Revisiting Neural Networks for Large Scale Survey Data Analysis

Taps Maiti

Michigan State University & National Science Foundation, USA

Contributors Students and Colleagues

My Early Involvement

- Application in Public Health/Medicine
  - Ghosh, M., **Maiti**, T., Kim, D., Chakraborty, S. and Tewari, A. (2004). "Bayesian neural network modeling in prostate cancer detection". *Journal of the American Statistical Association*, **99**, 601-608.
  - Chakraborty, S., Ghosh, M., **Maiti**, T., and Tewari, A. (2005)."Bayesian neural networks for bivariate binary data: An application to prostate cancer study". *Statistics in Medicine*, **24**, 3645-3662.
- Application in Large-Scale Survey
  - **Maiti**, T., Mukhopadhyay, P. and Miller, C. (2008) "Neural network imputation: an experience with the natural resources inventory surveys". *Journal of Agricultural, Biological, and Environmental Statistics*, **13**, 255-269.

NN in the context of imputation

- Imputation for missing values in a complex survey requires extensive knowledge of subject matter.
- Hot-deck imputation and model-based imputation are the two most commonly used imputation procedures for survey data. Hot-deck imputation often uses imputation cells.
- For every missing observation, one or more donors are selected from the same imputation cell.
- Performance of a hot-deck imputation procedure largely depends on the choice of the imputation cells.
- A wide knowledge of subject matter is necessary to define the imputation cells in such a way that the imputed data resembles real data.

Model-based Imputation

- Model-based imputation uses an explicit model for every survey variable.
- Predicted values from the fitted model are used as imputed values (Sarndal et al. 1992).
- A reasonable model needs to be defined for each variable to be imputed.
- In a complex large-scale survey setup, exploring several models for each variable is not an easy task.

The NRI Survey

- The National Resources Inventory (NRI) is a longitudinal survey conducted by the U.S. Department of Agriculture's Natural Resources Conservation Service (NRCS) in cooperation with Iowa State University.
- The survey is designed to assess conditions and trends for land cover, soil, water, and related natural resources on nonfederal lands in the United States.
- The NRI survey is a stratified two-stage area sample survey. Stratification is based on geographic region and primary sampling units (PSU) are identified according to the Public Land Survey (PLS) system.
- The 1997 sample contains approximately 300,000 PSU and over 800,000 sample points. *For detail, see*
- Bell, A., Deignei, D., Dorsch, R. K., Fuller, W. A., Keinzler, J., **Maiti**, T., Nusser, S. M., Peterson, T. C., and Wolter, K. (2003), "Estimation Procedure for the 2002 NRI," Technical report, Center for Survey Statistics and Methodology, National Resource Conservation Service.

Can the process be semi-automated?

- We developed an artificial neural network (ANN) approach to impute missing values in a complex survey, such as NRI.
- ANN models can be built automatically, with minimum human intervention, for any survey variables.
- Unlike hot-deck imputation, extensive background information about survey variables are not required to construct ANN models.
- The main objective of this article is to propose an imputation procedure that requires very minimum human intervention with little or no background knowledge and that can be applied to large-scale surveys with hundreds of variables with missing values.

Outcome is encouraging

The proposed naive ANN (with very little knowledge of the background information) performs better than the existing imputation procedure with respect to the mean squared error of prediction. *for detail, see*

- **Maiti**, T., Mukhopadhyay, P. and Miller, C. (2008) "Neural network imputation: an experience with the natural resources inventory surveys". *Journal of Agricultural, Biological, and Environmental Statistics*, **13**, 255-269.

A New Age Problem

Query: Who are better Entrepreneurs?

Domestic Vs Immigrants

A Fresh Look at the Behavioral Research

RUNNING THE SHOW

Why Immigrants Make Such Good Entrepreneurs

They've often overcome a lot of hardships. A business setback is nothing.

BY ADRIAN FURNHAM

OUTSIDERS FACE a tough struggle fitting into a new culture. They must figure out how to deal with, and overcome, frustration, loneliness and a steep learning curve.

And *that's* why immigrants make such great entrepreneurs—they're once again outsiders facing many of the same

grants for over a decade, trying to figure out what makes so many of them go into business for themselves in the West—at higher rates than natives do—and succeed, too. The Kauffman Foundation's annual Index of Startup Activity shows that immigrants were almost twice as likely as native-born to start new businesses in the U.S. in 2016. Al-

New American Economy found that in 2016, 40.2% of Fortune 500 firms had "at least one founder who either immigrated to the United States or was the child of immigrants."

I'm not surprised. What I've found is that immigrants not only have the qualities that help *any* entrepreneurs succeed—including aggressiveness and creative thinking

Do we have any better data driven proof?

Can we leverage data from modern BRAIN research

New Age Data: Brain Imaging

High dimensionality in 3D MRI imaging

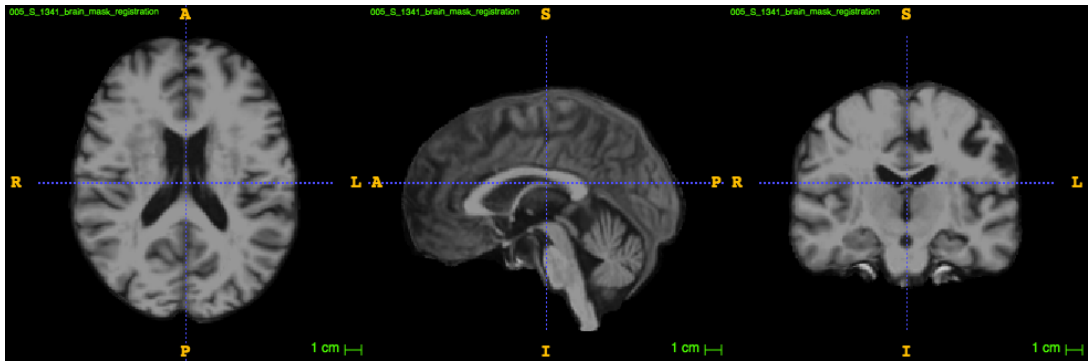
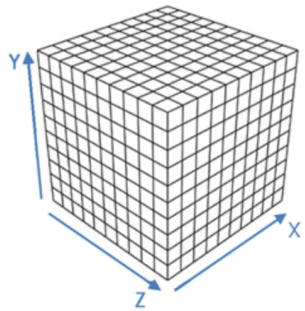


Figure 1: T1 Weighted MRI from ADNI



Big Data

- Mode MRI: $197 \times 233 \times 189$
- Mode fMRI: $79 \times 95 \times 79 \times 193$, take $>$ 130GB for 70 scans.
- PET, EEG,....

Deep Learning

Deep Learning

- Deep learning profoundly impacts science and society due to its impressive empirical success driven primarily by copious amounts of datasets and ever increasing computational resources.
- The key characteristic of deep learning is that accuracy empirically scales with the size of the model and the amount of training data.
- Neural networks with billions of parameters now typify state-of-the-art models.

Deep Learning

Figure 2: Genie of Aladdin

Statistical Framework of Neural network

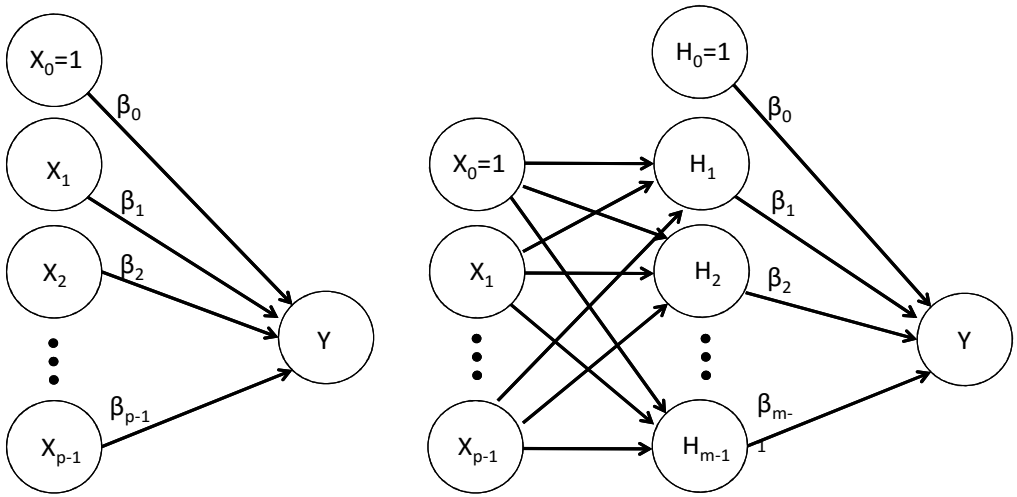


Figure 3: Regression vs. Neural network

Statistical Framework of Deep Neural network

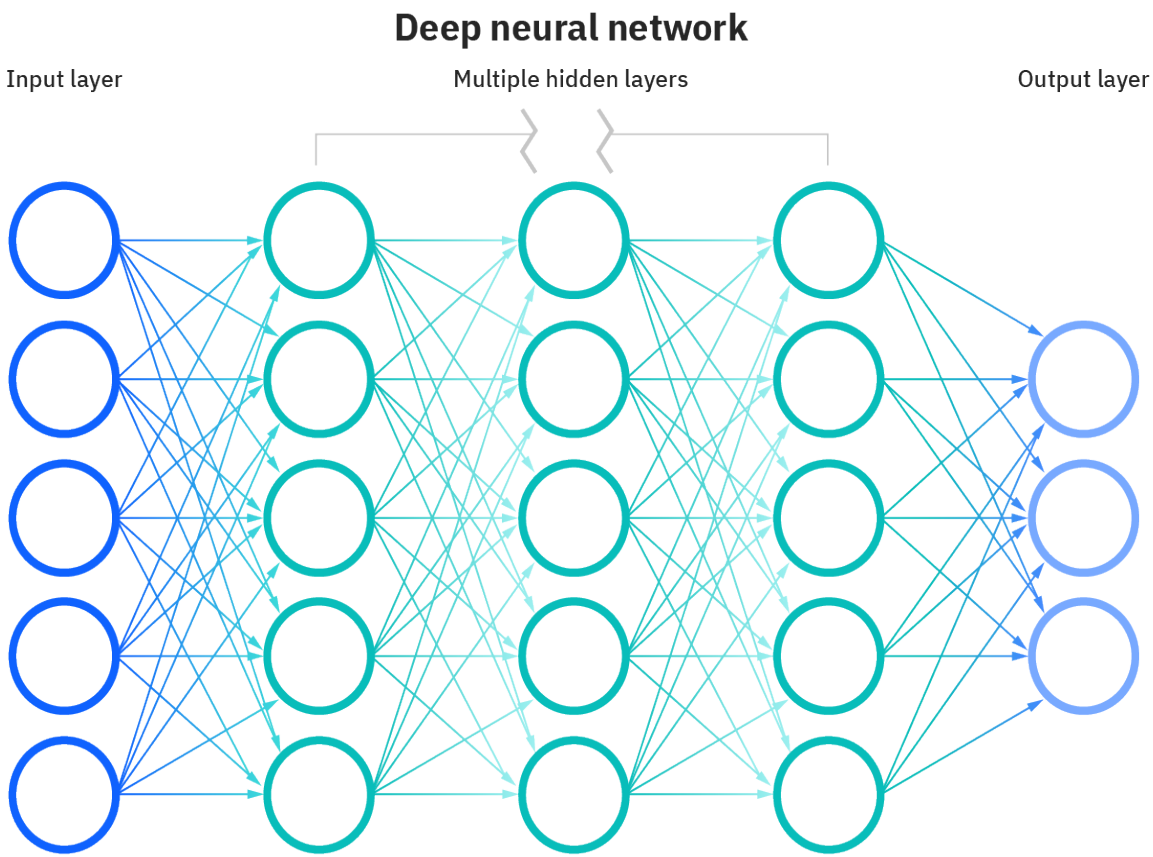


Figure 4: Deep Neural network

Statistical Framework of Deep Neural Network

- Consider the following conditional distribution of Y given $X = \boldsymbol{x}$

$$f_0(y|\boldsymbol{x}) = \exp\left[h_1(\eta_0(\boldsymbol{x}))y + h_2(\eta_0(\boldsymbol{x})) + h_3(y)\right] \tag{1}$$

  where $\eta_0(\cdot) : \mathcal{X} \to \mathbb{R}$ is a continuous function satisfying certain regularity assumptions and $\mathcal{X}$ is usually a compact subspace of $\mathbb{R}^p$.
- The functions h_1, h_2, h_3 are pre-determined and different choices give rise to different families of generalized linear models.
- For $h_1(u) = u$, $h_2(u) = -\log(1 + e^u)$, $h_3(y) = 1$, we get the classification model.
- For $h_1(u) = u$, $h_2(u) = -u^2$, $h_3(y) = -y^2/2 - \log(2\pi)/2$, we get the regression model with $\sigma^2 = 1$.
- Let $P_{X,Y}$ be joint distribution of (X, Y).

- In practice, the Bayes estimator is not useful since the function $\eta_0(\boldsymbol{x})$ is unknown.
- An estimator is obtained based on the training data, $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n)\}$ drawn from $P_{X,Y}$.
- A good estimator enjoys universal consistency properties, i.e., its risk approaches Bayes risk as $n \to \infty$ irrespective of P_X.
- To find this optimal class, we shall use deep neural networks (DNNs) as an approximation to $\eta_0(\boldsymbol{x})$.

Statistical Framework for Deep Neural network

- Consider a DNN with L hidden layers with $k_1, \cdots, k_L$ as the number of nodes. Also, $k_0 = p$ and $k_{L+1} = 1$. With $\overline{\boldsymbol{W}}_l = (\boldsymbol{v}_l, \boldsymbol{W}_l)$, define

$$\eta_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{v}_L + \boldsymbol{W}_L \psi(\boldsymbol{v}_{L-1} + \boldsymbol{W}_{L-1} \psi(\cdots \psi(\boldsymbol{v}_1 + \boldsymbol{W}_1 \psi(\boldsymbol{v}_0 + \boldsymbol{W}_0 \boldsymbol{x}))) \quad (2)$$

  $\boldsymbol{v}_l$: $k_{l+1} \times 1$ vectors and $\boldsymbol{W}_l$: $k_{l+1} \times k_l$ matrices, $l = 0, \cdots, L$;
  ψ is an activation function.

- $\boldsymbol{\theta} = \{\overline{\boldsymbol{W}}_0, \ldots, \overline{\boldsymbol{W}}_L\}$ denote all the parameters in the DNN model.
- # parameters: $K = \prod_{l=0}^{L} k_{l+1}(k_l + 1)$

Statistical Framework for Deep Neural network

- Approximating the true function $\eta_0(\boldsymbol{x})$, the density of $Y|\boldsymbol{X} = \boldsymbol{x}$ is

$$f_{\boldsymbol{\theta}}(y|\boldsymbol{x}) = \exp\left[h_1(\eta_{\boldsymbol{\theta}}(\boldsymbol{x}))y + h_2(\eta_{\boldsymbol{\theta}}(\boldsymbol{x})) + h_3(y)\right]. \tag{3}$$

- The likelihood function for the data $\mathcal{D}$ under the model and the truth is

$$P^n_{\boldsymbol{\theta}_n} = \prod_{i=1}^{n} f_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i), \qquad P^n_0 = \prod_{i=1}^{n} f_0(y_i|\boldsymbol{x}_i). \tag{4}$$

- For theoretical development, we shall assume $P_{\boldsymbol{X}} = U[0,1]^p$ and $\sigma_e^2 = 1$ and ψ is any $1-$Lipschitz continuous activation function.

Bayesian Inference

To start with, assume an independent normal prior (like regression model)

$$p(\boldsymbol{\theta}_n) = \prod_{j=1}^{K_n} \frac{1}{\sqrt{2\pi\sigma_{jn}^2}} e^{-\frac{1}{2\sigma_{jn}^2}(\theta_{jn}-\mu_{jn})^2} \tag{5}$$

Posterior distribution of $\boldsymbol{\theta}_n$ given $\boldsymbol{y}_n = [y_1, \cdots, y_n]^\top$ and $\boldsymbol{X}_n = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]^\top$

$$\pi(\boldsymbol{\theta}_n|\boldsymbol{y}_n, \boldsymbol{X}_n) = \frac{\pi(\boldsymbol{\theta}_n, \boldsymbol{y}_n, \boldsymbol{X}_n)}{\pi(\boldsymbol{y}_n, \boldsymbol{X}_n)} = \frac{L(\boldsymbol{\theta}_n)p(\boldsymbol{\theta}_n)}{\int L(\boldsymbol{\theta}_n)p(\boldsymbol{\theta}_n)d\boldsymbol{\theta}_n} \tag{6}$$

How do we make the inference? Popular MCMC?
Alternative methods include generative adversarial networks (GAN), (Goodfellow et al. (2014)), variational auto-encoders (Kingma and Welling (2013)), Normalizing Flows, (Rezende and Mohamed (2015)), etc.

Variational Bayes

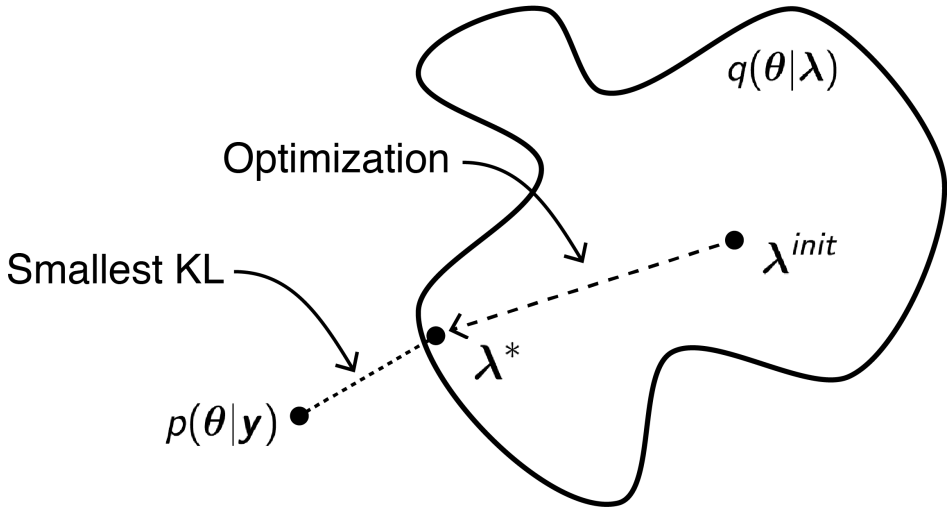
Variational Bayes (VB) recasts statistical inference as an optimization problem. Among predetermined family of distributions (called variational family), VB find the optimal approximation to the true posterior.

$$\mathcal{Q} = \{q(\theta) : q \text{ is a probability function}\}.$$

The "best" distribution in $\mathcal{Q}$ (called variational posterior) denoted by q^* is the minimizer of Kullback-Leibler (KL) divergence with $\pi(\theta \mid \boldsymbol{x})$:

$$q^* = \arg\min_{q \in \mathcal{Q}} \mathrm{KL}\left(q(\theta) \mid\mid \pi(\theta \mid \boldsymbol{x})\right).$$

Variational Bayes Inference - Overview



Variational inference (VI)

- θ_{jn} : independent with mean and standard deviation equal to m_{jn} and s_{jn} respectively.
- Variational family :

$$\mathcal{Q}_n = \left\{ q(\boldsymbol{\theta}_n) : q(\boldsymbol{\theta}_n) = \prod_{j=1}^{K_n} \frac{1}{\sqrt{2\pi s_{jn}^2}} e^{-\frac{1}{2s_{jn}^2}(\theta_{jn} - m_{jn})^2} \right\} \tag{7}$$

Popularized by Blei *et al.* (2017, 2019, JASA)

Variational posterior

- Reduce the KL-distance between the variational family and the true posterior. Blei and Lafferty (2007); Hinton and Van Camp (1993); Blundell et al. (2015).
- The variational posterior :

$$\pi^* = \underset{q \in \mathcal{Q}_n}{\operatorname{argmin}} \; d_{\mathrm{KL}}(q, \pi(.|\boldsymbol{y}_n, \boldsymbol{X}_n)). \tag{8}$$

Evidence Lower Bound:ELBO

- Bases on the model, simplifying further, we get

$$d_{\mathrm{KL}}(q, \pi(.|\boldsymbol{y}_n, \boldsymbol{X}_n)) = -\mathrm{ELBO}(q, \pi(., \boldsymbol{y}_n, \boldsymbol{X}_n)) + \log \pi(\boldsymbol{y}_n, \boldsymbol{X}_n) \quad (9)$$

- *Moving to optimization-based approach as opposed to popular sampling-based approach.*
- Black-Box Variational Bayes, noisy gradient, gradient decent; Python-TensorFlow-PyTorch, Adam are becoming common practice.

Handling ELBO with Black-Box

$$\begin{aligned}\text{ELBO}(q(.|\mathcal{V}_q), \pi(., \boldsymbol{y}_n, \boldsymbol{X}_n)) &= \int [\log \pi(\boldsymbol{\theta}_n, \boldsymbol{y}_n, \boldsymbol{X}_n) - \log q(\boldsymbol{\theta}_n|\mathcal{V}_q)] q(\boldsymbol{\theta}_n|\mathcal{V}_q) d\boldsymbol{\theta}_n \\ &= \int \log L(\boldsymbol{\theta}_n) q(\boldsymbol{\theta}_n|\mathcal{V}_q) d\boldsymbol{\theta}_n + \int [\log p(\boldsymbol{\theta}_n) - \log q(\boldsymbol{\theta}_n|\mathcal{V}_q)] q(\boldsymbol{\theta}_n|\mathcal{V}_q) d\boldsymbol{\theta}_n \\ &= \int \log L(\boldsymbol{\theta}_n) q(\boldsymbol{\theta}_n|\mathcal{V}_q) d\boldsymbol{\theta}_n - d_{\text{KL}}(q(.|\mathcal{V}_q), p(.)) = \mathcal{L}_{\mathcal{V}_q} - d_{\text{KL}}(q(.|\mathcal{V}_q), p(.))\end{aligned} \tag{10}$$

$$\begin{aligned}\nabla_{\mathcal{V}_q} \mathcal{L}_{\mathcal{V}_q} &= \nabla_{\mathcal{V}_q} \int \log L(\boldsymbol{\theta}_n) q(\boldsymbol{\theta}_n|\mathcal{V}_q) d\boldsymbol{\theta}_n = \int \log L(\boldsymbol{\theta}_n) \nabla_{\mathcal{V}_q} q(\boldsymbol{\theta}_n|\mathcal{V}_q) d\boldsymbol{\theta}_n \\ &= \int \nabla_{\mathcal{V}_q} \log q(\boldsymbol{\theta}_n|\mathcal{V}_q) \log L(\boldsymbol{\theta}_n) q(\boldsymbol{\theta}_n|\mathcal{V}_q) d\boldsymbol{\theta}_n = E_{q(.|\mathcal{V}_q)}(\log q(\boldsymbol{\theta}_n|\mathcal{V}_q) \log L(\boldsymbol{\theta}_n))\end{aligned} \tag{11}$$

$$\widehat{\nabla_{\mathcal{V}_q} \mathcal{L}}_{\mathcal{V}_q} = \frac{1}{W} \sum_{w=1}^{W} \nabla_{\mathcal{V}_q} \log q(\boldsymbol{\theta}_n[w]|\mathcal{V}_q) \log L(\boldsymbol{\theta}_n[w]) \tag{12}$$

BBVI

Algorithm 1 BBVI

1. Fix an initial value for variational family parameters $\mathcal{V}_q^1$.
2. Fix a step size sequence ρ_t, $t = 1, \cdots$.
3. Set $t = 1$.
4. Simulate W samples $\boldsymbol{\theta}_n[1], \cdots, \boldsymbol{\theta}_n[W]$ from $q(.|\mathcal{V}_q^t)$.
5. Compute $\widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t}$ as in (12)
6. Update
$$\mathcal{V}_q^{t+1} = \mathcal{V}_q^t + \rho_t \left(\widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t} - \nabla_{\mathcal{V}_q^t} d_{\mathrm{KL}}(q(.|\mathcal{V}_q), p(.)) \right) \tag{13}$$
7. Set $t = t + 1$.
8. Repeat steps 4-7 until the convergence of ELBO and
$$\text{ELBO} = \widehat{\mathcal{L}}_{\mathcal{V}_q^t} - d_{\mathrm{KL}}(q(.|\mathcal{V}_q), p(.))$$

Control Variate: Stabilizing the stochastic gradient

- A major drawback of algorithm 1 : the noisy estimator of the gradient has high variance.
- Control Variate (CV) Ross (2013) : reduce the variance of the MC approximations of the gradients.
- To reduce the variance of the function $A(x)$, then considers the function $B(x) = A(x) - c\left(\varphi(x) - E_q(\varphi(x))\right)$, where $\varphi(x)$ is function with finite expectation and c is a scalar. Such a choice ensures $E_q(B(x)) = E_q(A(x))$ and $\mathrm{Var}_q(B(x)) = \mathrm{Var}_q(A(x)) + c^2\mathrm{Var}_q(\varphi(x)) - 2c\mathrm{Cov}_q(A(x), \varphi(x))$ which is minimized at $c^\star = \mathrm{Cov}_q(A(x), \varphi(x))/\mathrm{Var}_q(\varphi(x))$.

BBVI-CV

Algorithm 2 BBVI-CV

1. Fix an initial value for variational parameter $\mathcal{V}_q^1$.
2. Fix a step size sequence ρ_t, $t = 1, \cdots$.
3. Set $t = 1$.
4. Simulate W samples $\boldsymbol{\theta}_n[1], \cdots, \boldsymbol{\theta}_n[W]$ from $q(.|\mathcal{V}_q^t)$.
5. Compute $c^{\star t} = \text{cov}(\boldsymbol{u}_1^t, \boldsymbol{u}_2^t)/\text{var}(\boldsymbol{u}_2^t)$ where $\boldsymbol{u}_1^t$ and $\boldsymbol{u}_2^t$ are same as in (18) and (19).
6. Compute $\widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t}$
7. Update
$$\mathcal{V}_q^{t+1} = \mathcal{V}_q^t + \rho_t \left(\widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t} - \nabla_{\mathcal{V}_q^t} d_{\text{KL}}(q(.|\mathcal{V}_q), p(.)) \right)$$
8. Set $t = t + 1$.
9. Repeat steps 4-7 until the convergence of ELBO and $\widehat{\quad}$

RMSprop Learning Rate: Stabilizing the learning rate

- Gradient decent:

$$\mathcal{V}_q^{t+1} = \mathcal{V}_q^t + \rho_t \left(\widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t} - \nabla_{\mathcal{V}_q^t} d_{\mathrm{KL}}(q(.|\mathcal{V}_q), p(.)) \right) \tag{14}$$

- Compute gradient and running average :

$$G_t = \widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t} - \nabla_{\mathcal{V}_q^t} d_{\mathrm{KL}}(q(.|\mathcal{V}_q), p(.))$$

$$R_t = 0.9 R_{t-1} + 0.1 G_t^2$$

- RMSprop update :

$$\mathcal{V}_q^{t+1} = \mathcal{V}_q^t + \rho_t \frac{G_t}{\sqrt{R_t} + \epsilon}$$

BBVI-RMS

Algorithm 3 BBVI-RMS

1. Fix an initial value for variational family parameters $\mathcal{V}_q^1$.
2. Fix a step size sequence ρ_t, $t = 1, \cdots$.
3. Set $t = 1$ and $\epsilon > 0$.
4. Simulate W samples $\boldsymbol{\theta}_n[1], \cdots, \boldsymbol{\theta}_n[W]$ from $q(.|\mathcal{V}_q^t)$.
5. Compute $\widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t}$ as in (12)
6. Compute
$$G_t = \widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t} - \nabla_{\mathcal{V}_q^t} d_{\mathrm{KL}}(q(.|\mathcal{V}_q), p(.))$$
$$R_t = 0.9 R_{t-1} + 0.1 G_t^2$$
7. Update
$$\mathcal{V}_q^{t+1} = \mathcal{V}_q^t + \rho_t \frac{G_t}{\sqrt{R_t} + \epsilon} \tag{15}$$
8. Set $t = t + 1$

BBVI-CV-RMS

Algorithm 4 BBVI-CV-RMS

1. Fix an initial value for variational parameter $\mathcal{V}_q^1$.
2. Fix a step size sequence ρ_t, $t = 1, \cdots$.
3. Set $t = 1$.
4. Simulate W samples $\boldsymbol{\theta}_n[1], \cdots, \boldsymbol{\theta}_n[W]$ from $q(.|\mathcal{V}_q^t)$.
5. Compute $c^{\star t} = \text{cov}(\boldsymbol{u}_1^t, \boldsymbol{u}_2^t)/\text{var}(\boldsymbol{u}_2^t)$ where $\boldsymbol{u}_1^t$ and $\boldsymbol{u}_2^t$
6. Compute $\widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t}$
7. Compute
$$G_t = \widehat{\nabla_{\mathcal{V}_q^t} \mathcal{L}}_{\mathcal{V}_q^t} - \nabla_{\mathcal{V}_q^t} d_{\text{KL}}(q(.|\mathcal{V}_q), p(.))$$
$$R_t = 0.9R_{t-1} + 0.1G_t^2$$
8. Update
$$\mathcal{V}_q^{t+1} = \mathcal{V}_q^t + \rho_t \frac{G_t}{\sqrt{R_t}}$$

Optimization impacts performance

- Other adaptive learning rates like AdaGrad, AdaDelta, ADAM, etc. serve the same purpose as that of RMSprop (see McMahan (2017) and Duchi et al. (2011)).
- Extensively studied the sensitivity to number of Monte Carlo samples, use of control variates and adapting learning rates, e.g., the RMSprop based gradient descent method.

Simulation study

Scenario 1: We simulate $n = 3000$ observations from a 2-2-2-1 network.

$$y_i = \begin{cases} 0, & \boldsymbol{b}_2 + \boldsymbol{A}_2\psi(\boldsymbol{b}_1 + \boldsymbol{A}_1(\psi(\boldsymbol{b}_0 + \boldsymbol{A}_0\boldsymbol{x}_i))) > 0 \\ 1, & \text{otherwise} \end{cases}$$

where $\boldsymbol{x}_i \in \mathbb{R}^2$, are i.i.d. from $N(0,1)$ and entries in $\boldsymbol{b}_j, \boldsymbol{A}_j, j = 0,1,2$ are i.i.d. from $U(0,1)$.

Scenario 2: We simulate $n = 3000$ observations from the following non linear function as

$$y_i = \begin{cases} 0, & 2e^{x_i[1]} + 3\sin(x_i[2]x_i[3]) + 4x_i[4]^3 - 3 > 0 \\ 1, & \text{otherwise} \end{cases}$$

where $\boldsymbol{x}_i \in \mathbb{R}^4$ are i.i.d. from $N(0,1)$.

Performance of algorithms algorithms 1, 2, 3, 4 for scenario 1.

			Testing accuracy(%)		Convergence time(s)	
Layers	Method	Sample size (S)	Fixed	RMSprop	Fixed	RMSprop
1	BBVI	200	97.41 ± 0.50	96.89 ± 0.93	23	114
		500	97.72 ± 0.38	97.52 ± 0.74	55	106
		1000	98.01 ± 0.33	97.38 ± 0.39	108	80
	BBVI-CV	200	97.82 ± 0.40	97.61 ± 0.60	21	6
		500	97.84 ± 0.40	97.67 ± 0.34	52	7
		1000	97.84 ± 0.42	97.94 ± 0.40	104	10
2	BBVI	200	97.79 ± 0.71	97.02 ± 1.10	200	98
		500	94.34 ± 3.82	97.75 ± 0.95	452	39
		1000	91.50 ± 5.17	98.11 ± 0.42	904	65
	BBVI-CV	200	96.34 ± 0.75	97.61 ± 0.44	118	17
		500	96.33 ± 0.73	97.30 ± 0.60	272	23
		1000	96.36 ± 0.74	97.74 ± 0.54	552	40

Table 1: Performance of algorithms algorithms 1, 2, 3, 4 for scenario 1.

Performance of algorithms 1, 2, 3, 4 for scenario 2

Layers	Method	Sample size (S)	Testing accuracy(%) Fixed	Testing accuracy(%) RMSprop	Convergence time(s) Fixed	Convergence time(s) RMSprop
1	BBVI	200	83.66 ± 14.51	88.71 ± 7.12	190	15
		500	90.22 ± 0.54	90.32 ± 0.98	364	390
		1000	90.28 ± 0.75	90.41 ± 0.71	732	710
	BBVI-CV	200	90.51 ± 0.87	90.42 ± 0.64	17	19
		500	90.51 ± 0.87	90.65 ± 0.61	36	33
		1000	90.53 ± 0.91	90.78 ± 0.49	69	37
2	BBVI	200	88.40 ± 0.50	89.89 ± 0.88	256	421
		500	90.52 ± 0.38	90.48 ± 0.74	518	544
		1000	90.61 ± 0.33	90.32 ± 0.65	906	608
	BBVI-CV	200	90.62 ± 0.40	91.11 ± 0.58	444	11
		500	90.74 ± 0.40	90.98 ± 0.54	862	12
		1000	90.72 ± 0.42	91.12 ± 0.53	1646	13

Table 2: Performance of algorithms 1, 2, 3, 4 for scenario 2

Performance of algorithms 1, 2, 3, 4 for scenario 1 and scenario 2 for 3 layers

		Scenario 1		Scenario 2	
Method	S	Testing accuracy(%)	Time(s)	Testing accuracy(%)	Time(s)
BBVI-RMS	200	97.76 ± 0.87	218	84.68 ± 4.85	423
	500	97.65 ± 0.83	169	88.00 ± 5.56	631
	1000	98.21 ± 0.73	132	90.69 ± 0.67	714
BBVI-CV-RMS	200	96.23 ± 1.05	212	84.53 ± 8.90	33
	500	97.83 ± 0.81	166	88.28 ± 2.03	37
	1000	98.42 ± 0.72	124	89.33 ± 1.67	45

Table 3: Performance of algorithms 1, 2, 3, 4 for scenario 1 and scenario 2 for 3 layers.

Learning

- Black-Box is not a *scientifically* black-box.
- Various tuning parameters play critical role for model performance.
- Optimization is an integral part of modern statistical learning with DNN.

Theoretical Guarantee

- WLG, let $X_i \sim U[0,1]^p$, implies $f_0(\boldsymbol{x}) = 1$ and $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = 1$,
- Define the Hellinger neighborhood of the true density function $P_0 = f_0(y|\boldsymbol{x})$ as

$$\mathcal{H}_\nu = \{\boldsymbol{\theta} : d_{\mathrm{H}}(P_0, P_{\boldsymbol{\theta}}) < \nu\}, \tag{16}$$

  where $P_{\boldsymbol{\theta}} = f_{\boldsymbol{\theta}}(y|\boldsymbol{x})$ and the Hellinger distance, d_{H} is $d_{\mathrm{H}}(P_1, P_2) = \frac{1}{2}\sqrt{\int(\sqrt{dP_1} - \sqrt{dP_2})^2}$.
- Let $\mathcal{F}_n$ denote a sequence of sieves satisfying $\mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \cdots \subseteq \mathcal{F}_n \cdots \subseteq \Omega$ where Ω is the natural parameter space of Θ.

Theoretical Guarantee

- **Assumption 1:** For some $C_0 > 0$, $\epsilon_n^*, \epsilon_n \to 0$, $n\epsilon_n^{*2}, n\epsilon_n^2 \to \infty$, there exists a sequence of tests ϕ_n,

$$P_0^n \phi_n + \sup_{\boldsymbol{\theta} \in \mathcal{F}_n, d_H(P_0, P_\theta) \geq \epsilon_n^*} P_0^n (1 - \phi_n) \leq \exp(-C_0 n \epsilon_n^2)$$

- **Assumption 2:** The prior Π satisfies $\Pi(\mathcal{F}_n^c) \leq \exp(-C_0 n \epsilon_n^2)$ and for some $M_n \to \infty$, $C_1 > 0$, $\Pi(\mathcal{N}_{C_1 \epsilon_n^2 / M_n}) \geq \exp(-C_1 n \epsilon_n^2 / M_n)$.
- **Assumption 3:** *Suppose* $\exists$ *a* $Q \in \mathcal{Q}_n$ *and* $d_{\mathrm{KL}}(Q, P) + E_Q d_{\mathrm{KL}}(P_0, P_{\boldsymbol{\theta}}) \leq C_2 n \epsilon_n^2 / M_n$.

Theoretical Guarantee

- **Theorem 1:** *If assumptions 1 and 2 hold, then the posterior of* (6) *in* P_0^n *probability satisfies*

$$\Pi(\mathcal{H}_{\epsilon_n^*}|\mathcal{D}) \leq 2\exp(-(C_0 - 2C_1/M_n)n\epsilon_n^2), \;\; n \to \infty$$

- **Theorem 2:** *If assumption 3 holds in addition to assumptions 1 and 2, then the variational posterior of* (19) *in* P_0^n *probability satisfies* $\Pi^*(\mathcal{H}_{\epsilon_n^*}|\mathcal{D}) \leq (2C_1 + C_2)/(C_0 M_n), \;\; n \to \infty$

Learning

- We establish the statistical validation of BDNN and quantify the loss due to variational approximation.
- For a general DNN, *the convergence to risk of Bayes estimator occurs at the rate* $\epsilon_n^{2/3}$ *for the variational posterior in contrast to* ϵ_n *for the true posterior*.
- For detail, see Bhattacharya et al. (2023b), Bhattacharya and Maiti (2021), Bhattacharya et al. (2023a) and Liu et al. (2023).

Do we have time?

▸ Forward

Need Low Cost Machines

- Gigantic models such as OpenAI GPT-3 (175 Billion) now typify the state-of-the-art across multiple domains such as natural language processing, computer vision, speech recognition etc. Toxic Bird Pits
- However, this form of model scaling is exorbitantly prohibitive in terms of computational requirements, financial commitment, energy requirements etc.
- There are numerous scenarios where training and deploying such huge models is practically infeasible.
- Examples of such scenarios include federated learning, autonomous vehicles, robotics, recommendation systems where models have to be refreshed daily/hourly or in an online manner for optimal performance.

Building Low Cost Machines

Low cost DNN
Structurally sparse and compressed Deep networks

Low Cost Deep Neural Network

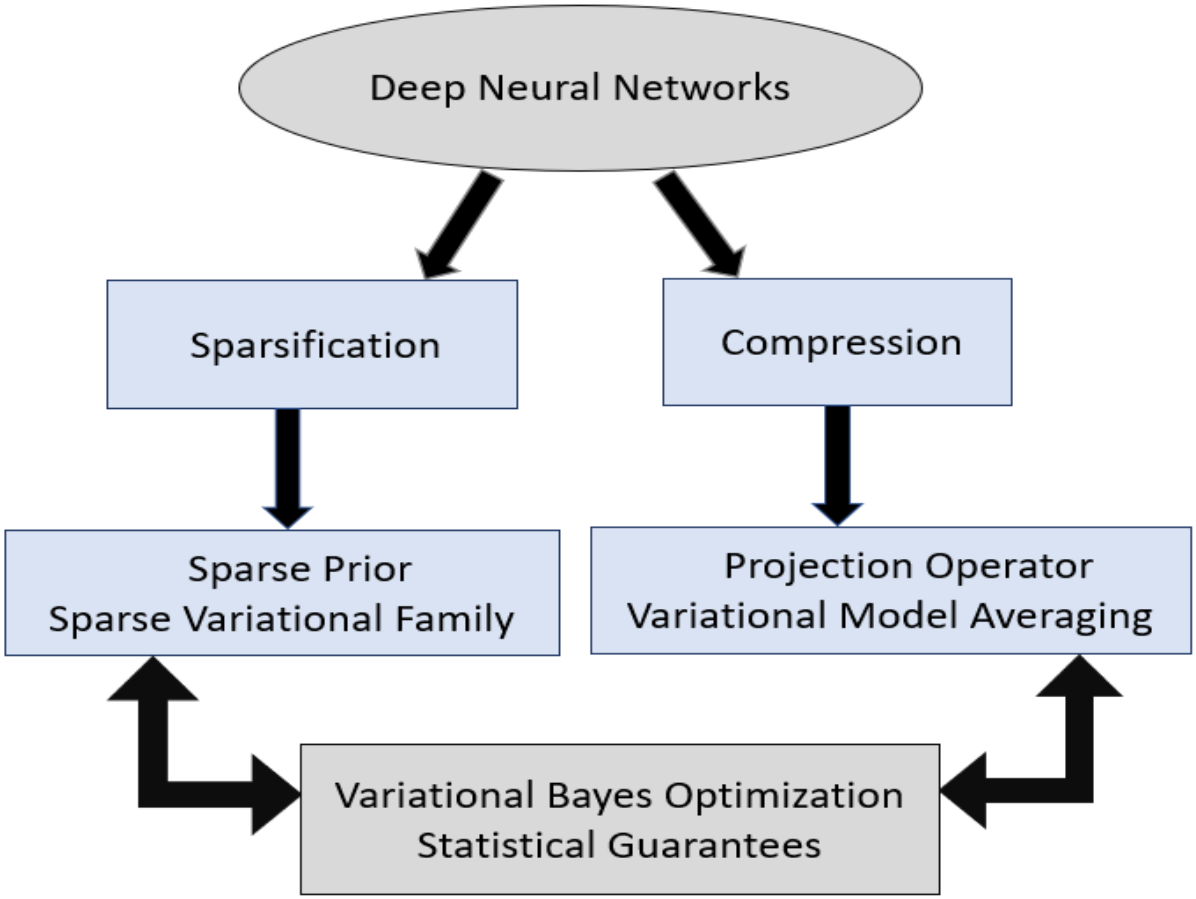


Figure 5: Low cost DNN

Sparse Networks

A promising avenue for improving the efficiency of DNNs is exploiting sparsity.

Model	ResNet-50 He et al. (2016)	EfficientNet-B0 Tan and Le (2019)	Transformer Gale et al. (2020)	Transformer B Vaswani et al. (20
Dataset	ImageNet	ImageNet	ImageNet-64x64	WMT EnDe
Efficiency	4.35×	3.32×	2.62×	1.3×
Sparsity Level	80%	90%	95%	97.5%
Sparsity Type	weights	weights	attention	ReLU

Table 4: Sparsity of common deep learning models

- Weight pruning approaches often result in non-structured sparse DNNs (Han et al., 2016; Molchanov et al., 2017; Zhu and Gupta, 2018; Frankle and Carbin, 2019). and lead to inefficient computational gains in practical setups (Wen et al., 2016).
- Designing algorithms for structurally sparsifying large networks is now gaining more attention Ghosh et al. (2019), Louizos et al. (2017).
- Current solutions to these problems are ad-hoc in nature and lack solid theoretical backing.
- A number of fundamental questions remain (1) Do structurally sparsified networks work well even at high levels of sparsification? (2) Can these networks be found equally efficient or statisticaly advantageous? (3) How do structurally sparse deep nets impact the statistical inference for function estimation and variable selection?

Sparse Deep Network

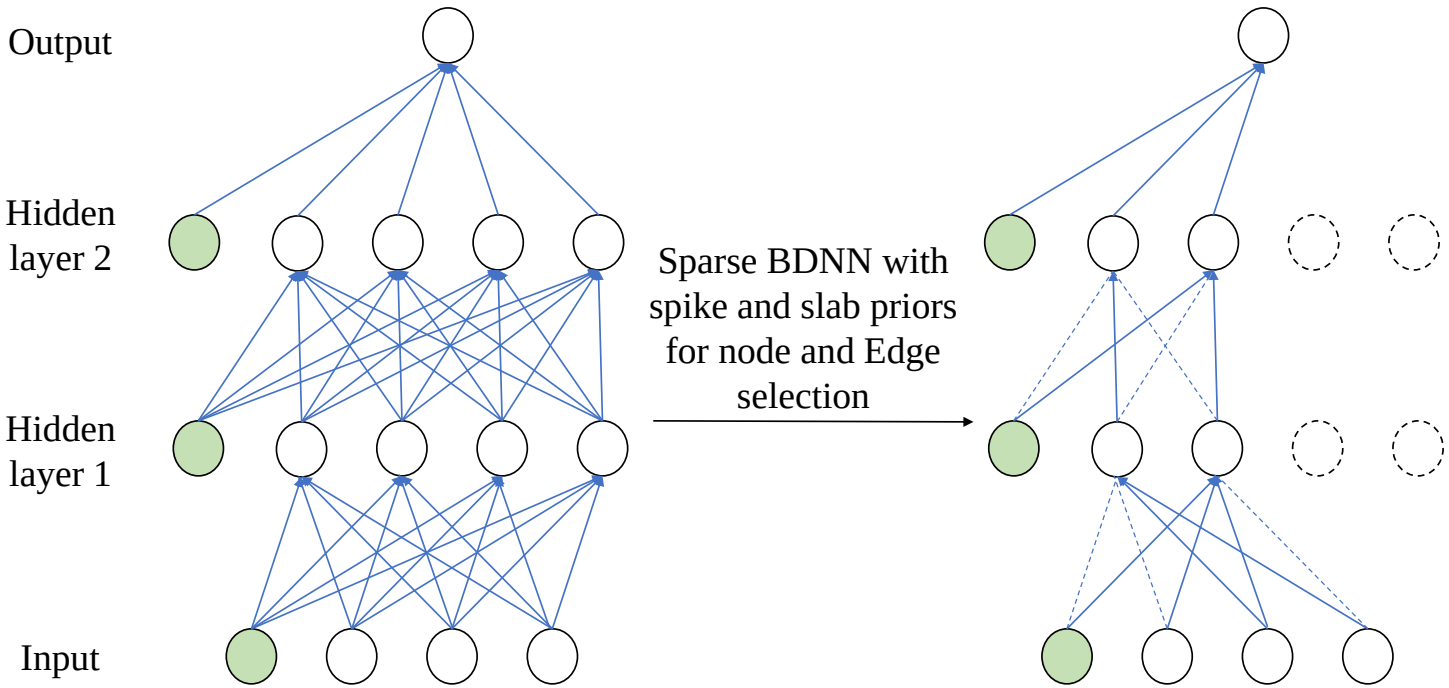


Figure 6: Spike and slab priors for simultaneous node and edge selection in dense network (left) leading to sparse network (right). Green nodes are bias nodes in input and hidden layers.

Structurally Sparse Deep Network

- $$f_0(y|\boldsymbol{x}) = \exp\left[h_1(\eta_0(\boldsymbol{x}))y + h_2(\eta_0(\boldsymbol{x})) + h_3(y)\right] \quad (17)$$
- With $\boldsymbol{W}_l = [\boldsymbol{w}_l^0, \boldsymbol{W}_l^1]$,

$$\eta_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{w}_L^0 + \boldsymbol{W}_L^1 \psi(\boldsymbol{w}_{L-1}^0 + \boldsymbol{W}_{L-1}^1 \psi(\cdots \psi(\boldsymbol{w}_1^0 + \boldsymbol{W}_1^1 \psi(\boldsymbol{w}_0^0 + \boldsymbol{W}_0^1 \boldsymbol{x})))), \quad (18)$$

  ψ: activation function, $\boldsymbol{w}_l^0$: $k_{l+1} \times 1$ vectors; $\boldsymbol{W}_l^1$: $k_{l+1} \times k_l$ matrices.
- The total number of parameters is $K = \prod_{l=0}^{L} k_{l+1}(k_l + 1)$.

Weight vs Structured Pruning

- Let $\overline{\boldsymbol{W}}_l^\top = \left[\overline{\boldsymbol{W}}_{l1}, \cdots, \overline{\boldsymbol{W}}_{lk_{l+1}}\right]$ be the row representation $\overline{\boldsymbol{W}}_l = (\boldsymbol{v}_l, \boldsymbol{W}_l)$, the $k_{l+1} \times (k_l + 1)$ matrix in l^{th} layer.
- To facilitate node selection, we assume the following prior on rows of $\overline{\boldsymbol{W}}_l$:

$$\overline{\boldsymbol{W}}_{li} | z_{li} \overset{\text{i.d.}}{\sim} (1 - z_{li})\delta_0(\overline{\boldsymbol{W}}_{li}) + z_{li} N(\boldsymbol{0}, \sigma_0^2 \boldsymbol{I}), z_{li} \overset{\text{i.d.}}{\sim} \text{Ber}(\nu_{l0}), \quad l = 0, \ldots, L.$$

- Note, the dependence of the prior selection probability ν_{l0} on l can allow for layer adaptive node selection if needed.
- When $z_{li} = 0$, $\overline{\boldsymbol{W}}_{li}$ is identically equal to 0, which in turn implies that there are no incoming edges onto node i of the neural network.

Variational Family

We posit the following mean field variational family ($\mathcal{Q}$)

$$\left\{ \overline{\boldsymbol{W}}_{li} | z_{li} \overset{\text{i.d.}}{\sim} (1 - z_{li})\delta_0(\overline{\boldsymbol{W}}_{li}) + z_{li} N(\boldsymbol{m}_{li}, \text{diag}(\boldsymbol{s}_{li}^2)), z_{li} \overset{\text{i.d.}}{\sim} \text{Ber}(\nu_{li}) \right\}$$

for $i = 1, \cdots, k_{l+1}, l = 0, \cdots, L$.

The above spike and slab variational family ensures that rows of the matrix $\overline{\boldsymbol{W}}_{li}$ corresponding to the insignificant nodes get dropped.

Variational posterior:

$$\pi^* = \underset{q \in \mathcal{Q}^{\mathbf{MF}}}{\operatorname{argmin}} \; d_{\mathrm{KL}}(q, \pi(|\mathcal{D})) \tag{19}$$

Note, the variational member q can be written as $q(\boldsymbol{\theta}, \boldsymbol{z}) = q(\boldsymbol{\theta}|\boldsymbol{z})q(\boldsymbol{z})$ where $q(\boldsymbol{z})$ is the probability mass function of $\boldsymbol{z}$ with respect to the counting measure and $q(\boldsymbol{\theta}|\boldsymbol{z})$ is the conditional density function with respect to the Lebesqgue measure of $\boldsymbol{\theta}$ given $\boldsymbol{z}$.

Variational posterior simplification

$$
\begin{aligned}
\pi^* &= \underset{q \in \mathcal{Q}^{\mathbf{MF}}}{\operatorname{argmin}} \sum_{z} \int [\log q(\boldsymbol{\theta}, \boldsymbol{z}) - \log \pi(\boldsymbol{\theta}, \boldsymbol{z} | \mathcal{D})] q(\boldsymbol{\theta}, \boldsymbol{z}) d\boldsymbol{\theta} \\
&= \underset{q \in \mathcal{Q}^{\mathbf{MF}}}{\operatorname{argmin}} \left(\sum_{z} \int [\log q(\boldsymbol{\theta}, \boldsymbol{z}) - \log \pi(\boldsymbol{\theta}, \boldsymbol{z}, \mathcal{D})] q(\boldsymbol{\theta}, \boldsymbol{z}) d\boldsymbol{\theta} + \log m(\mathcal{D}) \right) \\
&= \underset{q \in \mathcal{Q}^{\mathbf{MF}}}{\operatorname{argmin}} \; [-\mathrm{ELBO}(q, \pi(|\mathcal{D}))] + \log m(\mathcal{D}) = \underset{q \in \mathcal{Q}^{\mathbf{MF}}}{\operatorname{argmax}} \; \mathrm{ELBO}(q, \pi(|\mathcal{D}))
\end{aligned}
\tag{20}
$$

Since $\log m(\mathcal{D})$ is free from q, it suffices to maximize the evidence lower bound (ELBO) above.

Evidence Lower Bound

The ELBO cab be further simplified to

$$
\begin{aligned}
& - E_q[\log P_{\boldsymbol{\theta}}^n] + d_{\mathrm{KL}}(q, \pi) \\
& = -\mathbb{E}_{q(\boldsymbol{\theta}|z)q(z)}[\log P_{\boldsymbol{\theta}}^n] + d_{\mathrm{KL}}\left(q(\boldsymbol{\theta}|z)q(z), \pi(\boldsymbol{\theta}|z)\pi(z)\right) \\
& = -\mathbb{E}_{q(\boldsymbol{\theta}|z)q(z)}[\log P_{\boldsymbol{\theta}}^n] + \sum_{l,j} d_{\mathrm{KL}}(q(z_{lj})||\pi(z_{lj})) \\
& \quad + \sum_{l,j} [q(\boldsymbol{z}_{lj} = 1) d_{\mathrm{KL}}(q(\overline{\boldsymbol{w}}_{lj}|z_{lj} = 1)||\pi(\overline{\boldsymbol{w}}_{lj}|z_{lj} = 1)) + q(\boldsymbol{z}_{lj} = 0) d_{\mathrm{KL}}(q(\overline{\boldsymbol{w}}_{lj}|z_{lj} = 0)||\pi(\overline{\boldsymbol{w}}_{lj}|z_{lj} \\
& = -\mathbb{E}_{q(\boldsymbol{\theta}|z)q(z)}[\log P_{\boldsymbol{\theta}}^n] + \sum_{l,j} d_{\mathrm{KL}}(q(z_{lj})||\pi(z_{lj})) + \sum_{l,j} q(\boldsymbol{z}_{lj} = 1) d_{\mathrm{KL}}(q(\overline{\boldsymbol{w}}_{lj}|z_{lj} = 1)||\pi(\overline{\boldsymbol{w}}_{lj}|z_{lj} = 1 \\
& = -\mathbb{E}_{q(\boldsymbol{\theta}|z)q(z)}[\log P_{\boldsymbol{\theta}}^n] + \sum_{l,j} d_{\mathrm{KL}}(q(z_{lj})||\pi(z_{lj})) + \sum_{l,j} q(\boldsymbol{z}_{lj} = 1) d_{\mathrm{KL}}(N(\boldsymbol{\mu}_{lj}, \mathrm{diag}(\boldsymbol{\sigma}_{lj}^2))||N(0, \sigma_0^2 \boldsymbol{I})
\end{aligned}
$$

- The KL of discrete variables creates a challenge in practical implementation, Jang et al. (2017), Maddison et al. (2017).
- The continuous relaxation approximation is achieved through Gumbel-softmax (GS) distribution, that is $q(z_{lj}) \sim \text{Ber}(\gamma_{lj})$ is approximated by $q(\tilde{z}_{lj}) \sim \text{GS}(\gamma_{lj}, \tau)$, where

$$\tilde{z}_{lj} = (1 + \exp(-\eta_{lj}/\tau))^{-1}, \quad \eta_{lj} = \log(\gamma_{lj}/(1-\gamma_{lj})) + \log(u_{lj}/(1-u_{lj}))$$

$$u_{lj} \sim U(0,1)$$

where τ is the temperature.

Algorithm

Algorithm 5 Variational inference in SS-IG Bayesian neural networks

1. **Inputs:** training dataset, network architecture, and optimizer tuning parameters; *Model inputs*; *Variational inputs*
2. **REPEAT**
3. Generate S samples from $\boldsymbol{\zeta}_{lj} \sim N(0, \boldsymbol{I})$ and $u_{lj} \sim U(0,1)$
4. Generate S samples for $(z_{lj}, \tilde{z}_{lj})$ using u_{lj}
5. Use $\boldsymbol{\mu}_{lj}, \boldsymbol{\sigma}_{lj}, \boldsymbol{\zeta}_{lj}$ and z_{lj} to compute loss (ELBO) in forward pass
6. Use $\boldsymbol{\mu}_{lj}, \boldsymbol{\sigma}_{lj}, \boldsymbol{\zeta}_{lj}$ and $\tilde{z}_{lj}$ to compute gradient of loss in backward pass
7. Update the variational parameters with gradient of loss using stochastic gradient descent algorithm (e.g. Adam, Kingma and Ba (2015))
8. **UNTIL**change in ELBO $< \epsilon$

Numerical Validation -I

Data: 2-2-1; Model: 2-20-1

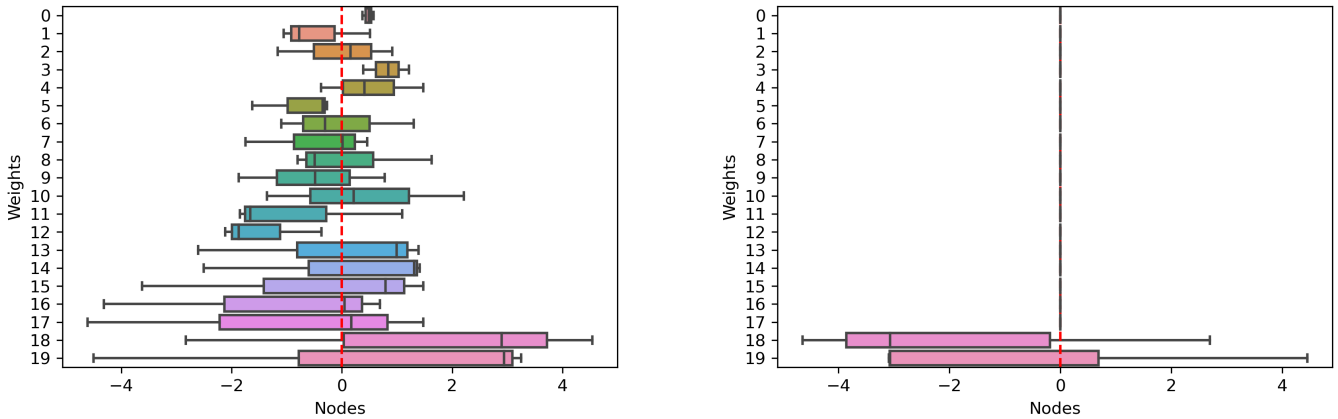


Figure 7: Node-wise weight magnitudes recovered by VBNN and proposed SS-IG model in the synthetic regression data generated using 2-2-1 network. The boxplots show the distribution of incoming weights into a given hidden layer node.

Numerical Validation -II

Consider a nonlinear regression example

$$y = \frac{7x_2}{1+x_1^2} + \sin(x_3 x_4) + 2x_5 + \varepsilon,$$

Table 5: Performance of the proposed SS-IG, SV-BNN, HS-BNN, and VBNN models

Model	Train RMSE	Test RMSE	Sparsity Estimate
SS-IG	1.2087±0.0490	1.1947±0.0587	(0.35,0.05)
SV-BNN	1.2897±0.0323	1.2760±0.0363	(0.45,0.35)
HS-BNN	1.2580±0.0305	1.2436±0.0394	(1.00, 1.00)
VBNN	1.1661±0.0335	1.1614±0.0349	NA

Numerical Validation -UCI Regression Datasets

Table 6: Results on UCI regression datasets

		Test RMSE				Sparsity Estimate	
Dataset	$n(k_0)$	SS-IG	SV-BNN	HS-BNN	VBNN	SS-IG	SV-BNN
Concrete	1030 (8)	7.92±0.68	8.22±0.70	5.34±0.53	7.34±0.62	0.42±0.06	0.98±0.02
Wine	1599 (11)	0.66±0.05	0.65±0.05	0.66±0.05	0.64±0.05	0.18±0.05	0.87±0.04
Power Plant	9568 (4)	4.28±0.20	4.32±0.19	4.34±0.18	4.27±0.17	0.18±0.03	0.24±0.03
Kin8nm	8192 (8)	0.09±0.00	0.11±0.01	0.10±0.00	0.09±0.00	0.43±0.04	0.47±0.04
Protein	45730 (9)	4.85±0.05	4.93±0.06	4.59±0.02	4.78±0.06	0.81±0.03	0.93±0.03
Year	515345 (90)	8.68±NA	8.78±NA	9.33±NA	8.67±NA	0.71±NA	0.78±NA

Image Classification: MNIST

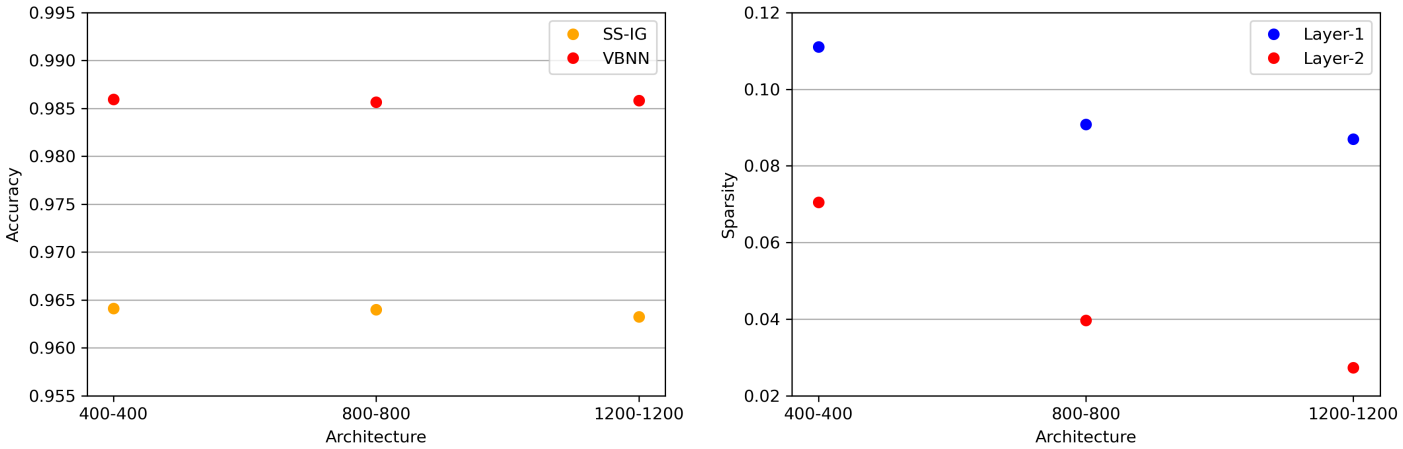


Figure 8: Left: Prediction accuracy per architecture; Right: Layer-wise sparsity per architecture

- 3 different architectures, 400, 800, and 1200 nodes each in 2-hidden layers.
- We train the models over 400 epochs with the help of Adam optimizer.

Image Classification: MNIST

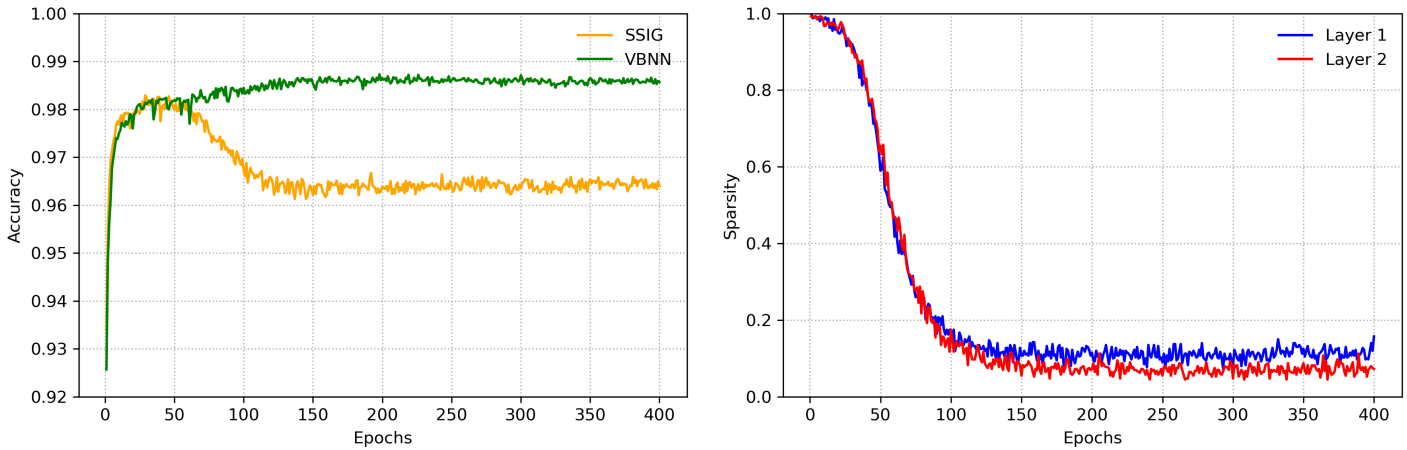


Figure 9: Left: Prediction accuracy over epochs; Right: Layer-wise sparsity over epochs

Our method is able to prune off more than 88% of first hidden layer nodes and more than 92% of second hidden layer nodes at the expense of 2% accuracy loss due to sparsification compared to the densely connected VBNN.

Theoretical Validation

- $\widetilde{\Pi}(.|\mathcal{D})$ denotes true marginal posterior and $\widetilde{\Pi}^*(.|\mathcal{D})$ denotes variational marginal posterior.
- Hellinger distance: $d^2_{\mathrm{H}}(P_0, P_{\boldsymbol{\theta}}) = \frac{1}{2}\int \left(\sqrt{f_{\boldsymbol{\theta}}(y|\boldsymbol{x})} - \sqrt{f_0(y|\boldsymbol{x})}\right)^2 dyd\boldsymbol{x}$.

Lemmas

Lemma 1 (Existence of Test Functions): Let $\epsilon_n \to 0$ and $n\epsilon_n^2 \to \infty$. There exists a testing function $\phi \in [0,1]$ and constants $C_1, C_2 > 0$,

$$\mathbb{E}_{P_0}(\phi) \leq \exp\{-C_1 n\epsilon_n^2\}$$

$$\sup_{\boldsymbol{\theta} \in \mathcal{H}_{\epsilon_n}^c,\ \eta_{\boldsymbol{\theta}} \in \mathcal{F}(L, \mathbf{m}k, \mathbf{m}s^\circ, \mathbf{m}B^\circ)} \mathbb{E}_{P_{\boldsymbol{\theta}}}(1-\phi) \leq \exp\{-C_2 n d_{\mathrm{H}}^2(P_0, P_{\boldsymbol{\theta}})\}$$

where $\mathcal{H}_{\epsilon_n} = \{\boldsymbol{\theta} : d_{\mathrm{H}}(P_0, P_{\boldsymbol{\theta}}) \leq \epsilon_n\}$ is the Hellinger neighborhood of radius ϵ_n.

Lemma 2 (Prior mass condition): Let $\epsilon_n \to 0$, $n\epsilon_n^2 \to \infty$ and $n\epsilon_n^2 / \sum_{l=0}^{L} u_l \to \infty$, then for $\widetilde{\Pi}$ and some constant $C_3 > 0$,

$$\widetilde{\Pi}(\mathcal{F}(L, \boldsymbol{k}, \boldsymbol{s}, \boldsymbol{B})^c) \leq \exp(-C_3 n\epsilon_n^2 / \sum_{l=0}^{L} u_l)$$

Lemma 3 (Kullback-Leibler conditions): Suppose $\sum_{l=0}^{L} r_l + \xi \to 0$ and $n(\sum_{l=0}^{L} r_l + \xi) \to \infty$ and the following two conditions hold for the prior $\widetilde{\Pi}$ and some $q \in \mathcal{Q}^{\mathbf{MF}}$

1. $\widetilde{\Pi}\left(\mathcal{N}_{\sum_{l=0}^{L} r_l + \xi}\right) \geq \exp(-C_4 n(\sum_{l=0}^{L} r_l + \xi))$
2. $d_{\mathrm{KL}}(q, \pi) + n \sum_{\boldsymbol{z}} \int d_{\mathrm{KL}}(P_0, P_{\boldsymbol{\theta}}) q(\boldsymbol{\theta}, \boldsymbol{z}) d\boldsymbol{\theta} \leq C_5 n(\sum_{l=0}^{L} r_l + \xi)$

where $\mathcal{N}_{\sum_{l=0}^{L} r_l + \xi}$ is the KL neighborhood of radius $\sum_{l=0}^{L} r_l + \xi$.

Main Theorem

Variational posterior consistency

Suppose Lemma 1 holds and Lemmas 2 and 3 hold for $\epsilon_n = \sqrt{(\sum_{l=0}^{L} r_l + \xi) \sum_{l=0}^{L} u_l}$. Then for some slowly increasing sequence $M_n \to \infty$, $M_n \epsilon_n \to 0$ and $\widetilde{\Pi}^*$,

$$\widetilde{\Pi}^*(\mathcal{H}^c_{M_n \epsilon_n}) \to 0, \quad n \to \infty \quad \text{in } P_0^n \text{ probability.}$$

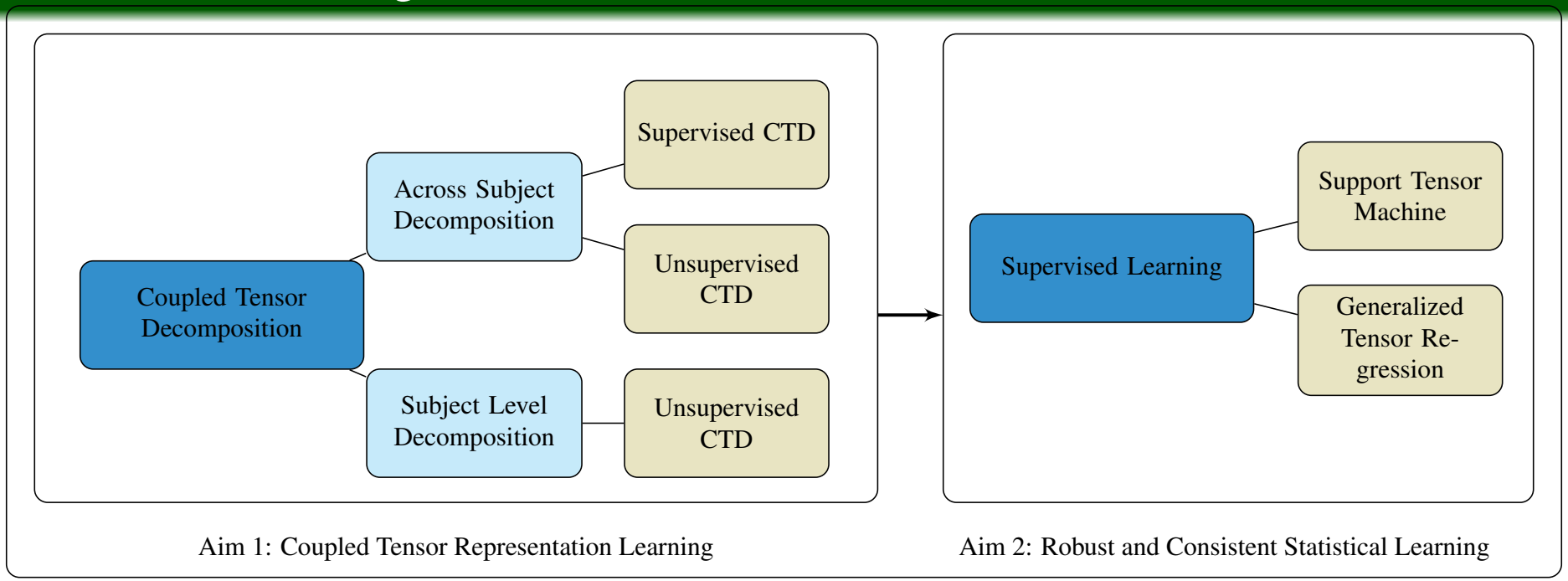
in P_0^n probability where $\mathcal{H}^c_{M_n \epsilon_n} = \{\boldsymbol{\theta} : d_{\mathrm{H}}(P_0, P_{\boldsymbol{\theta}}) \leq M_n \epsilon_n\}$ is the Hellinger neighborhood of radius $M_n \epsilon_n$.

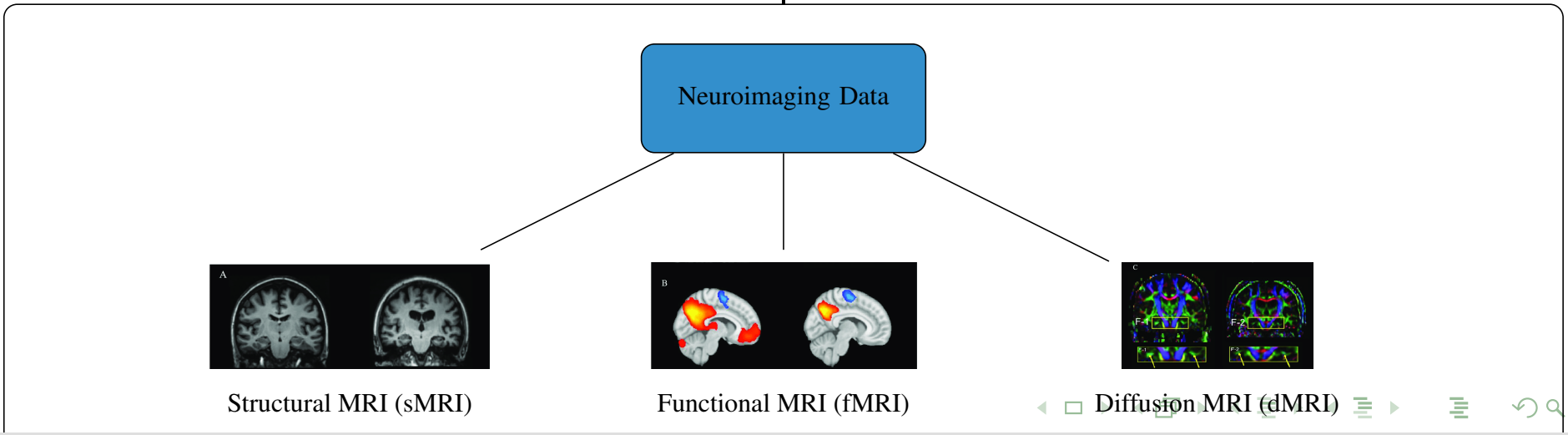
Related theoretical works: Polson and Ročková (2018), Chérief-Abdellatif (2020) and Bai et al. (2020); several results are from Schmidt-Hieber (2020).

Learning

- Increasingly use of deep learning in smart devices, designing resource efficient AI for real-time inference is of practical importance.
- Combining high-dimensional statistical techniques along with modern optimization, we develop trustworthy and computationally efficient deep learning that are applicable for a wide range of applications.
- Our method incorporates layer-wise prior inclusion probabilities and recovers underlying sparse model effectively.
- The sparse BNN can achieve comparable performance of a dense network.
- Our theoretical developments quantifies the loss due to variational method and trustworthy DNN.
- The *compression technique* is a different approach and has its own pros and cons.

Multimodal Learning





Big Data, Data Science, Machine Learning and Statistics

Big Data opens the door for statistics and machine learning

Bai, J., Song, Q., and Cheng, G. (2020). Efficient variational inference for sparse deep learning with theoretical guarantee. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 466–476. Curran Associates, Inc.

Bhattacharya, S., Liu, Z., and Maiti, T. (2023a). Consistent variational bayes classification with deep neural networks. *Technical report, Michigan State University*.

Bhattacharya, S., Liu, Z., and Maiti, T. (2023b). Variational bayes neural network: Posterior consistency, classification accuracy and computational challenges. *arXiv preprint arXiv:2011.09592*.

Bhattacharya, S. and Maiti, T. (2021). Statistical foundation of variational bayes neural networks. *Neural Networks*, 137:151–173.

Blei, D. M. and Lafferty, J. D. (2007). A correlated topic model of science. *The Annals of Applied Statistics*, 1(1):17–35.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight

uncertainty in neural network. In *Proceedings of Machine Learning Research*, volume 37, pages 1613–1622. PMLR.

Chérief-Abdellatif, B.-E. (2020). Convergence rates of variational inference in sparse deep learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML-2020)*, volume 119, pages 1831–1842, Vienna, Austria.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159.

Frankle, J. and Carbin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations (ICLR 2019)*, New Orleans, USA.

Gale, T., Zaharia, M., Young, C., and Elsen, E. (2020). Sparse gpu kernels for deep learning. *arXiv preprint arXiv:2006.10901*.

Ghosh, S., Yao, J., and Doshi-Velez, F. (2019). Model selection in bayesian neural networks via horseshoe priors. *Journal of Machine Learning Research*, (20):1–46.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Han, S., Mao, H., and Dally, W. J. (2016). Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *4th International Conference on Learning Representations (ICLR-2016)*, San Juan, Puerto Rico.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*.

Hinton, G. E. and Van Camp, D. (1993). Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory, COLT'93*, page 5–13. ACM press.

Jang, E., Gu, S., and Poole, B. (2017). Categorical reparameterization with

gumbel-softmax. In *5th International Conference on Learning Representations (ICLR 2017)*, Toulon, France.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*, San Diego, USA.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Liu, Z., Bhattacharya, S., and Maiti, T. (2023). Variational bayes compressed neural networks. *Technical report, Michigan State University*.

Louizos, C., Ullrich, K., and Welling, M. (2017). Bayesian compression for deep learning. In *Proceedings of the 30th Advances in Neural Information Processing Systems (NIPS 2020)*, pages 3288–3298, Long Beach, CA, USA.

Maddison, C. J., Mnih, A., and Teh, Y. W. (2017). The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations (ICLR 2017)*, Toulon, France.

McMahan, H. B. (2017). A survey of algorithms and analysis for adaptive online learning. *Journal of Machine Learning Research*, 18(90):1–50.

Molchanov, D., Ashukha, A., and Vetrov, D. (2017). Variational dropout sparsifies deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML-2027)*, volume 70, page 2498–2507, Sydney, NSW, Australia.

Polson, N. G. and Ročková, V. (2018). Posterior concentration for sparse deep learning. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1530–1538, Lille, France. PMLR.

Ross, S. M. (2013). *Simulation*. Academic Press, fifth edition.

Schmidt-Hieber (2020). Nonparametric regression using deep neural

networks with relu activation function. *Annals of Statistics*, pages 1875–1897.

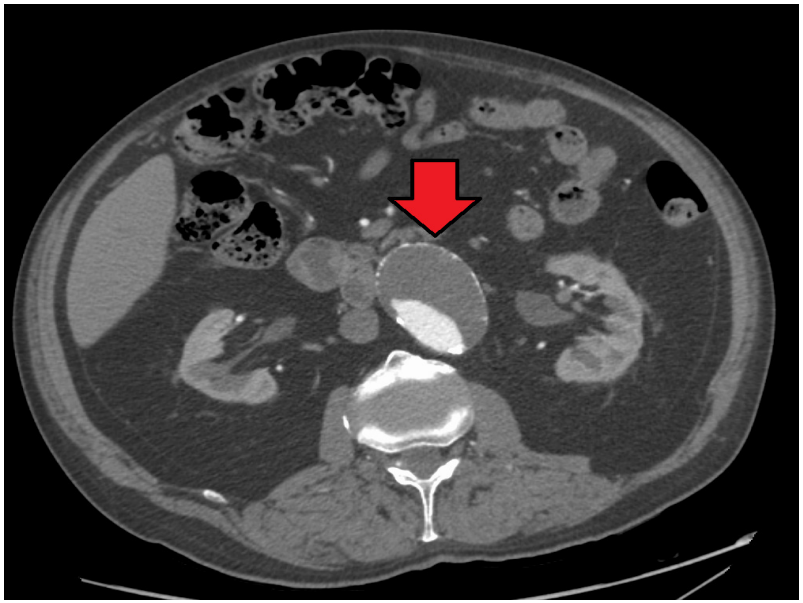
Tan, M. and Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR.

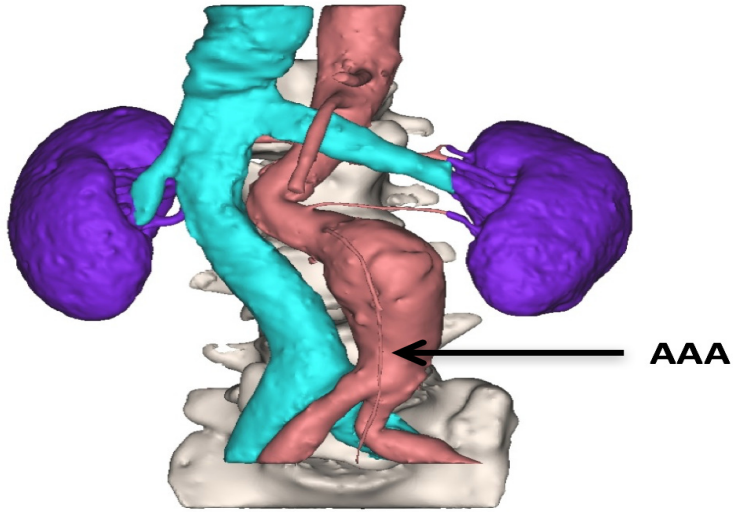
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Wen, W., Wu, C., Wang, Y., Chen, Y., and Li, H. (2016). Learning structured sparsity in deep neural networks. In *Proceedings of the 29th Advances in Neural Information Processing Systems (NIPS 2016)*, Barcelona, Spain.

Zhu, M. and Gupta, S. (2018). To prune, or not to prune: Exploring the efficacy of pruning for model compression. In *6th International Conference on Learning Representations (ICLR 2018), Workshop Track Proceedings*, Vancouver, Canada. OpenReview.net.

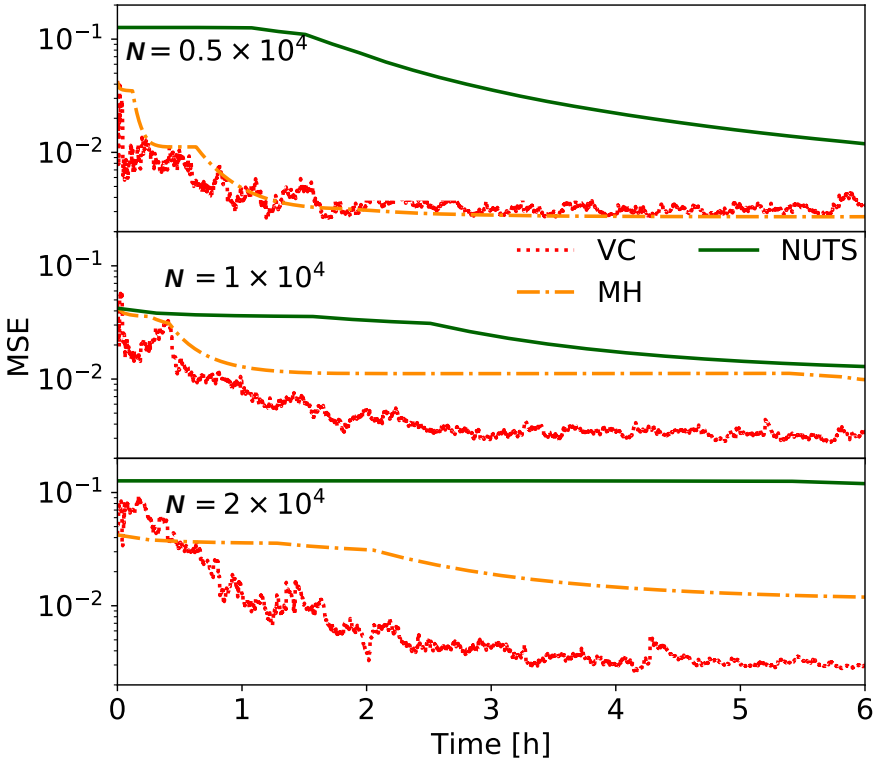
Preprocessing of CT scans

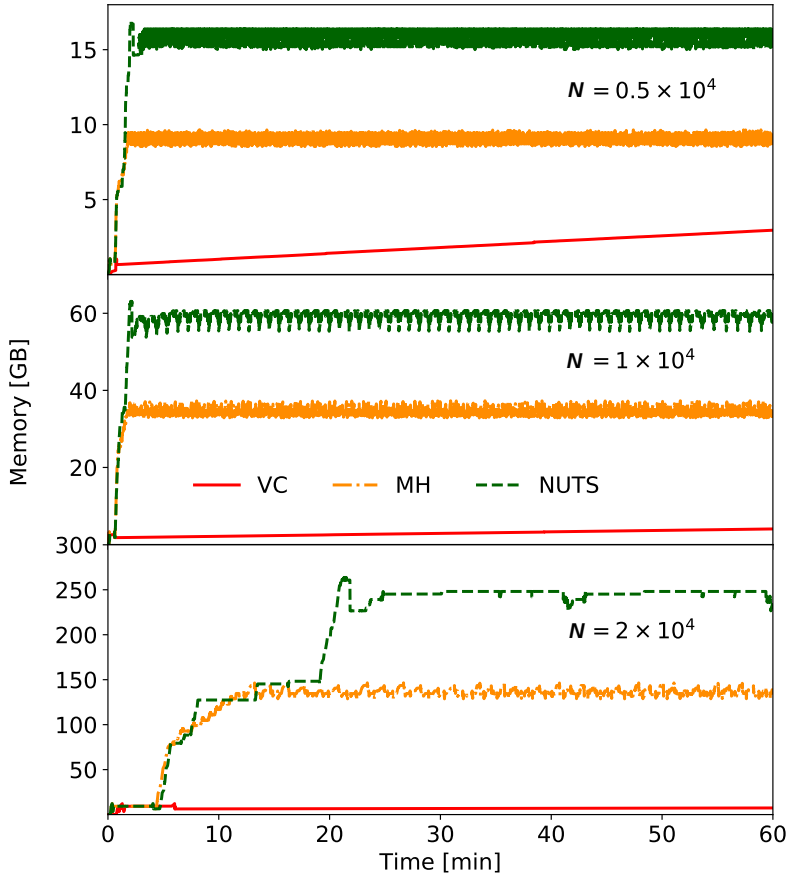




CT image: Abdominal Aortic Aneurysm.

Variational Bayes Inference - Calibration - Simulation Study





Variational Bayes Inference - Calibration - Simulation Study - Supplement

Comparison with the Metropolis-Hastings algorithm and No-U-Turn sampler: