Hierarchical models

MIPT

2022

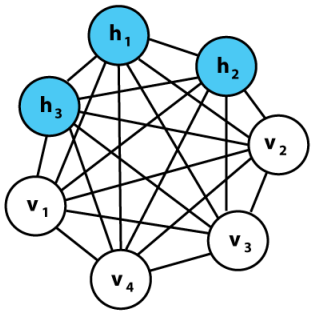
Boltzman machine

Energy-based model: $p(\mathrm{x}) = \frac{\exp(-E(\mathrm{x}))}{Z}$,

$$E(\mathrm{x}) = -\mathrm{x}^{\mathsf{T}}\mathrm{W}\mathrm{x} - \mathrm{w}_b^{\mathsf{T}}\mathrm{x}.$$

With latent variables:

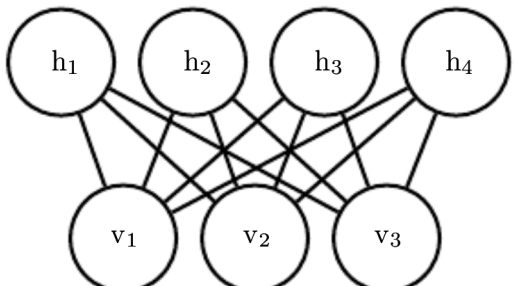
$$E(\mathrm{x}) = -\mathrm{x}^{\mathsf{T}}\mathrm{W}_v\mathrm{x} - \mathrm{x}^{\mathsf{T}}\mathrm{W}_{vh}\mathrm{h} - \mathrm{h}^{\mathsf{T}}\mathrm{W}_h\mathrm{h} - \mathrm{w}_{bh}^{\mathsf{T}}\mathrm{h} - \mathrm{w}_{bv}^{\mathsf{T}}\mathrm{x}.$$



Restricted Boltzman machine

Particular case of BM: can be represented as a bipartite graph:

$$E(\mathsf{x}) = -\mathsf{h}^{\mathsf{T}}\mathsf{W}\mathsf{h} - \mathsf{w}_{bh}^{\mathsf{T}}\mathsf{h} - \mathsf{w}_{bb}^{\mathsf{T}}\mathsf{x}.$$

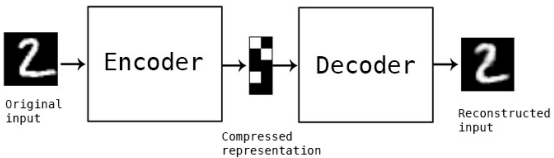


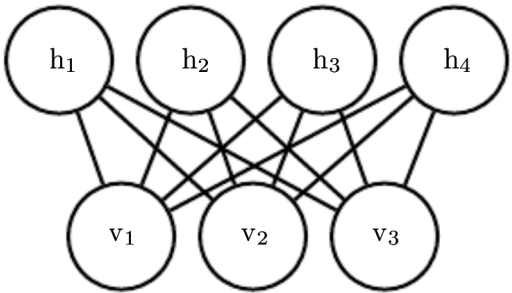
Properties of RBM

- Model is undirected
- $p(\mathrm{h}|\mathrm{x}) = \prod_{j=1}^{n_h} \sigma\left((2\mathrm{h} - 1) \cdot (\mathrm{w}_{bh} + \mathrm{W}^{\mathsf{T}}\mathrm{x})\right)_j$
- $p(\mathrm{x}|\mathrm{h}) = \prod_{j=1}^{n} \sigma\left((2\mathrm{x} - 1) \cdot (\mathrm{w}_{bv} + \mathrm{W}\mathrm{h})\right)_j$

RBM and AE

Swersky, 2010: Gaussian modification of RBM is equivalent to regularized Autoencoder.





Contrastive Divergence: idea

Energy-based model:

$$p(x|w) = \frac{\exp\left(-E_w(x)\right)}{Z(w)}, \quad Z = \int_x \exp\left(-E_w(x)\right),$$

$$\frac{\partial \log p(x|w)}{\partial w} = \mathrm{E}_{x' \sim p(x|w)} \frac{\partial E(x')}{\partial w} - \frac{\partial E(x)}{\partial w}$$

Algorithm for RBM:

- Take x from the dataset
- $h_0 \sim p(h_0|x)$
- $x_1 \sim p(x|h_0)$
- ...
- Obtain x_k
- $\frac{\partial \log p(x|w)}{\partial w} = \frac{\partial E(x_k)}{\partial w} - \frac{\partial E(x)}{\partial w}$

Discriminative model as EBM

Our key observation in this work is that one can slightly re-interpret the logits obtained from f_θ to define $p(\mathbf{x}, y)$ and $p(\mathbf{x})$ as well. Without changing f_θ, one can re-use the logits to define an energy based model of the joint distribution of data point $\mathbf{x}$ and labels y via:

$$p_\theta(\mathbf{x}, y) = \frac{\exp\left(f_\theta(\mathbf{x})[y]\right)}{Z(\theta)}\,, \tag{5}$$

where $Z(\theta)$ is the unknown normalizing constant and $E_\theta(\mathbf{x}, y) = -f_\theta(\mathbf{x})[y]$.

By marginalizing out y, we obtain an unnormalized density model for $\mathbf{x}$ as well,

$$p_\theta(\mathbf{x}) = \sum_y p_\theta(\mathbf{x}, y) = \frac{\sum_y \exp\left(f_\theta(\mathbf{x})[y]\right)}{Z(\theta)}\,. \tag{6}$$

Notice now that the $\text{LogSumExp}(\cdot)$ of the logits of *any* classifier can be re-used to define the energy function at a data point $\mathbf{x}$ as

$$E_\theta(\mathbf{x}) = -\text{LogSumExp}_y(f_\theta(\mathbf{x})[y]) = -\log \sum_y \exp(f_\theta(\mathbf{x})[y])\,. \tag{7}$$

Optimization:

$$\log p_\theta(\mathbf{x}, y) = \log p_\theta(\mathbf{x}) + \log p_\theta(y|\mathbf{x}).$$

second term: CE.

How to optimize first term?

SGLD

Gradient descent modification:

$$T = \mathrm{x} - \lambda \nabla_{\mathrm{x}} L(\mathrm{x}, \mathrm{w}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \frac{\lambda}{2})$$

where λ changes with a number of iterations:

$$\sum_{\tau=1}^{\infty} \lambda_\tau = \infty, \quad \sum_{\tau=1}^{\infty} \lambda_\tau^2 < \infty.$$

Statement [Welling, 2011]. Distribution $T \circ T \circ \ldots T$ converges to $p(\mathrm{x})$.

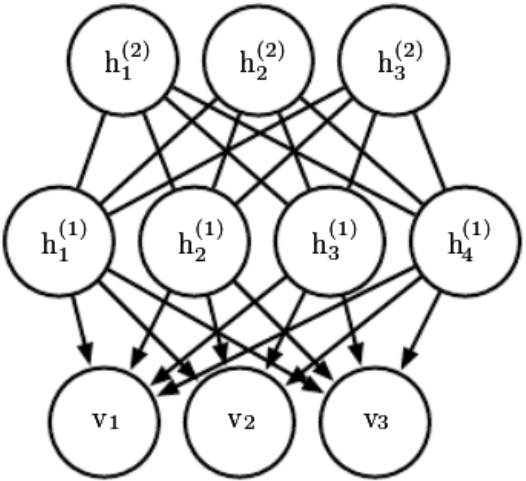
Algorithm

Algorithm 1 JEM training: Given network f_θ, SGLD step-size α, SGLD noise σ, replay buffer B, SGLD steps η, reinitialization frequency ρ

while not converged **do**
  Sample $\mathbf{x}$ and y from dataset
  $L_{\text{clf}}(\theta) = \text{xent}(f_\theta(\mathbf{x}), y)$
  Sample $\widehat{\mathbf{x}}_0 \sim B$ with probability $1 - \rho$, else $\widehat{\mathbf{x}}_0 \sim \mathcal{U}(-1, 1)$ ▷ Initialize SGLD
  for $t \in [1, 2, \ldots, \eta]$ **do** ▷ SGLD
    $\widehat{\mathbf{x}}_t = \widehat{\mathbf{x}}_{t-1} + \alpha \cdot \frac{\partial \text{LogSumExp}_{y'}(f_\theta(\widehat{\mathbf{x}}_{t-1})[y'])}{\partial \widehat{\mathbf{x}}_{t-1}} + \sigma \cdot \mathcal{N}(0, I)$
  end for
  $L_{\text{gen}}(\theta) = \text{LogSumExp}_{y'}(f(\mathbf{x})[y']) - \text{LogSumExp}_{y'}(f(\widehat{\mathbf{x}}_t)[y'])$ ▷ Surrogate for Eq 2
  $L(\theta) = L_{\text{clf}}(\theta) + L_{\text{gen}}(\theta)$
  Obtain gradients $\frac{\partial L(\theta)}{\partial \theta}$ for training
  Add $\widehat{\mathbf{x}}_t$ to B
end while

Deep belief networks

- Stack of multiple RBM
- Optimization is done layerwise
- The model is directed

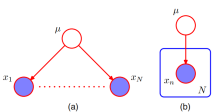


Bayesian networks

- Models are set using directed acyclic graphs
- Joint distribution for the graph with K vertices:

$$p(v_1, \ldots, v_k) = \prod_{i=1}^{K} p(v_i | \text{parent}(v_i))$$

- Example: linear regresssion



DAG and Plate notation (Bishop)

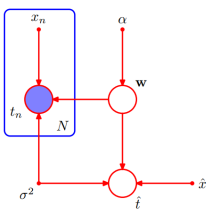
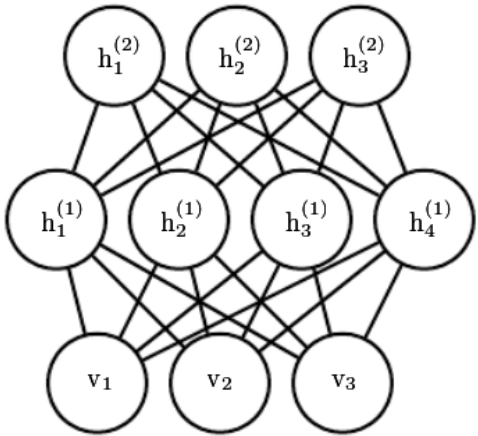


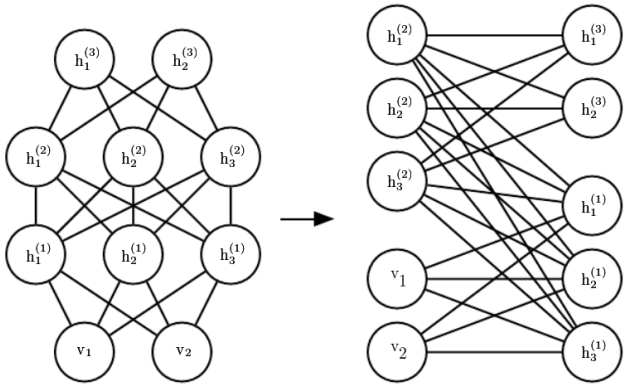
Plate notation for regression model (Bishop)

Deep Boltzman machine

- RBM with one visible and multiple hidden layers
- Model is undirected

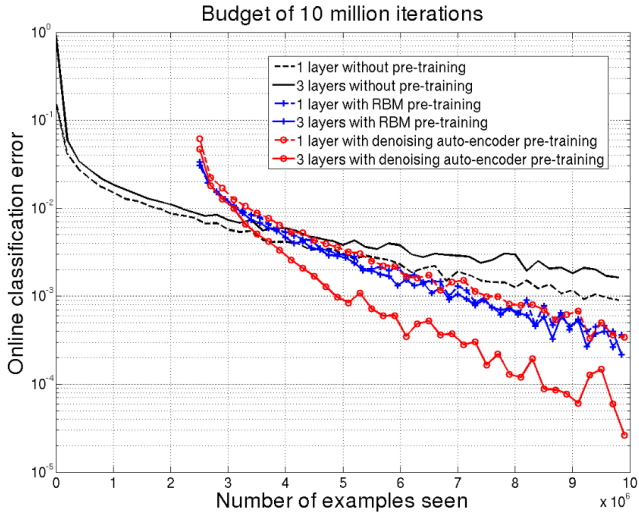


Deep Boltzman machine

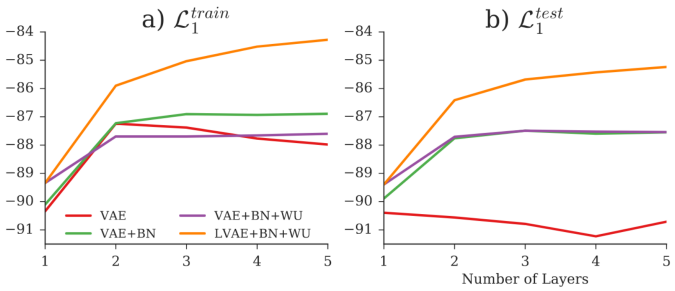


Condition on one part of the graph $\rightarrow$ get independent variables in the second part of the graph.

Greedy layerwise training



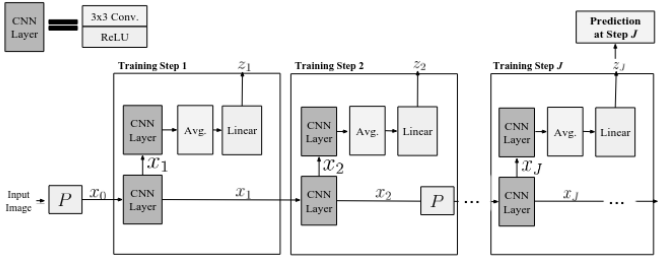
Greedy layerwise training: not always working



Layerwise training: current state

Belilovsky et al., 2019: greedy training gives more interpetability of the model and allows to train models more efficiently.
Train model in layerwise regime with L blocks. Each block is:

1. CNN + Relu layer
2. Auxilary classifier

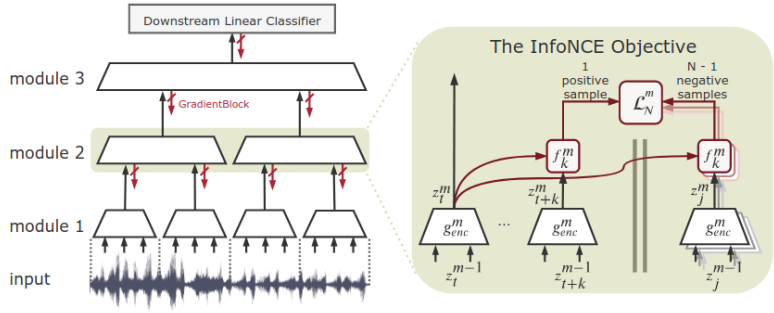


Layerwise training: current state

	Top-1 (Ens.)	Top-5 (Ens.)
SimCNN ($k = 1$ train)	58.1 (59.3)	79.7 (80.8)
SimCNN ($k = 2$ train)	65.7 (67.1)	86.3 (87.0)
SimCNN ($k = 3$ train)	69.7 (71.6)	88.7 (89.8)
VGG-11 ($k = 3$ train)	67.6 (70.1)	88.0 (89.2)
VGG-11 (e2e train)	67.9	88.0
Alternative	[Ref.]	[Ref.]
DTargetProp (Bartunov et al., 2018)	1.6 [28.6]	5.4 [51.0]
FeedbackAlign (Xiao et al., 2019)	6.6 [50.9]	16.7 [75.0]
Scat. + Linear (Oyallon et al., 2018)	17.4	N/A
Random CNN	12.9	N/A
FV + Linear (Sánchez et al., 2013)	54.3	74.3
Reference e2e CNN		
AlexNet	56.5	79.1
VGG-13	69.9	89.3
VGG-19	72.9	90.9
Resnet-152	78.3	94.1

Layerwise training: current state

Löwe et al.: greedily optimize lower bound of mutual information between layers.



Layerwise training: current state

Table 1: STL-10 classification results on the test set. The GIM model outperforms the CPC model, despite a lack of end-to-end backpropagation and without the use of a global objective. ($\pm$ standard deviation over 4 training runs.)

Method	Accuracy (%)
Deep InfoMax [Hjelm et al., 2019]	78.2
Predsim [Nøkland and Eidnes, 2019]	80.8
Randomly initialized	27.0
Supervised	71.4
Greedy Supervised	65.2
CPC	80.5 ± 3.1
Greedy InfoMax (GIM)	$\mathbf{81.9} \pm 0.3$

Table 2: GPU memory consumption during training. All models consist of the ResNet-50 architecture and only differ in their training approach. GIM allows efficient greedy training.

Method	GPU memory (GB)
Supervised	6.3
CPC	7.7
GIM - all modules	7.0
GIM - 1st module	**2.5**

Information bottleneck

General case:

$$I(\mathrm{X},\mathrm{H}) - \lambda I(\mathrm{H},\mathrm{Y}) \approx I(\mathrm{X},\mathrm{H}) + \lambda I(\mathrm{X},\mathrm{Y}|\mathrm{H})$$

Deep learning case:

$$I(\mathrm{H}_{i-1},\mathrm{H}_i) + \lambda I(\mathrm{Y},\mathrm{H}_{i-1}|\mathrm{H}_i)$$

Information bottleneck

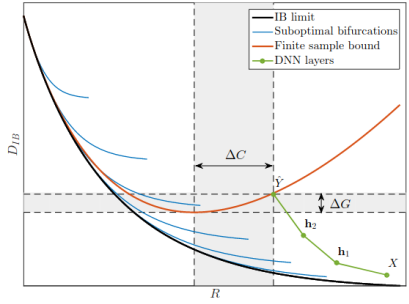


Fig. 2. A qualitative information plane, with a hypothesized path of the layers in a typical DNN (green line) on the training data. The black line is the optimal achievable IB limit, and the blue lines are sub-optimal IB bifurcations, obtained by forcing the cardinality of $\hat{X}$ or remaining in the same representation. The red line corresponds to the upper bound on the *out-of-sample* IB distortion (mutual information on Y), when training from a finite sample. While the training distortion may be very low (the green points) the actual distortion can be as high as the red bound. This is the reason why one would like to shift the green DNN layers closer to the optimal curve to obtain lower complexity and better generalization. Another interesting consequence is that getting closer to the optimal limit requires stochastic mapping between the layers.

References

- Goodfellow I., Bengio Y., Courville A. Deep learning. – MIT press, 2016.
- Carreira-Perpinan M. A., Hinton G. On contrastive divergence learning //International workshop on artificial intelligence and statistics. – PMLR, 2005. – C. 33-40.
- Restricted Boltzmann Machine, a complete analysis: https://medium.com/datatype/restricted-boltzmann-machine-a-complete-analysis-part-3-contrastive-divergence-algorithm-3d06bbebb10c
- Swersky, K. (2010). Inductive Principles for Learning Restricted Boltzmann Machines. Master's thesis, University of British Columbia
- Sønderby C. K. et al. Ladder variational autoencoders //Advances in neural information processing systems. – 2016. – T. 29.
- Grathwohl W. et al. Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One. – 2020.
- Welling M., Teh Y. W. Bayesian learning via stochastic gradient Langevin dynamics //Proceedings of the 28th international conference on machine learning (ICML-11). – 2011. – C. 681-688.
- Belilovsky E., Eickenberg M., Oyallon E. Greedy layerwise learning can scale to imagenet //International conference on machine learning. – PMLR, 2019. – C. 583-593.
- Löwe S., O'Connor P., Veeling B. Putting an end to end-to-end: Gradient-isolated learning of representations //Advances in neural information processing systems. – 2019. – T. 32.
- Alemi A. A. et al. Deep variational information bottleneck //arXiv preprint arXiv:1612.00410. – 2016.