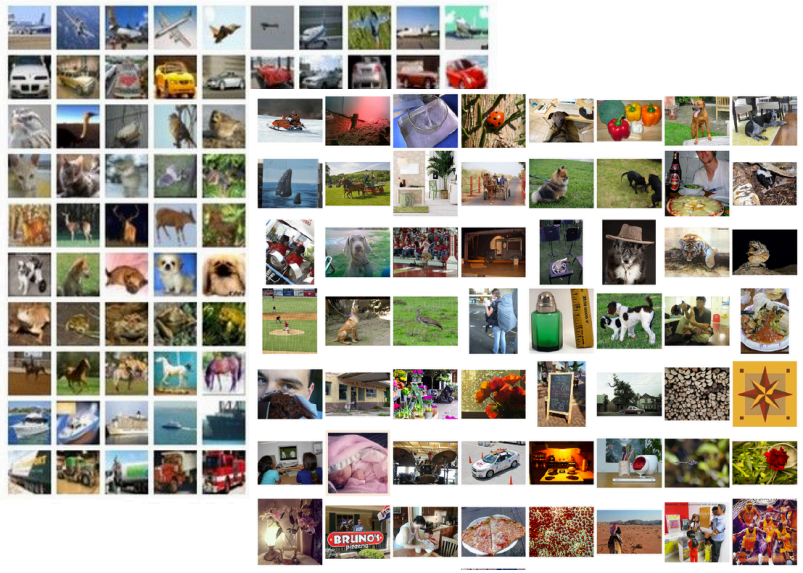
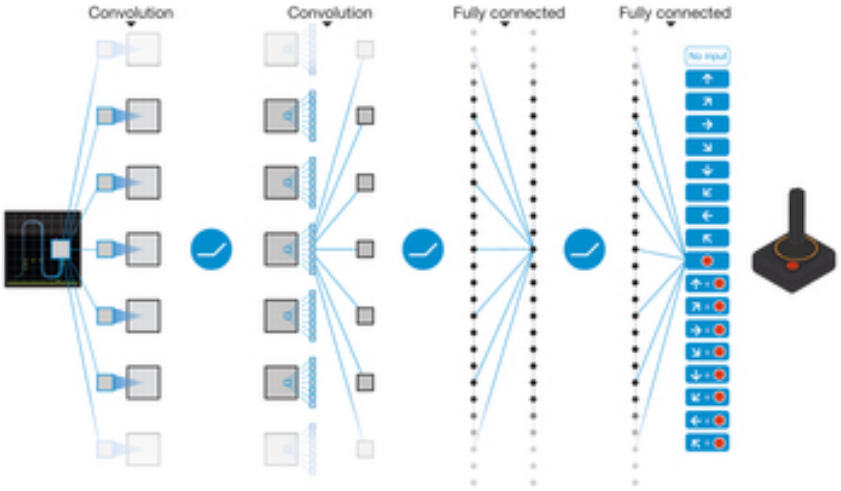
# Assumed Density Filtering Methods For Learning Bayesian Neural Networks

S. Ghosh*, **F. M. Delle Fave**, J. Yedidia**
Disney Research Pittsburgh
* IBM research
** Lyric Labs

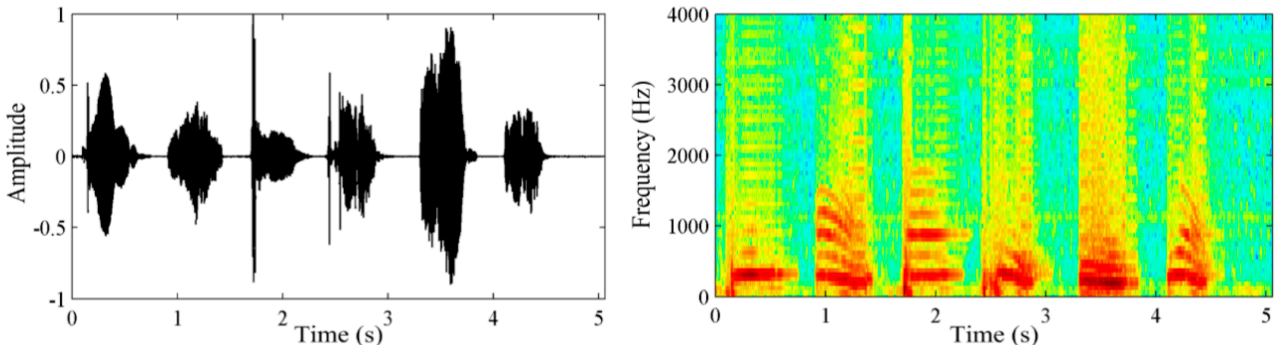
# (Deep) Neural Networks – Impressive Performance



Krizhevsky ‘09 ’15, Russakovsky ‘15




Mnih ‘15




Maas’15, Graves’14, Hinton’12

# Outline

I. Motivation and Challenges
II. Assumed Density Filtering (ADF)
III. ADF for Multi-Class Classification
IV. Experiments
V. Summary

# Outline

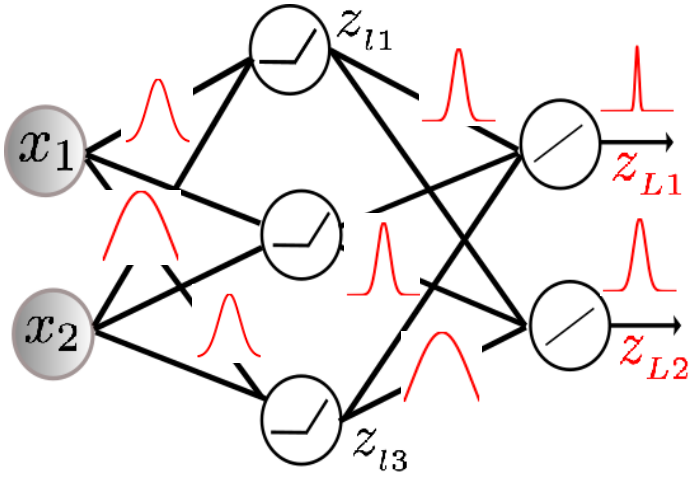
I. Motivation and Challenges
II. Assumed Density Filtering (ADF)
III. ADF for Multi-Class Classification
IV. Experiments
V. Summary

# Some challenges remain:

- Neural nets don't model uncertainty around predictions well.
- Prone to over fitting.
- Pesky learning parameters: learning rate, annealing schedule, pre-conditioners.

# Bayesian Neural Networks

- Endow model parameters with distributions
  (*old idea: Denker'91, Mackay'92, Neal'95* )




- Provides predictive uncertainty.
- Model selection via marginal likelihood
- Standard guards against overfitting

*Learning:* $$p(\mathcal{W} \mid \mathbf{x}, \mathbf{y}, \lambda) \propto p(\mathcal{W} \mid \lambda) \prod_{n=1}^{N} p(y_n \mid x_n, \mathcal{W})$$

*Inference:* $$p(y_{\text{test}} \mid x_{\text{test}}, \lambda) = \int p(y_{\text{test}} \mid \mathcal{W}, x_{\text{test}})\, p(\mathcal{W} \mid \mathbf{x}, \mathbf{y}, \lambda) d\mathcal{W}$$

# Posterior Intractability

- $p(\mathcal{W} \mid \mathbf{x}, \mathbf{y})$ is intractable, must be approximated.

MCMC:

- Traditional MCMC methods don't scale.

Stochastic gradient MCMC methods have been proposed (*Welling'11*).

Variational inference:

$$\vartheta^* = \underset{\vartheta}{\operatorname{argmin}} \operatorname{KL}(q(\mathcal{W} \mid \vartheta) || p(\mathcal{W} \mid \mathbf{x}, \mathbf{y}))$$

- Black Box variational inference: Stochastic gradient descent on variational free energy (*Graves'11, Blundell'15*)

**Still pesky learning parameters!**

# Outline

I. Motivation and Challenges
II. Assumed Density Filtering (ADF)
III. ADF for Multi-Class Classification
IV. Experiments
V. Summary

# Assumed Density Filtering (*Opper'98*)

1) Fully factorized approximation:

$$q(\mathcal{W} \mid \vartheta) = \prod_{l=1}^{L} \prod_{i=1}^{V_l} \prod_{j=1}^{V_{l-1}+1} q(w_{ijl} \mid \vartheta_{ijl})$$

# Assumed Density Filtering (*Opper'98*)

1) Fully factorized approximation:

$$q(\mathcal{W} \mid \vartheta) = \prod_{l=1}^{L} \prod_{i=1}^{V_l} \prod_{j=1}^{V_{l-1}+1} q(w_{ijl} \mid \vartheta_{ijl})$$

2) Online algorithm

$$\tilde{q}(w) = \frac{1}{Z} s(w) q(w \mid \vartheta^{n-1})$$

Update posterior beliefs after observing new evidence

# Assumed Density Filtering (*Opper'98*)

1) Fully factorized approximation:

$$q(\mathcal{W} \mid \vartheta) = \prod_{l=1}^{L} \prod_{i=1}^{V_l} \prod_{j=1}^{V_{l-1}+1} q(w_{ijl} \mid \vartheta_{ijl})$$

2) Online algorithm

$$\tilde{q}(w) = \frac{1}{Z} s(w) q(w \mid \vartheta^{n-1})$$

Update posterior beliefs after observing new evidence

3) After update q no longer in simple parametric form

-> project to tractable approximating family

$$\vartheta^n = \underset{\vartheta}{\operatorname{argmin}} \, \mathrm{KL}(\tilde{q}(w) || q(w \mid \vartheta))$$

# The ADF Algorithm:

For Gaussian approximations (*Minka'01*):

$$m_{ijl}^{n} = m_{ijl}^{n-1} + v_{ijl}^{n-1} \frac{\partial \ln Z}{\partial m_{ijl}^{n-1}},$$

$$v_{ijl}^{n} = v_{ijl}^{n-1} - (v_{ijl}^{n-1})^2 \left[ \left( \frac{\partial \ln Z}{\partial m_{ijl}^{n-1}} \right)^2 - 2 \frac{\partial \ln Z}{\partial v_{ijl}^{n-1}} \right]$$

Both updates require the marginal likelihood:

$$\ln Z = \ln \int p(y_n \mid x_n, \mathcal{W}) q(\mathcal{W} \mid \vartheta^{n-1}) d\mathcal{W}$$

# ADF algorithms: Mechanics

Approximate marginal likelihood:

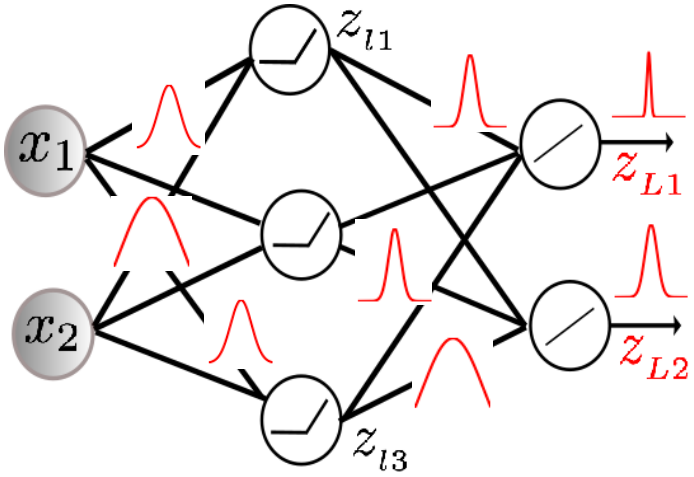
$$\ln \mathrm{Z} \approx \ln \int p(\mathbf{y}_n \mid \mathbf{z}_L) \mathcal{N}(\mathbf{z}_L \mid \nu_L, \Psi_L) d\mathbf{z}_L$$

# ADF algorithms: Mechanics

Approximate marginal likelihood:

$$\ln \mathrm{Z} \approx \ln \int p(\mathbf{y}_n \mid \mathbf{z}_L)\mathcal{N}(\mathbf{z}_L \mid \nu_L, \Psi_L)d\mathbf{z}_L$$

1 - Forward propagate distributions and approximate layer outputs with Gaussians by moment matching:



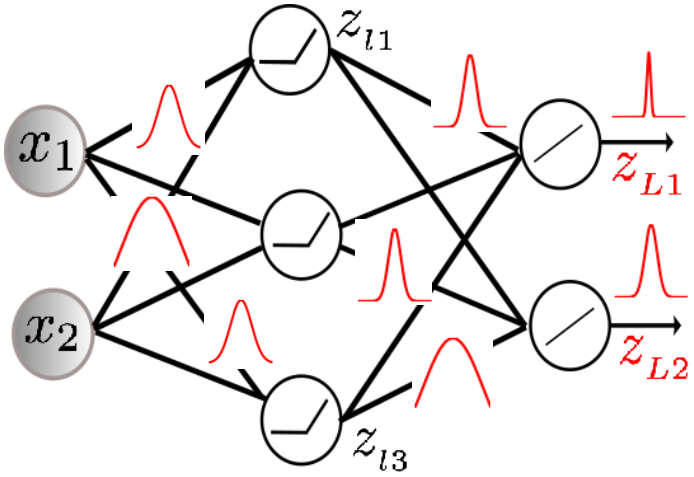

$$z_l \sim \mathcal{N}\left(\begin{bmatrix}\mathbb{E}[z_{1l}]\\ \mathbb{E}[z_{1l}]\\ \vdots\\ 1\end{bmatrix}, \begin{bmatrix}\mathrm{var}(z_{1l}) & 0 & \cdots & 0\\ 0 & \mathrm{var}(z_{2l}) & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 0\end{bmatrix}\right)$$

# ADF algorithms: Mechanics

Approximate marginal likelihood:

$$\ln \mathrm{Z} \approx \ln \int p(\mathbf{y}_n \mid \mathbf{z}_L)\mathcal{N}(\mathbf{z}_L \mid \nu_L, \Psi_L)d\mathbf{z}_L$$

1 - Forward propagate distributions and approximate layer outputs with Gaussians by moment matching:




$$z_l \sim \mathcal{N}\left(\begin{bmatrix}\mathbb{E}[z_{1l}]\\ \mathbb{E}[z_{1l}]\\ \vdots \\ 1\end{bmatrix}, \begin{bmatrix}\mathrm{var}(z_{1l}) & 0 & \cdots & 0\\ 0 & \mathrm{var}(z_{2l}) & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 0\end{bmatrix}\right)$$

2 - Backward propagate gradients of the marginal likelihood.

# ADF for Bayesian Neural Networks

Probabilistic Back propagation (PBP) *Hernandez-Lobato' 15*:

$$q(w_{ijl} \mid \vartheta_{ijl}) = \mathcal{N}(w_{ijl} \mid m_{ijl}, v_{ijl})$$

Expectation Back propagation (EBP) *Soudry' 14*:

$$q(w_{ijl} \mid \vartheta_{ijl}) = \mathcal{N}(w_{ijl} \mid m_{ijl}, 1)$$

# EBP vs PBP: Direct Comparison Necessary

PBP might not always be the better algorithm:

- ADF methods approximate uncertainty incorrectly
  Multiple passes over same data points

EBP cruder approximation might be good enough

Comprehensive comparisons were not performed:

| | can do | need work |
|---|---|---|
| **EBP** | binary neurons, binary classification | regression, count regression, multi-class classification, rectified linear units |
| **PBP** | rectified linear neurons continuous regression | count regression, binary classification, multi-class classification |

# Outline

I. Motivation and Challenges
II. Assumed Density Filtering (ADF)
III. ADF for Multi-Class Classification
IV. Experiments
V. Summary

# Multi-class Classification

Likelihood: Softmax transformed parameterizations

$$\mathbf{y}_n \mid \mathbf{z}_L \sim \mathrm{Cat}(\sigma(\mathbf{z}_L)) \qquad \sigma(a)_j = e^{a_j} / \sum_{k=1}^{C} e^{a_k}$$

$$\mathbf{z}_L = g(x_n, \mathcal{W}) \sim \mathcal{N}(\mathbf{z}_L \mid \nu_L, \Psi_L) \in \mathbb{R}^C$$

*Number of classes*

Unfortunately, again: intractable marginal likelihood

$$\ln Z \approx \ln \int e^{\mathbf{y}_n^T \mathbf{z}_L - \mathrm{lse}(\mathbf{z}_L)} \mathcal{N}(\mathbf{z}_L | \nu_L, \Psi_L) d\mathbf{z}_L$$

Log of the sum of exponentials

# Making the intractable tractable

Unfortunately, again: intractable marginal likelihood

$$\ln Z \approx \ln \int e^{\mathbf{y}_n^T \mathbf{z}_L - \mathrm{lse}(\mathbf{z}_L)} \mathcal{N}(\mathbf{z}_L | \nu_L, \Psi_L) d\mathbf{z}_L$$

First Idea: lower bound on the marginal likelihood:

Log Bound

$$\ln Z \geq \mathbf{y}^T \nu_L - \mathrm{lse}(\nu_L + \psi_L/2)$$

$$\nabla_{\nu,\psi} \ln Z \approx \nabla_{\nu,\psi} [\mathbf{y}^T \nu_L - \mathrm{lse}(\nu_L + \psi_L/2)]$$

# Second Idea: Stochastic Marginal Likelihood

$$\ln \mathrm{Z} \approx \ln \frac{1}{S} \sum_{s=1}^{S} p(y_n \mid \mathbf{z}_L^{(s)}) \qquad \mathbf{z}_L^{(s)} \sim \mathcal{N}(\mathbf{z}_L \mid \nu_L, \Psi_L)$$

We use the "re-parameterization trick" (*Kingma'13*) to compute low variance stochastic gradients

$$\mathbf{z}_L^{(s)} = \nu_L + M\epsilon^{(s)} \triangleq t(\nu_L, \psi_L, \epsilon^{(s)}) \qquad \epsilon^{(s)} \sim \mathcal{N}(0, I) \quad \Psi_L = MM^T$$

Stochastic Gradients

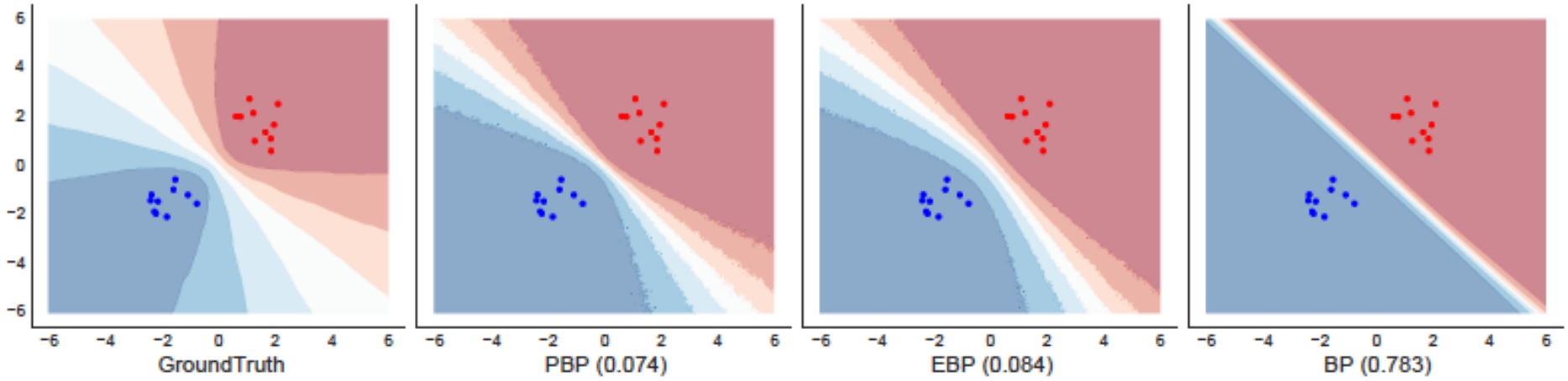
*(more accurate than log bound gradients)*

$$\nabla_{\nu,\Psi} \ln \mathrm{Z} \approx \frac{1}{Z} \left[ \frac{1}{S} \sum_{s=1}^{S} \nabla_{\nu,\Psi} p\left( \mathbf{y}_n \mid t(\nu_L, \Psi_L, \epsilon^{(s)}) \right) \right]$$

# Outline

I. Motivation and Challenges
II. Assumed Density Filtering (ADF)
III. ADF for Multi-Class Classification
IV. Experiments
V. Summary

# EBP vs PBP: Posterior Quality



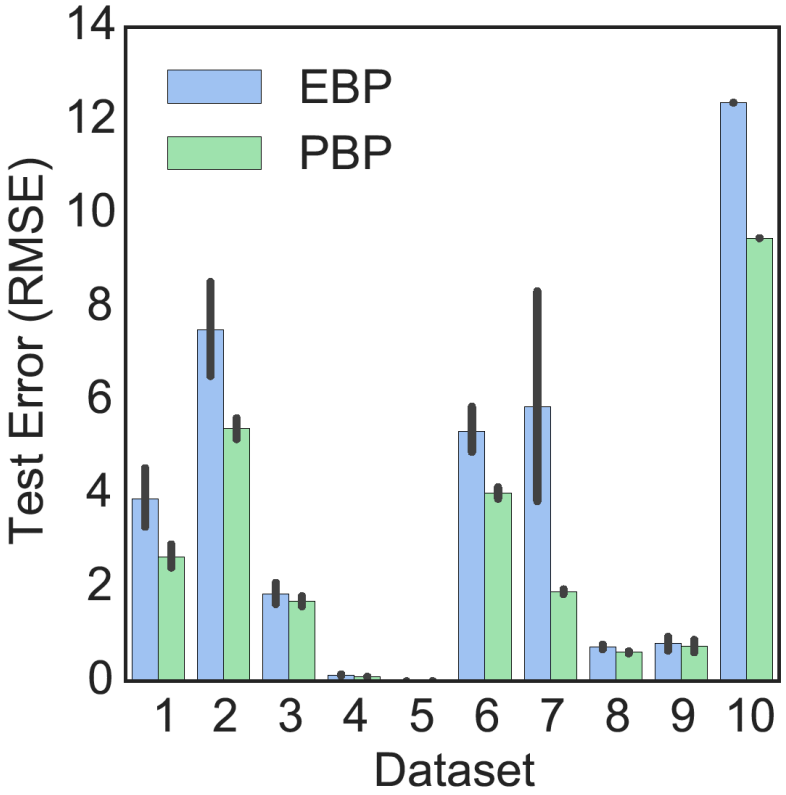

Results:

- PBP outperforms EBP
  - Posterior much more similar to the ground truth.

# EBP vs PBP: Regression

Setup:
1 hidden layer – 50 (100) units
10 datasets
10 experiments: 90 / 10 split

| | Dataset | $N$ | d |
|---|---|---|---|
| 1 | Boston | 506 | 13 |
| 2 | Concrete Compression Strength | 1030 | 8 |
| 3 | Energy Efficiency | 768 | 8 |
| 4 | Kin8nm | 8192 | 8 |
| 5 | Naval Propulsion | 11,934 | 16 |
| 6 | Combined Cycle Power Plant | 9568 | 4 |
| 7 | Protein Structure | 9568 | 4 |
| 8 | Wine Quality Red | 1599 | 11 |
| 9 | Yacht Hydrodynamics | 308 | 6 |
| 10 | Year Prediction MSD | 515,345 | 90 |



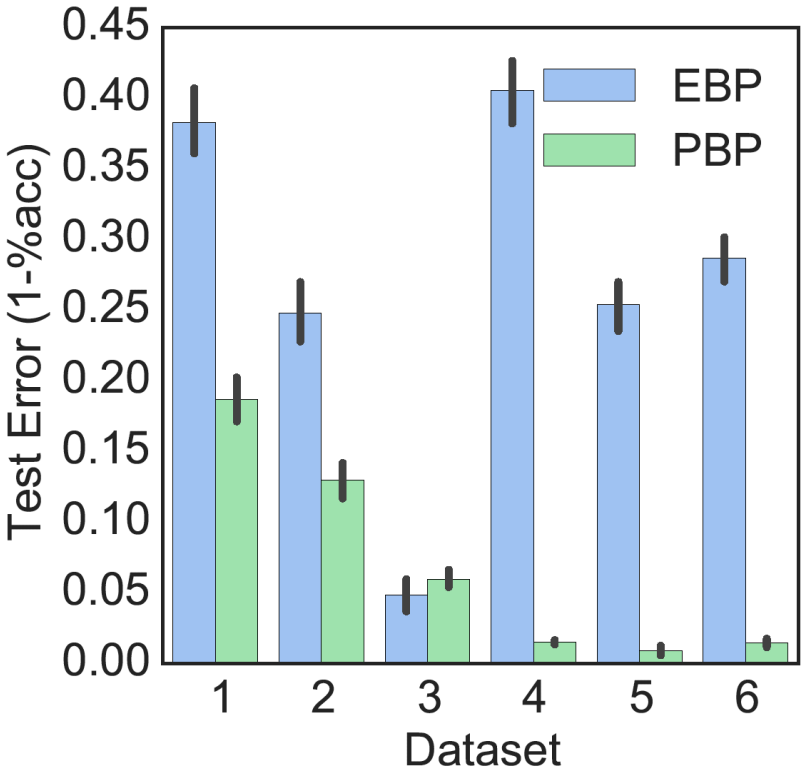

Results:

PBP consistently outperforms EBP

# EBP vs PBP: Binary Classification

Setup:
1 hidden layer – 120 units
6 datasets
10 experiments: 90 / 10 split

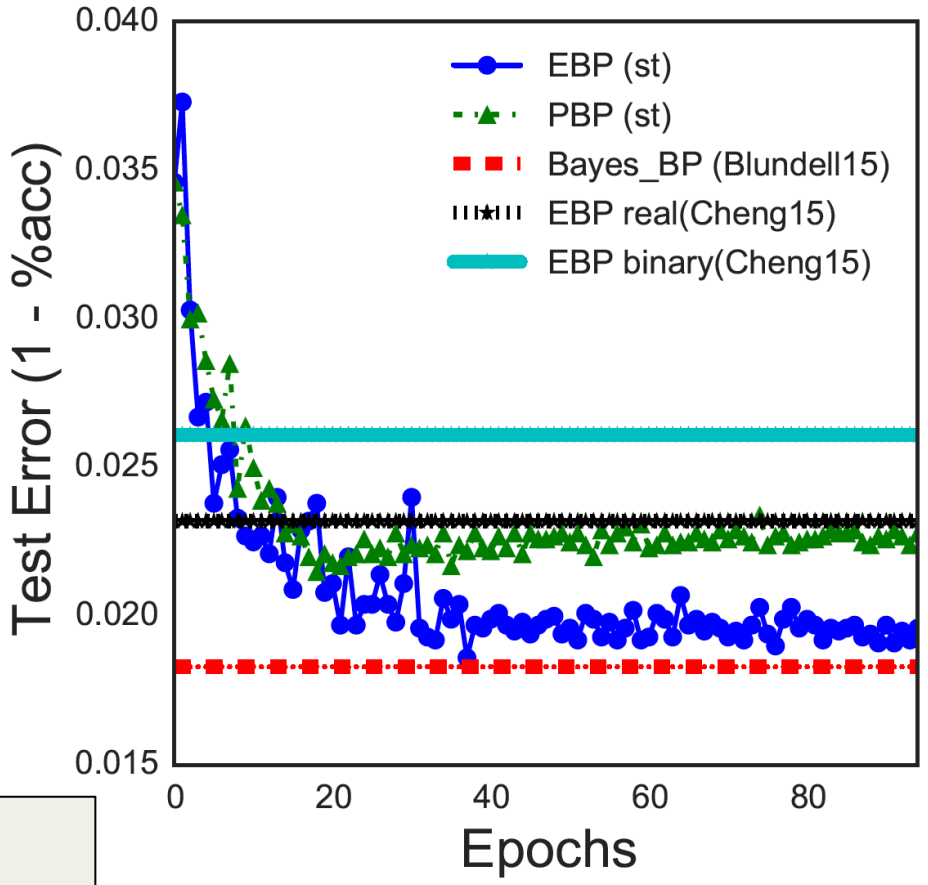
| | Dataset | $N$ | d |
|---|---|---|---|
| 1 | 20News group comp vs HW | 1943 | 29409 |
| 2 | 20News group elec vs med | 1971 | 38699 |
| 3 | Spam or ham d0 | 2500 | 26580 |
| 4 | Spam or ham d1 | 2500 | 27523 |
| 5 | Reuters news I8 | 2000 | 12167 |
| 6 | Reuters news I6 | 2000 | 11463 |




Results:
Consistent w. Regression
PBP better than EBP on most datasets

# EBP vs PBP: Multi-Class

Setup:
EBP vs PBP stochastic bound (100 samples)
2 hidden layer – 400 units
1 experiment: 100 epochs
Training: 60 000 images
Test: 10 000 images




Results:
On MNIST --- PBP appears to overfit marginally quicker than EBP.

# Outline

I. Motivation and Challenges
II. Assumed Density Filtering (ADF)
III. ADF for Multi-Class Classification
IV. Experiments
V. Summary

# Summary

- Extend two algorithms PBP and EBP
  - Multi-class classification
  - Regression + Count Regression
  - Rectified Linear Units
- Experiments on different learning tasks
  - PBP outperforms EBP