Natural Boltzmann Machines

Adrian Millea

Supervisor: Abbas Edalat
Imperial College London

am2714@imperial.ac.uk

June 15, 2016

Overview

1. Motivation - Fisher Information Matrix (FIM)
2. Background - Restricted Boltzmann Machine (RBM)
3. Natural Boltzmann Machines
4. Discussion and future work

Motivation

Problem:

- In deep nets with heavy datasets, the SGD steps are expensive (Imagenet: millions of images, each with hundreds of thousands of pixels, takes weeks to train)
- With SGD learning is far from the optimum regime

Because:

- Non-Euclidean geometry in model space (given the data)

Solution:

- Reparameterize such that the covariance is the identity map = Natural gradient [Amari 1998]

Natural gradient

Steepest descent direction in a Riemannian manifold:

$$-\widehat{\nabla} L = -G^{-1}(w)\nabla L(w)$$

and $\nabla L(w) = \left(\frac{\partial}{\partial w_1} L(w) ..., \frac{\partial}{\partial w_n} L(w)\right)^T$ w = is the parameter vector, G = associated metric, given by:

$$G = \mathbb{E}_{p(x;w)}\left[\frac{\partial \log p(x;w)}{\partial w^i}\frac{\partial \log p(x;w)}{\partial w^j}\right]$$

Can also be derived as:

$$argmin_{dw} L(w + dw)$$
$$\text{such that } KL(p_w || p_{w+dw}) = constant$$

Whitening in parameter space

Whitening transformation: X original vector (usually data vector), C = covariance matrix (non-singular), mean 0

$$Y = WX$$
$$W^T W = C^{-1}$$

the new Y which has unit diagonal covariance.
In deep nets [Desjardins et al., 2015] :

$$\mathbb{E}\left[hh^T\right] = U\Sigma U^T$$

where h is the vector of activations, U is the matrix of eigenvectors, Σ diagonal matrix of eigenvalues. The transformation is given by: $U\Sigma^{-\frac{1}{2}}$

Other approaches:

- Low rank updates of the covariance [Roux et al., 2008]
- Model the FIM with a graphical model (assuming binary hiddens and visibles) [Grosse and Salakhudinov, 2015]
- Implicitly compute matrix-vector product using a linear solver [Desjardins et al., 2013]
- Diagonal approximation

Restricted Boltzmann Machines

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-E(\mathbf{v}, \mathbf{h}))$$

$$Z = \sum_{\mathbf{v}, \mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}))$$

$E(\mathbf{v}, \mathbf{h})$ is the energy of the specific configuration given by:

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{a}\mathbf{v} - \mathbf{b}\mathbf{h} - \mathbf{v}\mathbf{W}\mathbf{h}^T$$

The probability of a visible vector is given by marginalizing over the hidden units:

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}))$$

Exact FIM

Denoting by $\mathbf{g}$ the vector of sufficient statistics, given by: $(\mathbf{v}, \mathbf{h}, vec(\mathbf{vh^T}))$, the Fisher matrix is given by its covariance
$\mathbf{G} = Cov(\mathbf{g}, \mathbf{g}) = \mathbb{E}[\mathbf{gg^T}] - \mathbb{E}[\mathbf{g}]\mathbb{E}[\mathbf{g}]^T$, with:

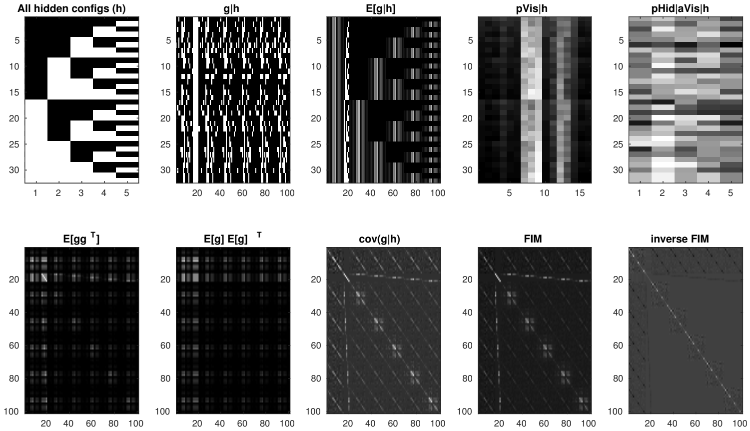
$$\mathbb{E}[\mathbf{g}] = \sum_{\mathbf{h}} p(\mathbf{h})\mathbb{E}[\mathbf{g}|\mathbf{h}]$$

$$\mathbb{E}[\mathbf{gg}^T] = \sum_{\mathbf{h}} p(\mathbf{h})(\mathbb{E}[\mathbf{g}|\mathbf{h}]\mathbb{E}[\mathbf{g}|\mathbf{h}]^T + Cov(\mathbf{g}|\mathbf{h}))$$

where the conditional expectation is over $\mathbf{v}|\mathbf{h}$, and is given by:

$$\mathbb{E}[\mathbf{g}|\mathbf{h}] = \sum_{\mathbf{v}|\mathbf{h}} p(\mathbf{v}|\mathbf{h})\mathbf{g}|\mathbf{h}$$

Exact FIM - example



Because...

- Computing the covariance is expensive (squared), SVD even more (cubic) in the number of parameters.

Solution / Main idea:

- Cheap incremental update such that we minimize covariance of activations.

Covariance penalty - no extra parameters

We incrementally decorrelate at each SGD step by adding an extra term to the gradient update.

$$\mathcal{L} = (1-\gamma)(-\log P(\mathbf{v},\mathbf{h})) + \gamma cov(\mathbf{v},\mathbf{h})^2$$

where γ is a weight coefficient which will change over time.

$$\frac{\partial cov(\mathbf{v},\mathbf{h})^2}{\partial w_{ij}} = Tr\left[(\frac{\partial cov(\mathbf{v},\mathbf{h})^2}{\partial cov(\mathbf{v},\mathbf{h})})^T \frac{\partial cov(\mathbf{v},\mathbf{h})}{\partial w_{ij}} \right]$$

We can easily compute the two terms in this equation. The first term is simply: $2cov(\mathbf{v},\mathbf{h})$. The second term is given by:

$$\frac{\partial cov(\mathbf{v},\mathbf{h})}{\partial w_{ij}} = \frac{\partial \mathbb{E}[\mathbf{v}\mathbf{h}^T]}{\partial w_{ij}} - \frac{\partial \mathbb{E}[\mathbf{v}]\mathbb{E}[\mathbf{h}]^T}{\partial w_{ij}} - \frac{\mathbb{E}[\mathbf{v}]\partial \mathbb{E}[\mathbf{h}]^T}{\partial w_{ij}} \tag{1}$$

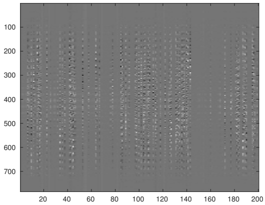
where all the expectations are with respect to the model distribution. We can readily compute the partial derivatives of the expectations with finite differences, given by:

$$\frac{\partial \mathbb{E}[\mathbf{v}\mathbf{h}^T]}{\partial w_{ij}} = \frac{\mathbb{E}[\mathbf{v}_{t+1}\mathbf{h}_{t+1}^T] - \mathbb{E}[\mathbf{v}_t\mathbf{h}_t^T]}{w_{ij}^{t+1} - w_{ij}^t}$$

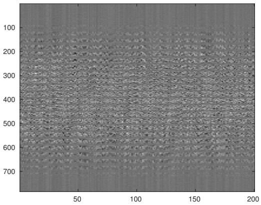
Covariance penalty - no extra parameters

Observations:

- very efficient ! (avoid matrix inversion, matrix vector product)
- generalizes better, test error always lower than train error
- sparse solutions
- for now, similar in accuracy with SGD



(a) Weight matrix trained with covariance penalty.



(b) Weight matrix trained with SGD.

Figure : Comparison between SGD and the covariance penalty technique(CP).

With extra parameters - Source separation

- Add a set of parameters B such that $B\mathbf{x} = \mathbf{y}$ with $\mathbb{E}[\mathbf{y}\mathbf{y}^T] = \mathbf{I}$

In the context of ICA, the matrix B is called the separating matrix, and the goal is to find the source signals $\mathbf{s}$ ($\mathbf{x} = A\mathbf{s}$, with A unknown) and $\mathbf{x}$ observed.

- The general rule for updating matrix B is given by:

$$B_{t+1} = B_t - \lambda_t H(\mathbf{y}_t) B_t$$

$$H(\mathbf{y}) = \mathbf{y}\mathbf{y}^T - I + f'(\mathbf{y})\mathbf{y}^T - \mathbf{y}f'(\mathbf{y})^T$$

with $f(\mathbf{y})$ (contrast function) given by:

$$\phi_4(B) = f(\mathbf{y}) = \sum_{i=1}^{n} |y_i|^4$$

Discussion and future work

- Incrementally penalizing covariance is more efficient
- A form of regularization
- We don't need to get the exact natural gradient. Getting closer to it still works better than SGD.
- We can use any method for decorrelation, contrast functions used in ICA
- Decorrelation in the brain comes as inhibitory feedback [Tetzlaff 2012]
- Hebbian-like rules

References

- Shun-Ichi Amari. Natural gradient works efficiently in learning. Neural computation, 10(2):251276, 1998.
- Guillaume Desjardins, Karen Simonyan, Razvan Pascanu, et al. Natural neural networks. In Advances in Neural Information Processing Systems, pages 20622070, 2015.
- Roger Grosse and Ruslan Salakhudinov. Scaling up natural gradient by sparsely factorizing the inverse fisher matrix. In Proceedings of the 32nd International Conference on Machine Learning (ICML-15), pages 23042313, 2015.
- Desjardins, Guillaume, et al. "Metric-free natural gradient for joint-training of Boltzmann machines." arXiv:1301.3545 (2013).
- Roux, Nicolas L., Pierre-Antoine Manzagol, and Yoshua Bengio. "Topmoumoute online natural gradient algorithm." Advances in neural information processing systems. 2008.
- Tetzlaff, Tom, et al. "Decorrelation of neural-network activity by inhibitory feedback." PLoS Comput Biol 8.8 (2012): e1002596.

Thank you