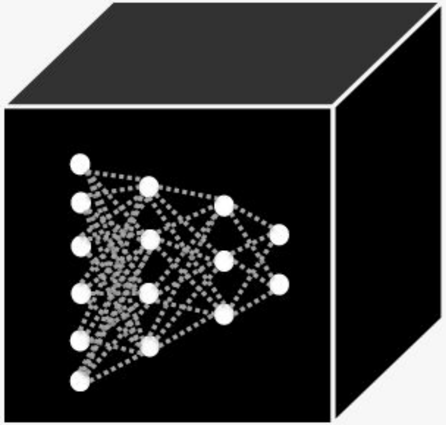
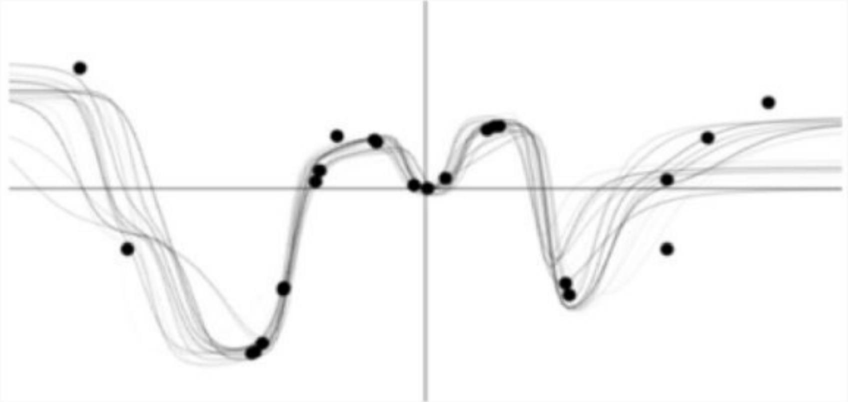
Batch Ensemble: Quantifying Uncertainty in Deep Nets

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

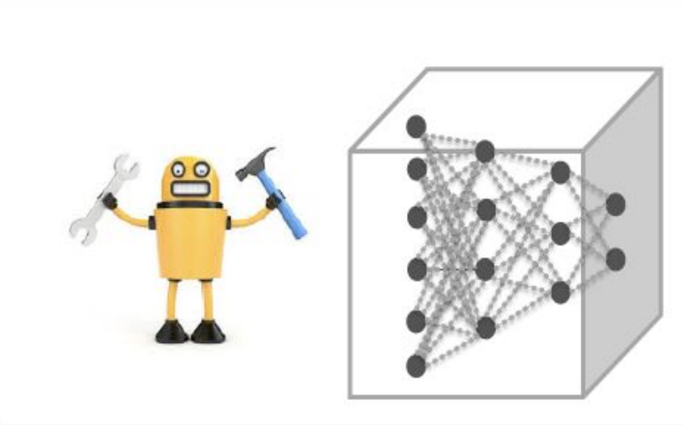




Understanding what a model does not know is a critical part of any scientific analysis

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

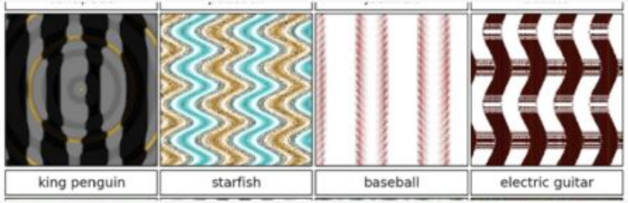
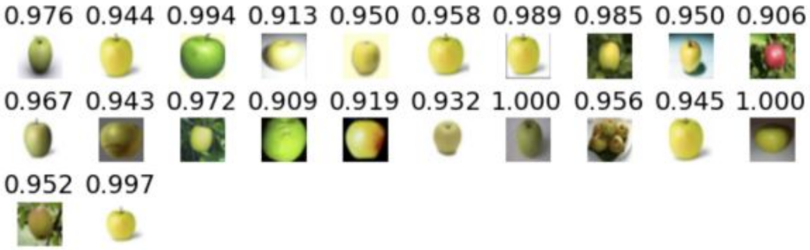


Figure 1. Evolved images that are unrecognizable to humans, but that state-of-the-art DNNs trained on ImageNet believe with $\geq$ 99.6% certainty to be a familiar object. This result highlights differences between how DNNs and humans recognize objects.

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



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

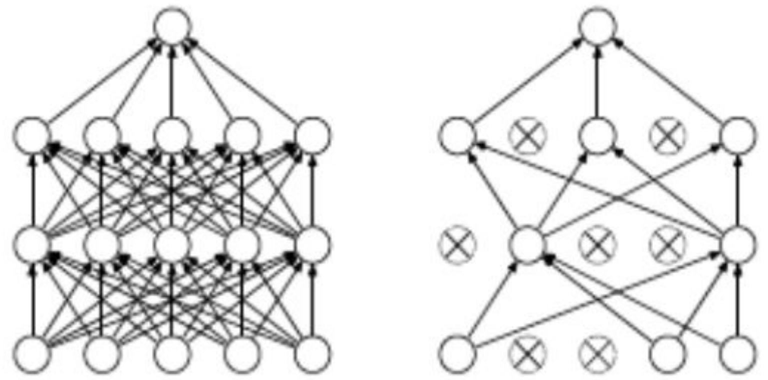
Candidate: Dropout

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

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

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



Candidate: Bayesian Nets A very strong candidate, the best up until 2020

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

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

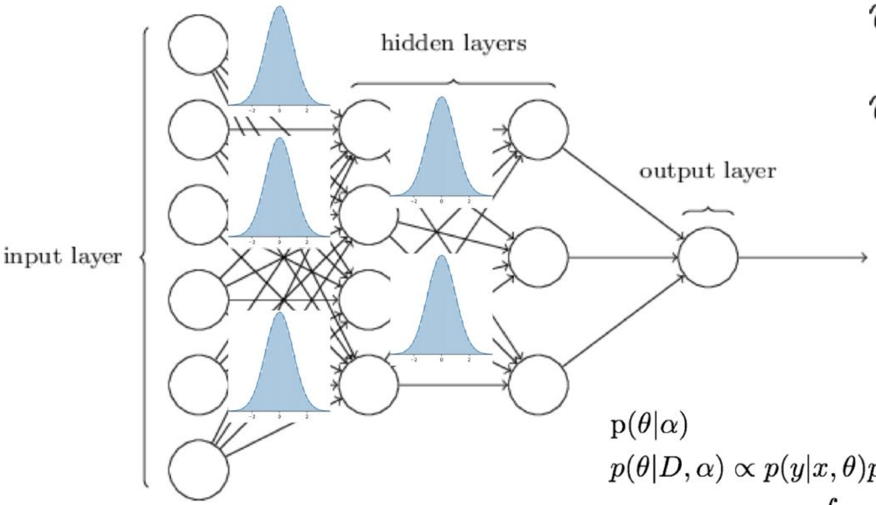
- A Bayesian Neural Network is a Neural Network with distributions over weights and biases. The loss which we are trying to minimize is the Posterior Distribution.

- 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

Candidate: Bayesian Nets A very strong candidate, the best up until 2020



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

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

Downsides:

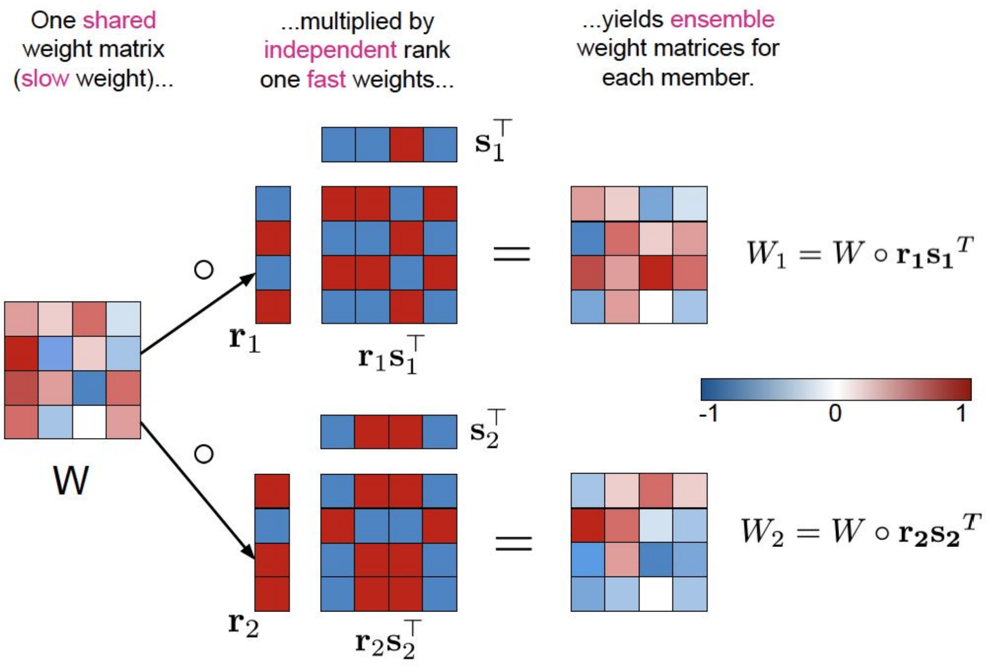
- Computationally more expensive
- Changes basic network architectures - scalability questions
- Benchmarked by Gustaffson and Ovadia as less effective than Ensembling (discrepency)

A Solution: Ensembling

Experiments since 1990’s
Since improved: Deep Ensembles, BatchEnsemble
Think of a Jury - bias across members, versus within members

BatchEnsemble

- “Novel ensemble weight generation” ..via matrix multiplication
- Simple Hadamard product of a shared weight matrix W and independent rank one matrix per Ensemble member
- Not as mathematically satisfying as Bayesian Nets…



Parallelizable

- More Nifty Linear Algebra
- ϕ : activation function
- n: index in mini-batch
- output: next layer's activations from previous ensemble member
-

$$y_n = \phi\left(\overline{W}_i^\top x_n\right) \tag{2}$$

$$= \phi\left(\left(W \circ r_i s_i^\top\right)^\top x_n\right) \tag{3}$$

$$= \phi\left(\left(W^\top (x_n \circ r_i)\right) \circ s_i\right), \tag{4}$$

$$Y = \phi\left(((X \circ R)W) \circ S\right). \tag{5}$$

Costs

Computational

- Hadamard product is the only add: relatively cheap to matrix multiplication
- Practically no computational overhead

Memory

- Extra vector sets R and S are the only adds: relatively cheap to matrices
- Practically no memory overhead (Naive ensemble has an order of magnitude more

	Vanilla	BatchE	DEN	PNN	RCL
Computational	1	**1.11**	9.58	1.12	26.41
Memory	1	**1.10**	5.31	4.16	2.52

Lifelong Learning with Batch Ensemble

- Naive ensembles cannot feasibly be applied to sequential tasks due to memory and computational overhead
- Only fast weights adapted to new task (R, S)
- Thus the shared weight is only trained on the 1st task
- This raises concerns:
  - only the information learnt in the first task can be transferred to subsequent tasks
  - rank-1 matrices contain little informational change so subsequent tasks cannot be significantly varied
- Yet computationally superior in testing/training...

For tasks:

$$t \in \{1, 2, \ldots, T\}$$

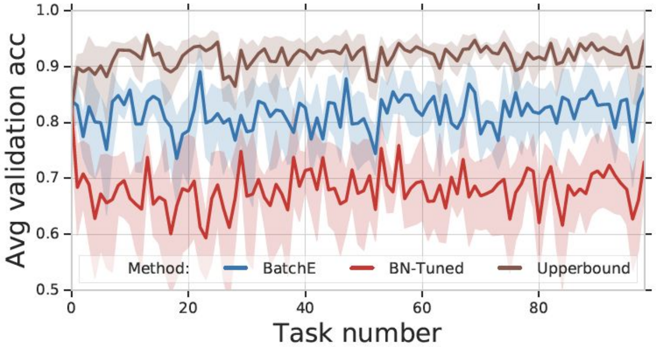
Task 1

$$\min_{W, s_1, r_1} L_1(W, s_1, r_1; D_1),$$

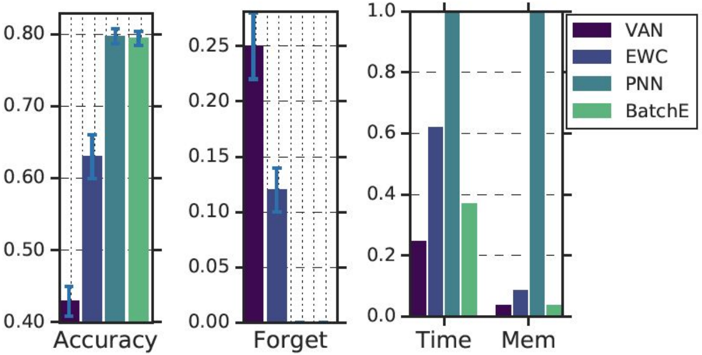
Task 2

$$\min_{s_t, r_t} L_t(s_t, r_t; D_t).$$

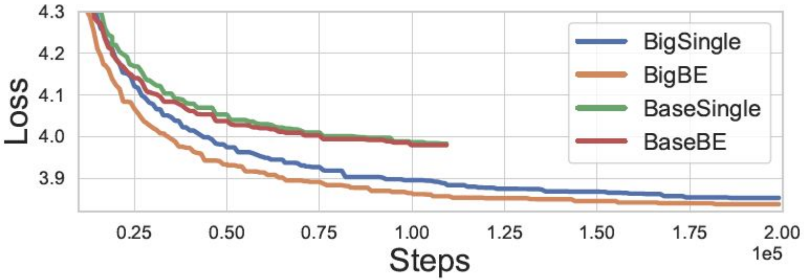
Results and Benchmark Comparisons w/ Naive Ensembles



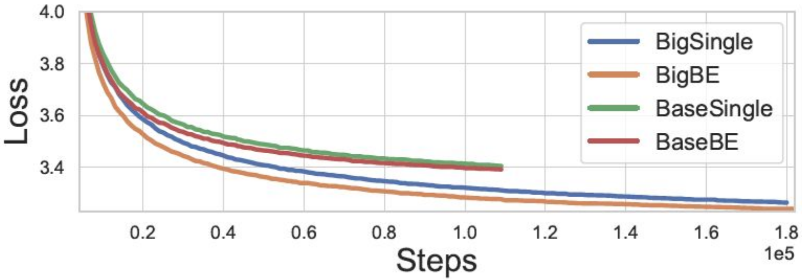
(a) Averaged validation accuracy on Split-ImageNet



(b) Various of measures on Split-CIFAR100



(a) English-German



(b) English-French

	Vanilla	MC-drop	BatchEnsemble	NaiveSmall
CIFAR10	95.31	95.72	**95.94**	95.59
CIFAR100	78.32	78.89	**80.32**	79.09

Marginal improvements over Naive Ensembles in classification

Questions of corruption, dataset diversity, predictive uncertainty

- Corrupted dataset CIFAR-10
- Diversity: independent errors from independent members makes for much more effective ensemble
- Predictive uncertainty still best captured by a distribution (fitting a Bayesian inference problem) rather than a point-estimate

Across all of these, BatchEnsemble does comparatively the same against Naive Ensemble because of the rank one minimal perturbations, but well against Dropout Ensemble : the authors focus on comparisons to the latter

Conclusions

- Uncertainty quantification is burgeoning…
- Batch ensemble is a perfect example of why it's not necessarily an esoteric, math heavy field
- Bayesian NN's versus Ensembling (Batch, Deep, Naive) - yet another gap between practice and theory
-