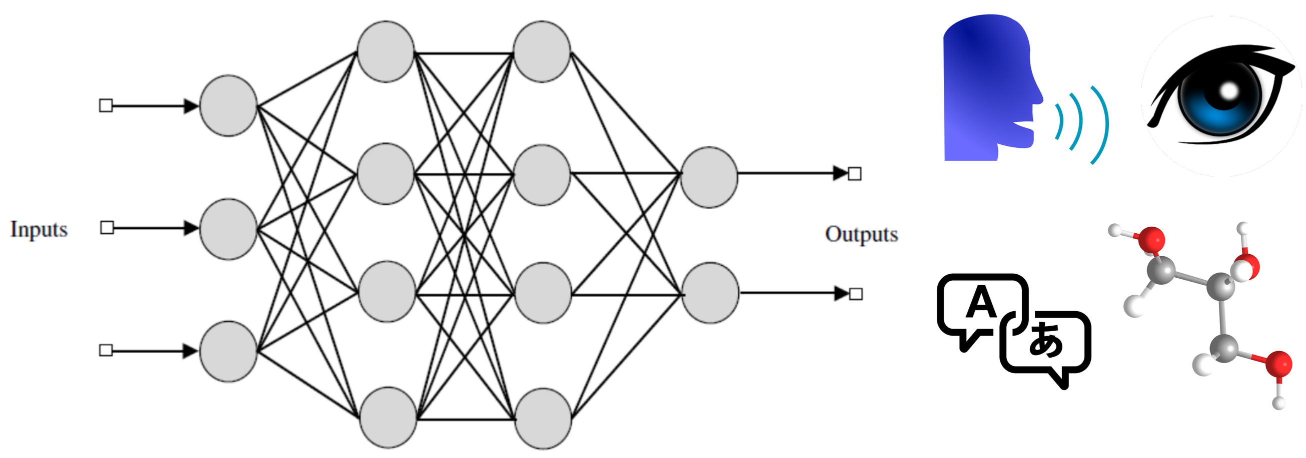
Probabilistic Backpropagation for Scalable Learning of Bayesian Neural Networks

José Miguel Hernández-Lobato[1] and Ryan P. Adams[1] [1]Harvard University

HARVARD UNIVERSITY

ICML Lille — International Conference on Machine Learning

1. Motivation

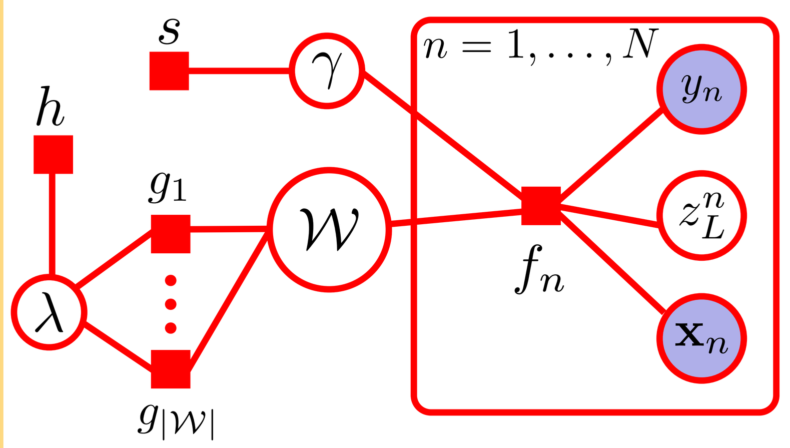


- Multilayer neural networks are **state-of-the-art** techniques, but...
  - Require tuning of **hyper-parameters**.
  - Are affected by **overfitting** problems.
  - Lack estimates of **uncertainty** in their predictions.

The **Bayesian approach** can solve these problems but existing methods lack scalability, **until now...**

2. Probabilistic Multilayer Neural Networks

- **ReLUs activations** for the hidden units: $a(x) = \max(x, 0)$.
- **The likelihood**: $p(\mathbf{y}|\mathcal{W}, \mathbf{X}, \gamma) = \prod_{n=1}^{N} \mathcal{N}(y_n | z_L(\mathbf{x}_n|\mathcal{W}), \gamma^{-1}) \equiv f_n$.
- **The priors**: $p(\mathcal{W}|\lambda) = \prod_{l=1}^{L} \prod_{i=1}^{V_l} \prod_{j=1}^{V_{l-1}+1} \mathcal{N}(w_{ij,l}|0, \lambda^{-1}) \equiv g_k$, $p(\lambda) = \text{Gamma}(\lambda|\alpha_0^\lambda, \beta_0^\lambda) \equiv h$, $p(\gamma) = \text{Gamma}(\gamma|\alpha_0^\gamma, \beta_0^\gamma) \equiv s$.



The **posterior approximation** is

$$q(\mathcal{W}, \gamma, \lambda) = \left[\prod_{l=1}^{L} \prod_{i=1}^{V_l} \prod_{j=1}^{V_{l-1}+1} \mathcal{N}(w_{ij,l}|m_{ij,l}, v_{ij,l})\right] \text{Gamma}(\gamma|\alpha^\gamma, \beta^\gamma)\, \text{Gamma}(\lambda|\alpha^\lambda, \beta^\lambda) \,.$$

3. Probabilistic Backpropagation

After seeing the n-th data point, Bayes rule updates our beliefs $q(w)$ as

$$p(w) = Z^{-1} \mathcal{N}(y_n | z_L(\mathbf{x}_n|w), \sigma^2) q(w) \,,$$

where Z is the normalization constant. Network output Modeling noise

PBP uses $q(w) = \mathcal{N}(w|m, v)$ and approximates $p(w)$ with $\mathcal{N}(w|m^{\text{new}}, v^{\text{new}})$

We **match moments** between p(w) and $q^{\text{new}}(w)$

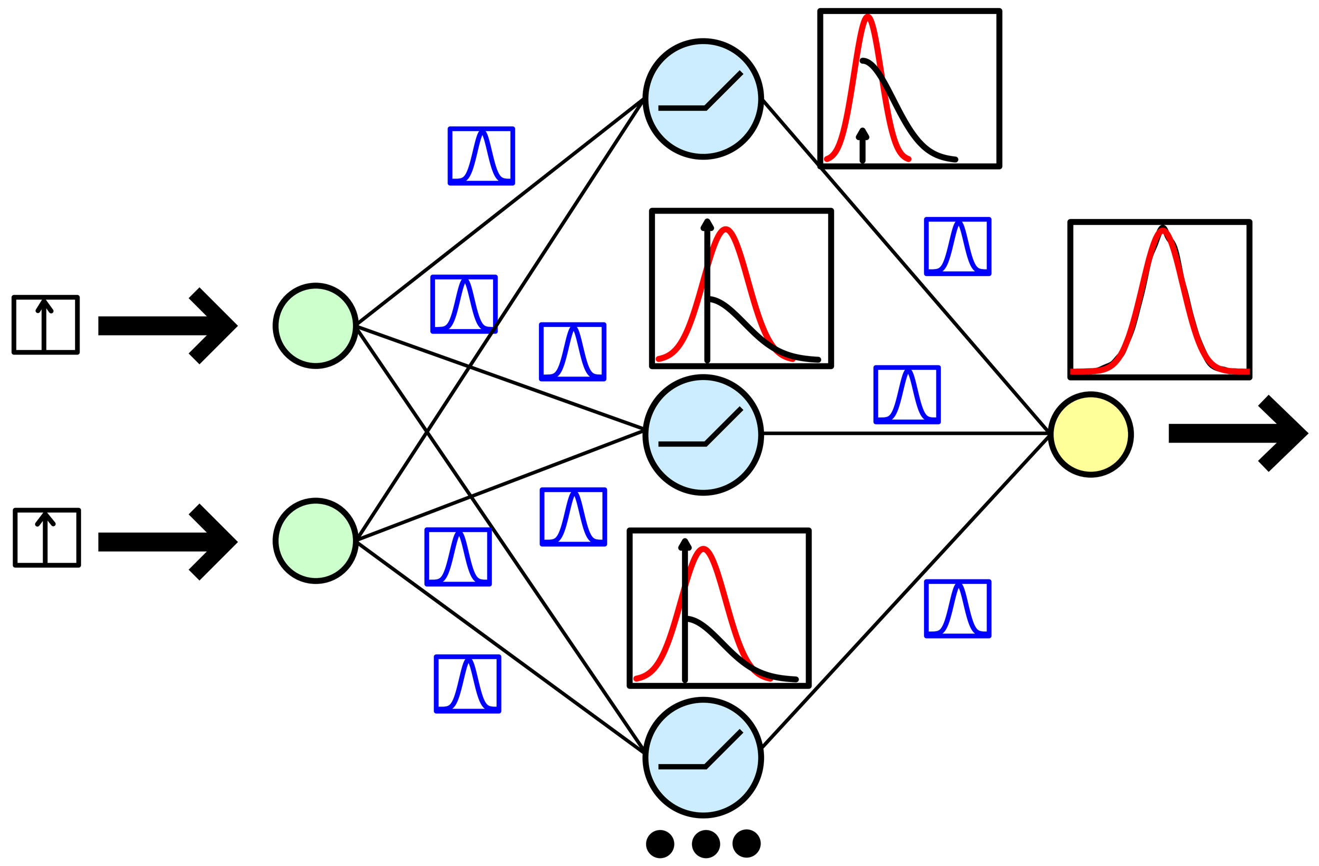
$$\boxed{\begin{aligned} m^{\text{new}} &= m + v \frac{\partial \log Z}{\partial m}, \\ v^{\text{new}} &= v - v^2 \left[\left(\frac{\partial \log Z}{\partial m} \right)^2 - 2 \frac{\partial \log Z}{\partial v} \right] . \end{aligned}}$$

We need a way to approximate Z and then obtain its gradients!

Easy if $z_L(\mathbf{x}_n|w)$ is Gaussian distributed when $w \sim q(w)$.

4. Forward Pass

Propagate distributions through the network and approximate them with **Gaussians** by moment matching.



Given $\log Z$, we compute its gradients by **backpropagation**.

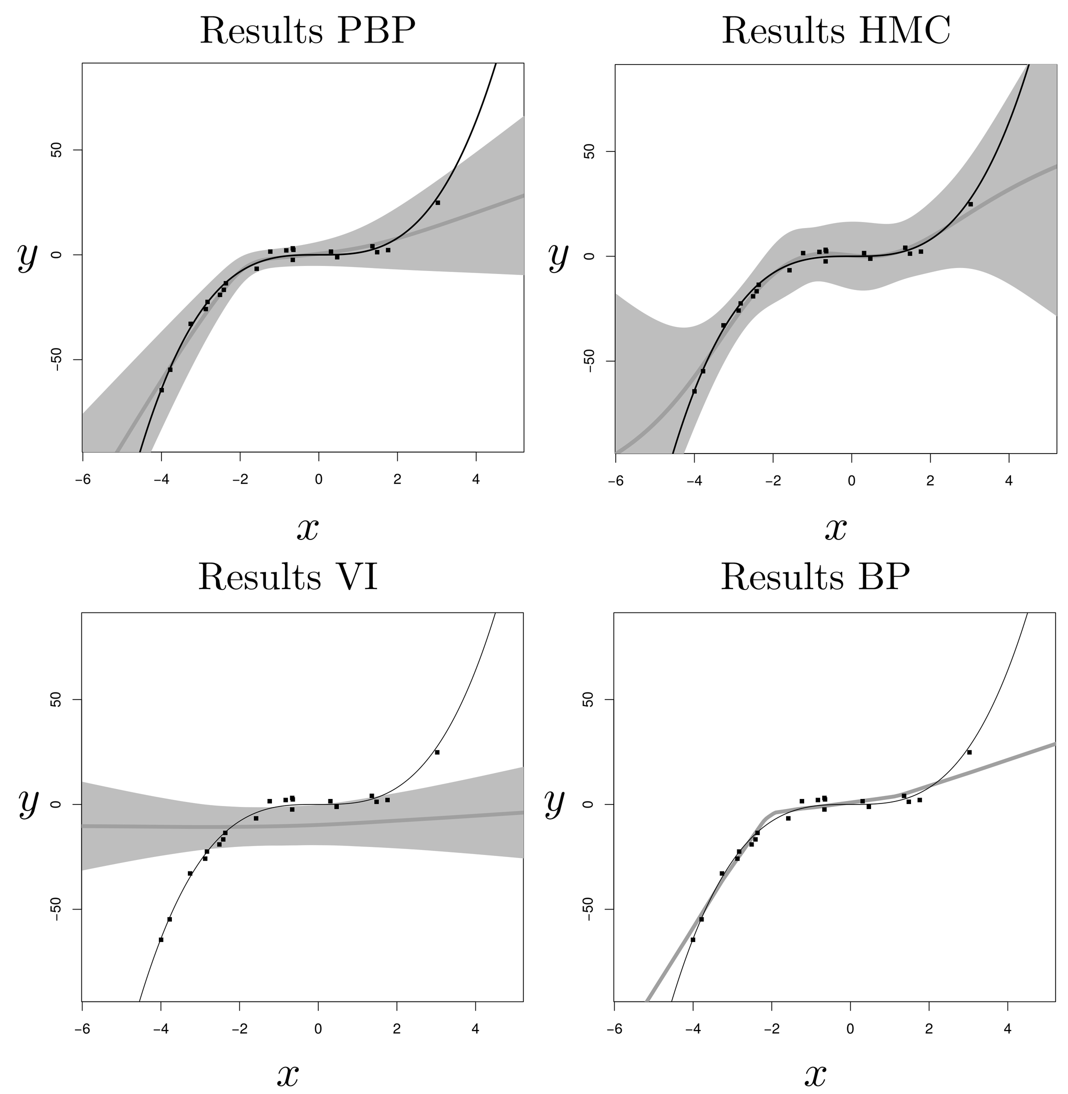
5. Results on Toy Dataset

40 training epochs.

100 hidden units.

VI uses **two stochastic approximations** to the ELBO.

BP and VI tuned with Bayesian optimization (`www.whetlab.com`).



6. Results Predictive Performance and Running Time

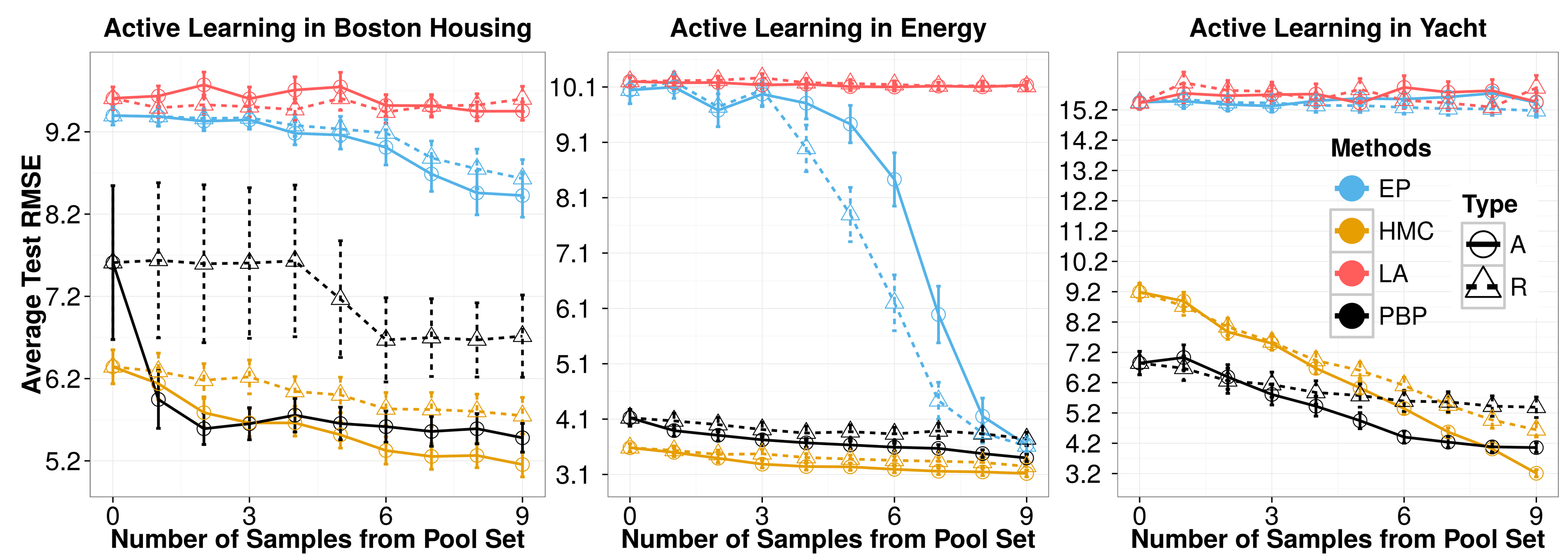
Dataset	N	d	Avg. Test RMSE and Std. Errors VI	BP	PBP	Avg. Test LL and Std. Errors VI	PBP	Avg. Running Time in Secs VI	BP	PBP
Boston Housing	506	13	4.320±0.2914	3.228±0.1951	**3.014±0.1800**	-2.903±0.071	**-2.574±0.089**	1035	677	**13**
Concrete Compression Strength	1030	8	7.128±0.1230	5.977±0.2207	**5.667±0.0933**	-3.391±0.017	**-3.161±0.019**	1085	758	**24**
Energy Efficiency	768	8	2.646±0.0813	**1.098±0.0738**	1.804±0.0481	-2.391±0.029	**-2.042±0.019**	2011	675	**19**
Kin8nm	8192	8	0.099±0.0009	**0.091±0.0015**	0.098±0.0007	**0.897±0.010**	0.896±0.006	5604	2001	**156**
Naval Propulsion	11,934	16	0.005±0.0005	**0.001±0.0001**	0.006±0.0000	**3.734±0.116**	3.731±0.006	8373	2351	**220**
Combined Cycle Power Plant	9568	4	4.327±0.0352	4.182±0.0402	**4.124±0.0345**	-2.890±0.010	**-2.837±0.009**	2955	2114	**178**
Protein Structure	45,730	9	4.842±0.0305	**4.539±0.0288**	4.732±0.0130	-2.992±0.006	**-2.973±0.003**	7691	4831	**485**
Wine Quality Red	1599	11	0.646±0.0081	0.645±0.0098	**0.635±0.0079**	-0.980±0.013	**-0.968±0.014**	1195	917	**50**
Yacht Hydrodynamics	308	6	6.887±0.6749	1.182±0.1645	**1.015±0.0542**	-3.439±0.163	**-1.634±0.016**	954	626	**12**
Year Prediction MSD	515,345	90	9.034±NA	8.932±NA	**8.879± NA**	-3.622±NA	**-3.603± NA**	142,077	65,131	**6119**

7. Results with More than One Hidden Layer

Table : Average Test RMSE.

Method	Standard BP				PBP			
Dataset	**1 Layer**	**2 Layers**	**3 Layers**	**4 Layers**	**1 Layer**	**2 Layers**	**3 Layers**	**4 Layers**
Boston	3.23±0.20	3.18±0.24	3.02±0.18	2.87±0.16	3.01±0.18	**2.80±0.16**	2.94±0.16	3.09±0.15
Concrete	5.98±0.22	5.40±0.13	5.57±0.13	5.53±0.14	5.67±0.09	**5.24±0.12**	5.73±0.11	5.96±0.16
Energy	1.18±0.12	0.68±0.04	**0.63±0.03**	0.67±0.03	1.80±0.05	0.90±0.05	1.24±0.06	1.18±0.06
Kin8nm	0.09±0.00	0.07±0.00	0.07±0.00	0.07±0.00	0.10±0.00	**0.07±0.00**	0.07±0.00	0.07±0.00
Naval	0.00±0.00	**0.00±0.00**	0.00±0.00	0.00±0.00	0.01±0.00	0.00±0.00	0.01±0.00	0.00±0.00
Power Plant	4.18±0.04	4.22±0.07	4.11±0.04	4.18±0.06	4.12±0.03	**4.03±0.03**	4.06±0.04	4.08±0.04
Protein	4.54±0.02	4.18±0.03	4.02±0.03	**3.95±0.02**	4.69±0.01	4.24±0.01	4.10±0.02	3.98±0.03
Wine	0.65±0.01	0.65±0.01	0.65±0.01	0.65±0.02	**0.63±0.01**	0.64±0.01	0.64±0.01	0.64±0.01
Yacht	1.18±0.16	1.54±0.19	1.11±0.09	1.27±0.13	1.01±0.05	**0.85±0.05**	0.89±0.10	1.71±0.23
Year	8.93±NA	8.98±NA	8.93±NA	9.04±NA	**8.87± NA**	8.92±NA	8.87±NA	8.93±NA

8. Results Active Learning



9. Summary

- PBP is a state-of-the-art method for scalable inference in NNs.
- PBP is very similar to traditional backpropagation.
- PBP often outperforms backpropagation at a lower cost.
- Very fast C code available at `https://github.com/HIPS`