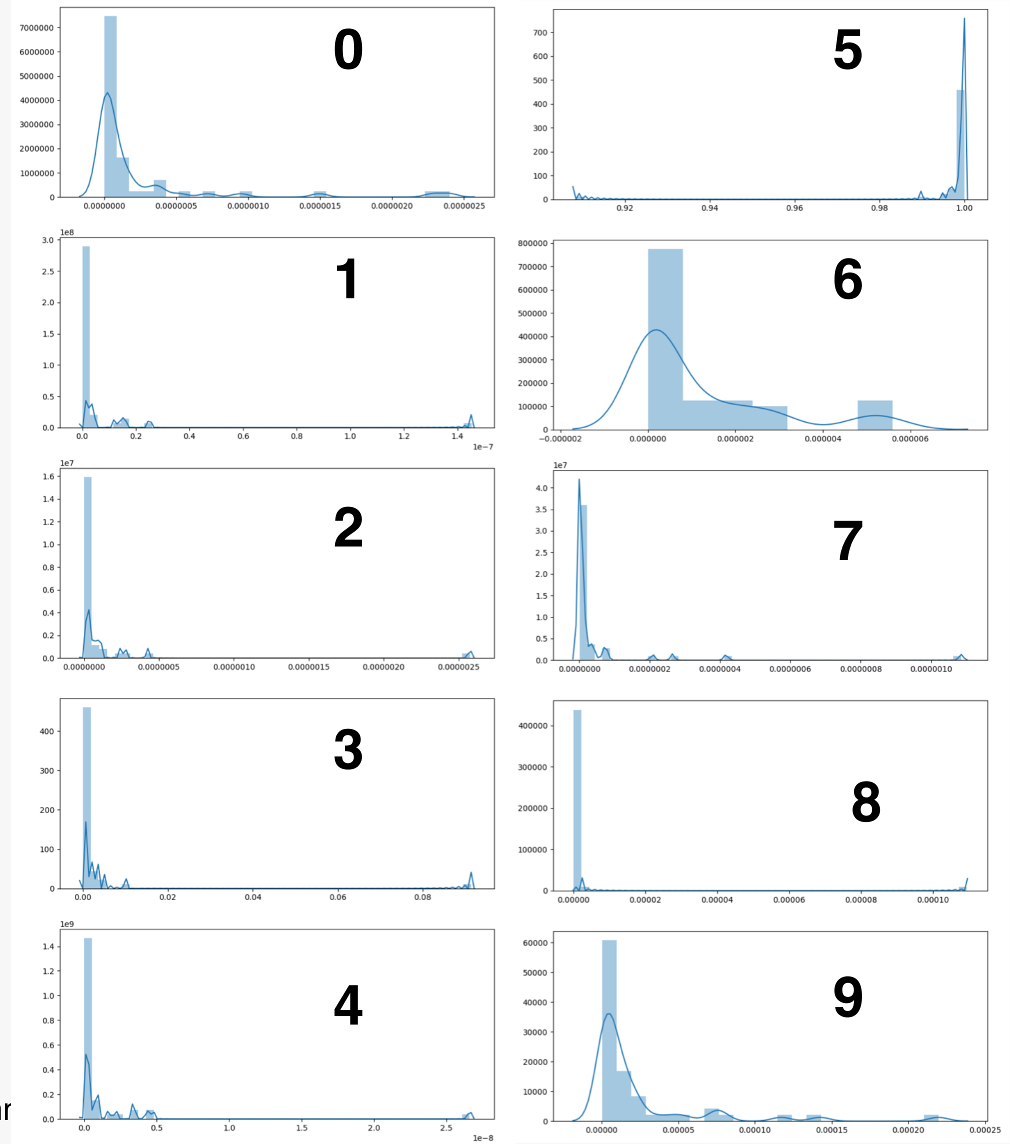


3 or 5 ?

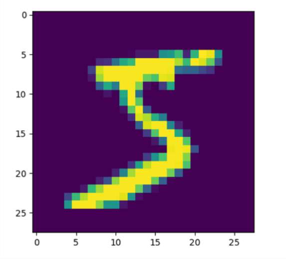
BNN results:



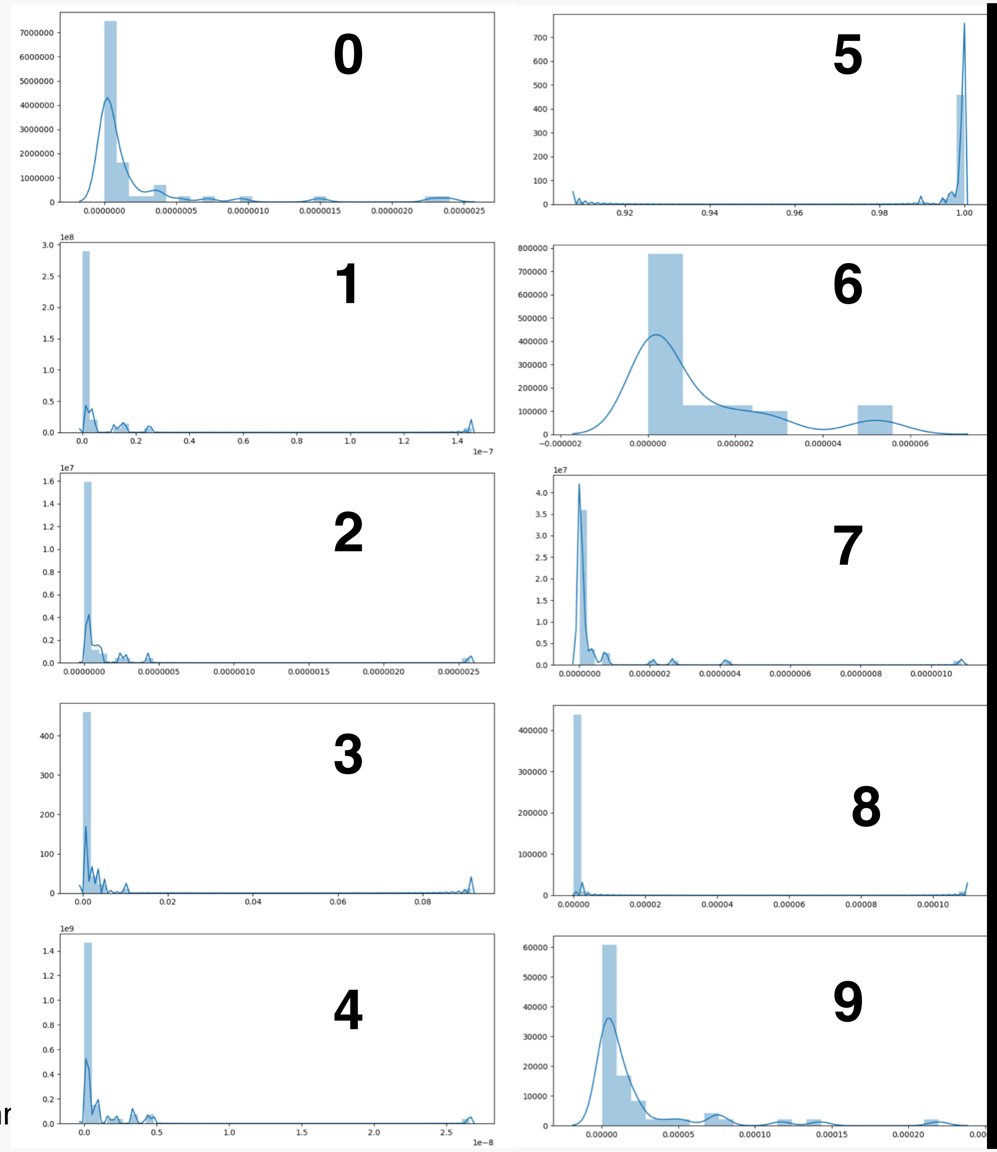
Iter:400



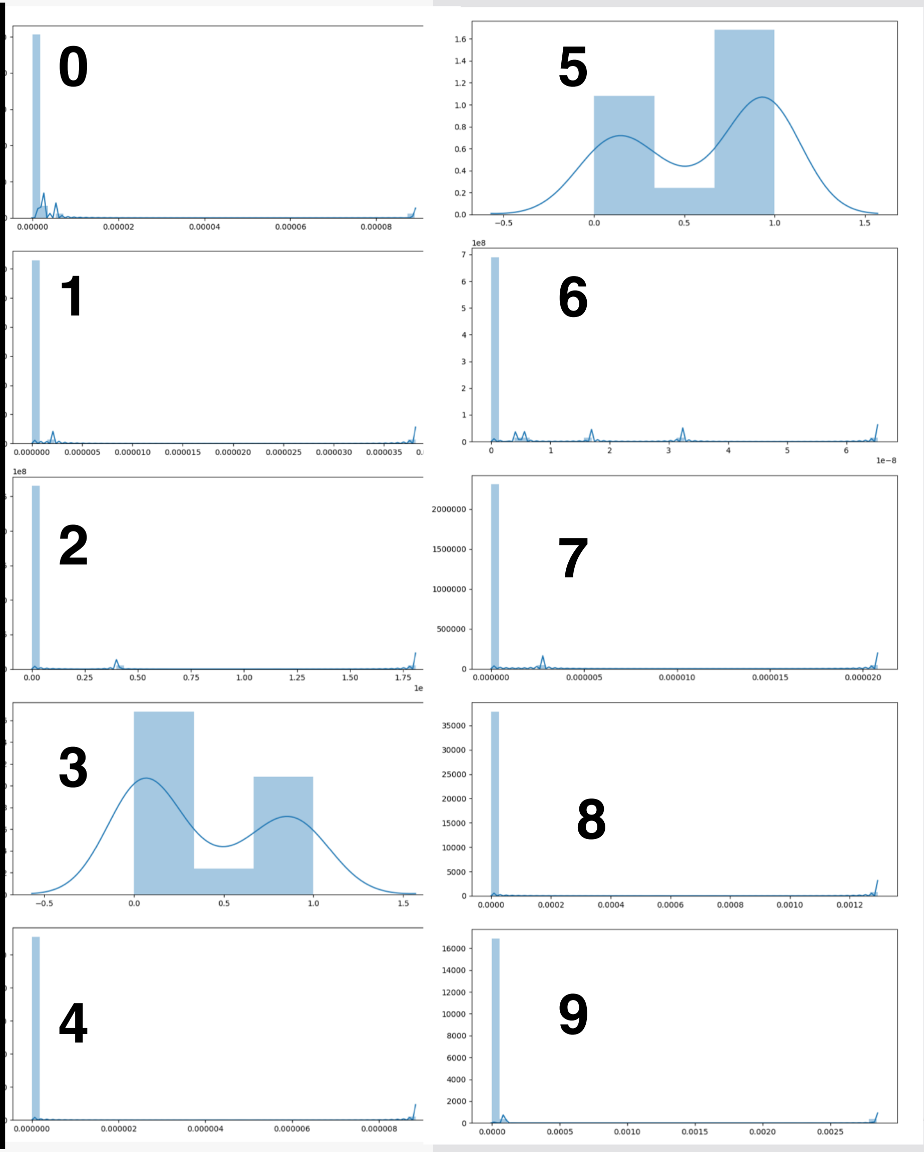
BNN results:



Iter:400

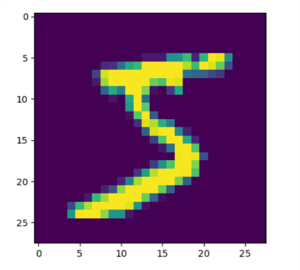


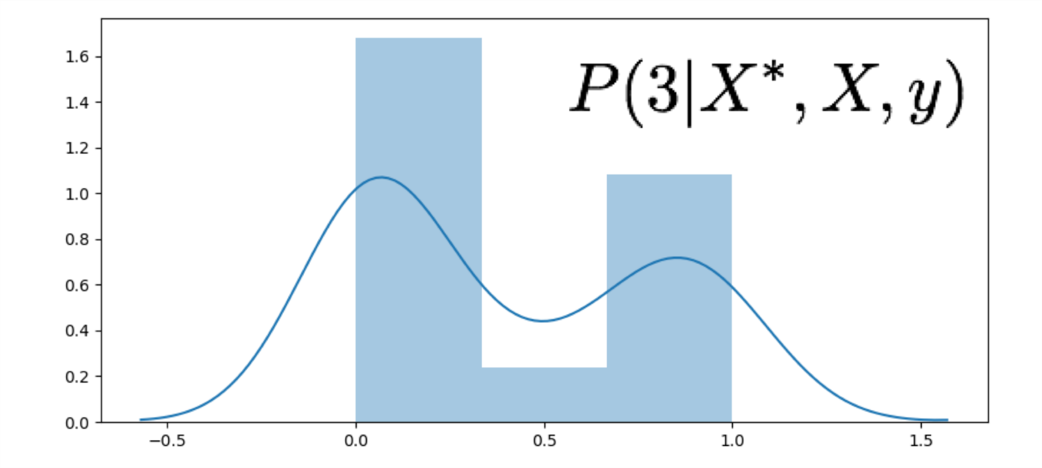
Iter:6000

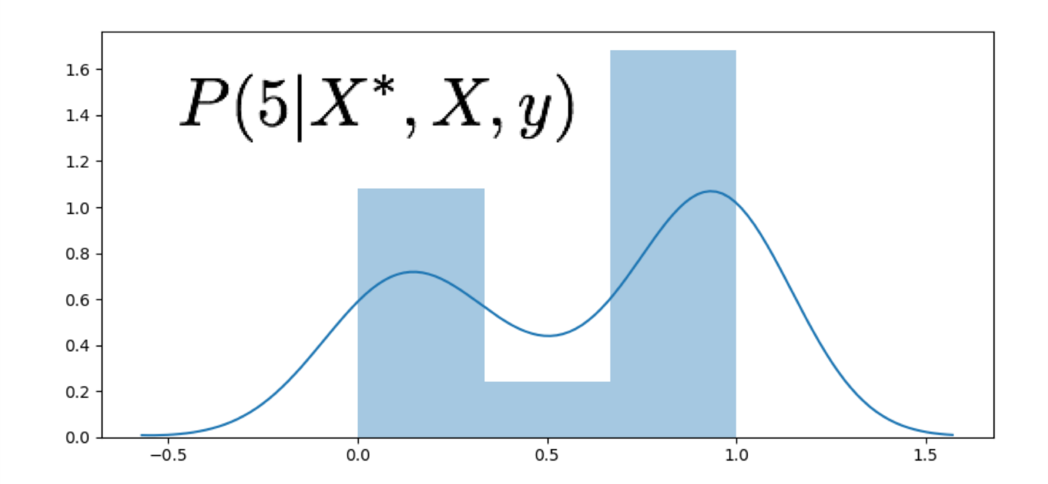


MNIST results:

Distribution of Predictive samples

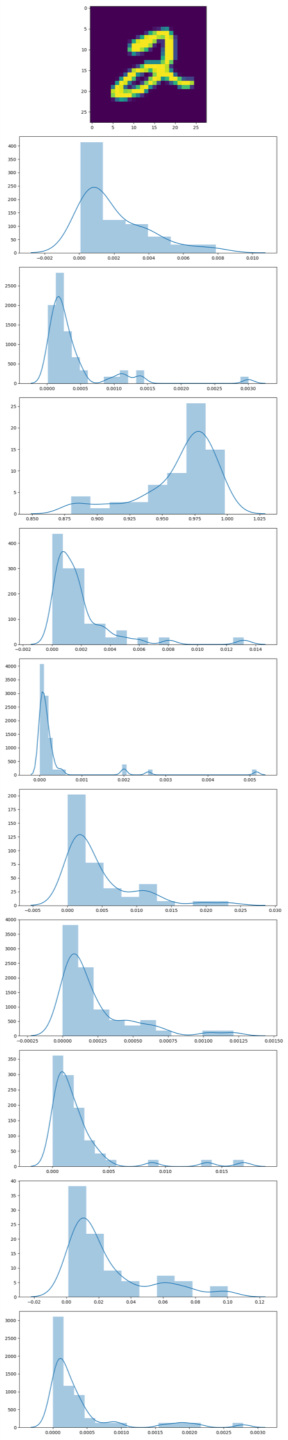




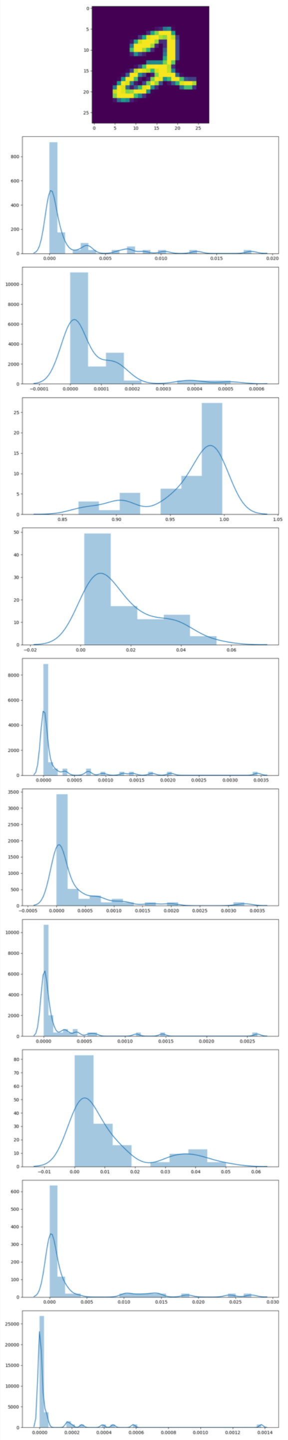


BNN results:

Iter:400

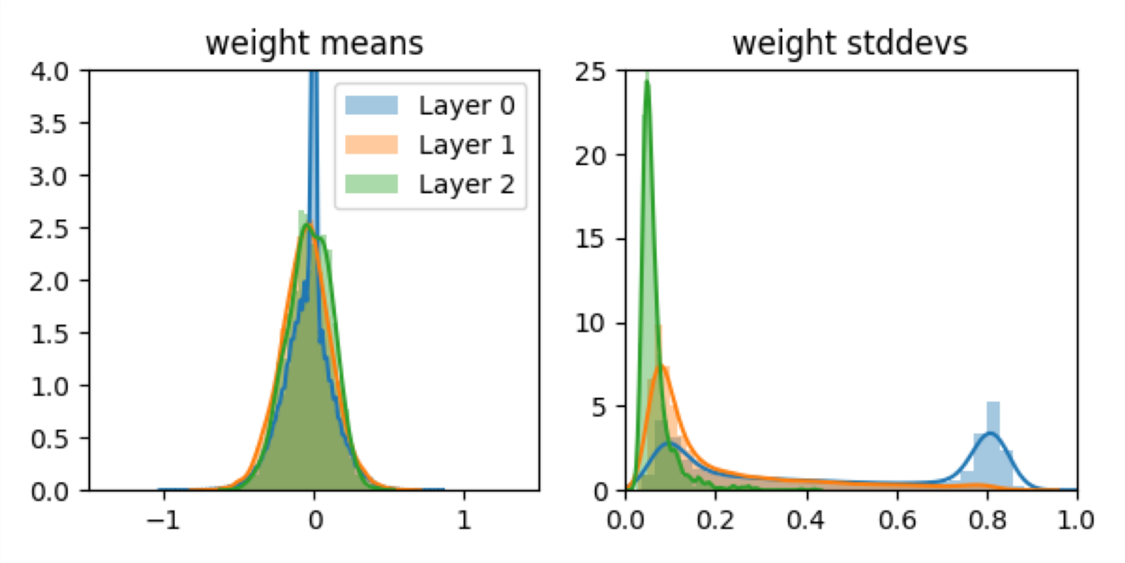


Iter:6000

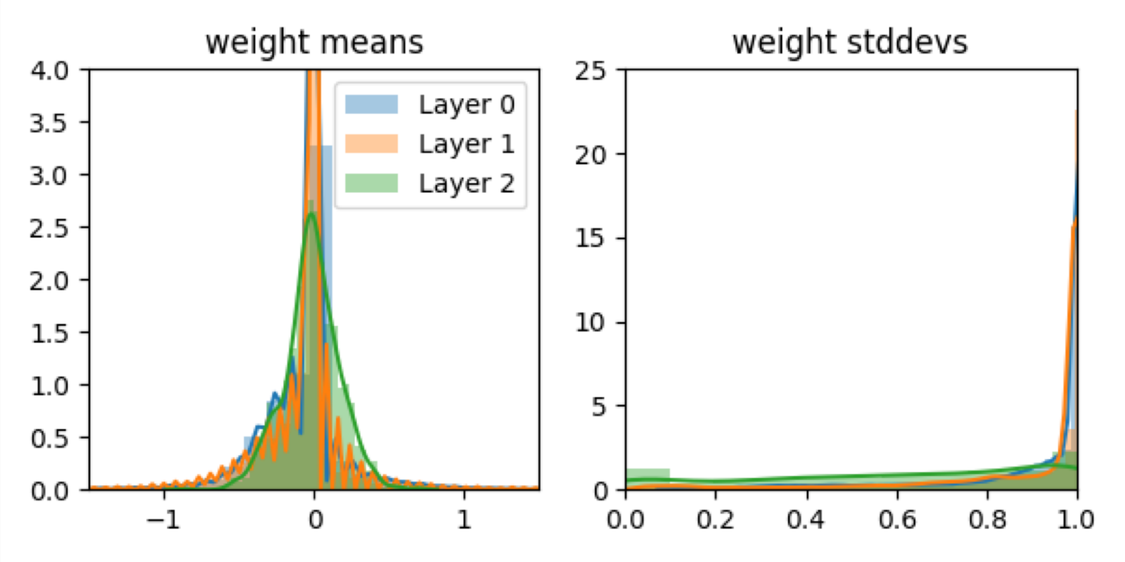


BNN results: weights

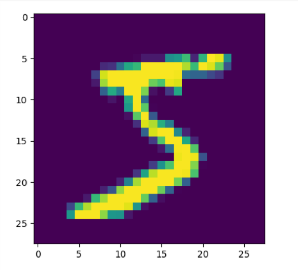
Iter:400



Iter:6000



Standard Neural Network:
Softmax outputs

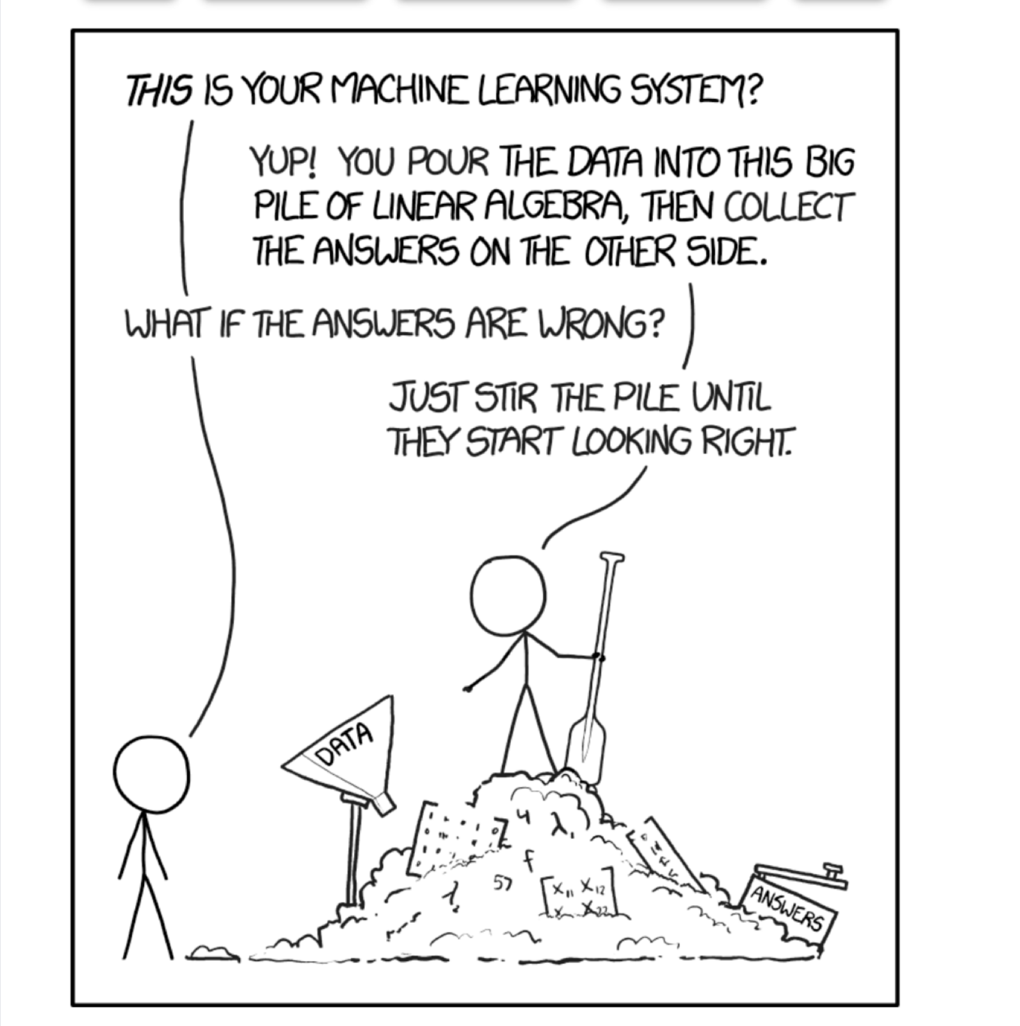



```
0: 1.2586000e-29
1: 0.0000000e+00
2: 0.0000000e+00
3: 5.2634514e-20
4: 0.0000000e+00
5: 1.0000000e+00
6: 0.0000000e+00
7: 0.0000000e+00
8: 1.0346410e-36
9: 1.7145724e-26
```

Softmax is **not** a measure of model or statistical uncertainty.

A model can be uncertain in prediction even with high softmax

Thank you !


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS