Bayesian neural networks for sparse coding

Danil Kuzin [1] Olga Isupova [2] Lyudmila Mihaylova [1]

[1]Department of Automatic Control and Systems Engineering, University of Sheffield, UK

[2]Machine Learning Research Group, University of Oxford, UK

ICASSP 2019

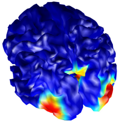
Outline

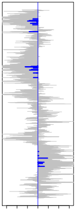
Problem

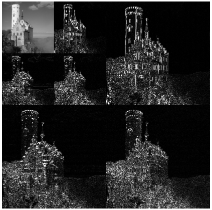
Linear regression

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

How to find the most meaningful components of $\boldsymbol{\beta}$?







Sparse methods

- variable selection
- sparse coding
- sparse approximations

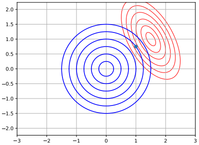
Challenges

- computationally intensive
- structural assumptions
- uncertainty estimation

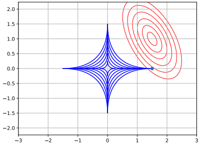
Sparse frequentist regression

Add sparsity-inducing penalty

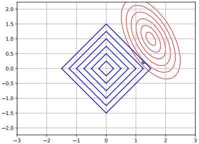
$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left[||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + ||\boldsymbol{\beta}||_p^p \right]$$



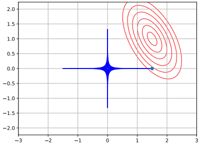
l_2-penalty



$l_{1/2}$-penalty



l_1-penalty

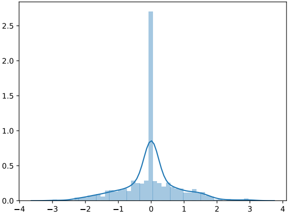


$l_{1/8}$-penalty

Sparse Bayesian regression

Add sparsity-inducing prior

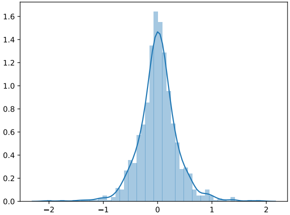
Strong sparsity



$$\beta_d \sim (1 - z_d)\mathcal{N}(0, \sigma^2) + z_d \delta_0$$
$$z_d \sim \mathrm{Ber}(\omega)$$

- probability of exact zero
- discrete variables

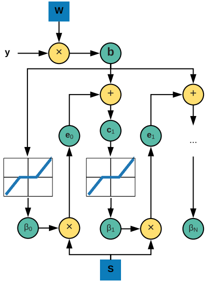
Weak sparsity



$$\beta_d \sim \mathcal{N}(0, \sigma_d^2)$$
$$\sigma_d^2 \sim \mathrm{IG}(a)$$

- continuous at zero
- continuous variables

LISTA



Problem

Estimate $\boldsymbol{\beta}$ from observations $\mathbf{y}$ collected as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, s.t. elements $\boldsymbol{\beta}$ contain zeros.

Original LISTA

- Represent iterative soft-thresholding algorithm as a recurrent neural network with shared weights
- Learn weights with backpropagation through time
- Overfitting
- No uncertainty estimation

Require: observation $\mathbf{y}$, current weights $\mathbf{W}$, $\mathbf{S}$, number of layers L

1: *Initialisation.* Dense layer $\mathbf{b} \leftarrow \mathbf{W}\mathbf{y}$
2: *Initialisation.* Soft-thresholding nonlinearity $\widehat{\boldsymbol{\beta}}_0 \leftarrow h_\lambda(\mathbf{b})$
3: **for** $l = 1$ **to** L **do**
4: Dense layer $\mathbf{c}_l \leftarrow \mathbf{b} + \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$
5: Soft-thresholding nonlinearity $\widehat{\boldsymbol{\beta}}_l \leftarrow h_\lambda(\mathbf{c}_l)$

BayesLISTA

- Add priors for NN weights

$$p(\mathbf{W}) = \prod_{d=1}^{D} \prod_{k=1}^{K} \mathcal{N}(w_{ij}; 0, \eta^{-1}), \quad p(\mathbf{S}) = \prod_{d'=1}^{D} \prod_{d''=1}^{D} \mathcal{N}(s_{d'd''}; 0, \eta^{-1}),$$

- Propagate distribution for $\widehat{\boldsymbol{\beta}}$ through layers
- Compute prediction as noisy NN output

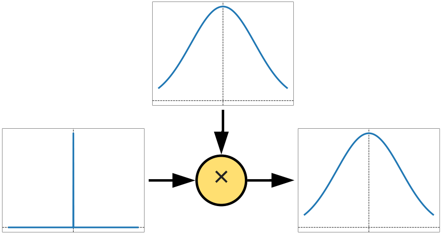
$$p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{W}, \mathbf{S}, \gamma, \lambda) = \prod_{d=1}^{D} \mathcal{N}\left(\beta_d; [f(\mathbf{y}; \mathbf{S}, \mathbf{W}, \lambda)]_d, \gamma^{-1}\right)$$

- Update weights with PBP

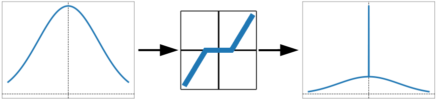
Uncertainty propagation

At every step the output of soft-thresholding can be closely approximated with the spike and slab distribution

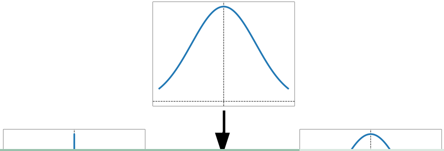
1. $\mathbf{b} = \mathbf{W}\mathbf{y}$ is Gaussian-distributed



2. $\widehat{\boldsymbol{\beta}}_0 = h_\lambda(\mathbf{b})$ is approximated with the spike and slab distribution



3. $\mathbf{e}_l = \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$ is approximated with the Gaussian distribution



BackProp-PBP

Approximate posterior

$$q(\mathbf{W}, \mathbf{S}, \gamma, \eta) = \prod_{d=1}^{D} \prod_{k=1}^{K} \mathcal{N}(w_{dk}; m^w_{dk}, v^w_{dk}) \prod_{d'=1}^{D} \prod_{d''=1}^{D} \mathcal{N}(s_{d'd''}; m^s_{d'd''}, v^s_{d'd''}) \\ \times \mathrm{Gam}(\gamma; a^\gamma, b^\gamma)\mathrm{Gam}(\eta; a^\eta, b^\eta)$$

Probabilistic backpropagation [HL&A]: use derivatives of the logarithm of a normalisation constant to update weight distributions

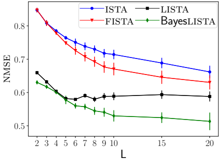
$$q(a) = Z^{-1} f(a) \mathcal{N}(a; m, v)$$

$$Z \approx \prod_{d=1}^{D} \left[\omega_d^{\widehat{\boldsymbol{\beta}}} \mathcal{T}\left(\beta_d; 0, \beta^\gamma/\alpha^\gamma, 2\alpha^\gamma\right) + \left(1 - \omega_d^{\widehat{\boldsymbol{\beta}}}\right) \mathcal{N}\left(\beta_d; m_d^{\widehat{\boldsymbol{\beta}}}, \beta^\gamma/(\alpha^\gamma - 1) + v_d^{\widehat{\boldsymbol{\beta}}}\right) \right],$$

where $\{\omega_d^{\widehat{\boldsymbol{\beta}}}, m_d^{\widehat{\boldsymbol{\beta}}}, v_d^{\widehat{\boldsymbol{\beta}}}\}$ are the parameters of the spike and slab distribution for $[\widehat{\boldsymbol{\beta}}]_d$.

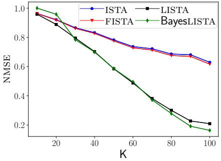
Synthetic Experiments

Different depth performance



NMSE

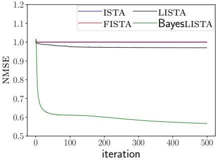
Different observation size performance



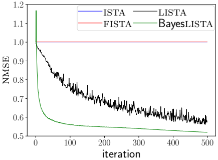
NMSE

MNIST Experiments

Results for increasing number of iterations for observation size K=100 and K=250

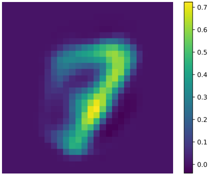


NMSE, $K = 100$

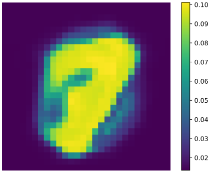


NMSE, $K = 250$

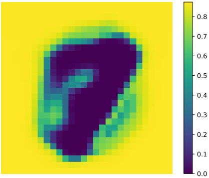
Posterior parameters for an image of digit 7



$\boldsymbol{\beta}$ posterior mean



$\boldsymbol{\beta}$ posterior std

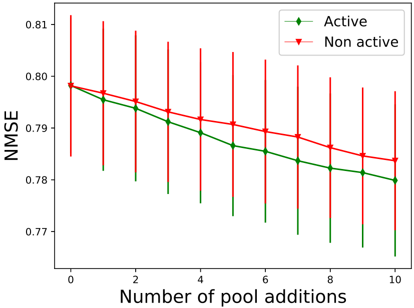


$\boldsymbol{\beta}$ posterior spike indicator

Active Learning

Idea

Use the estimated uncertainty to choose next training data with largest variance



Key contributions

- uncertainty propagation to make inference feasible
- active learning for deep sparse coding

Conclusions and future work

Future work

- Scalable stochastic inference
- Neural networks architecture