Boltzmann machine

Probabilistic updating

$a_i \in \{0,1\}$

$\Pr[a_i = 1|\mathbf{a}_{-i}] = \text{logistic}\left(\frac{a_i^{\text{in}}}{T}\right) = \frac{1}{1+e^{-\left(\sum_j w_{ij}a_j+b_i\right)/T}}$

$\Pr[a_i|\mathbf{a}_{-i}] \propto e^{\left(\sum_j w_{ij}a_j+b_i\right)a_i/T}$

Boltzmann distribution

From statistical mechanics (thermodynamics)

$\Pr[\mathbf{a}] \propto e^{-E(\mathbf{a})/T}$

Same equation as softmax

Boltzmann machine and Gibbs sampling

Hopfield energy $E(\mathbf{a}) = -\mathbf{a}^{\mathrm{T}}W\mathbf{a} - \mathbf{b}^{\mathrm{T}}\mathbf{a}$

$\frac{\Pr[a_i=1|\mathbf{a}_{-j}]}{\Pr[a_i=0|\mathbf{a}_{-j}]} = e^{-[E(1,\mathbf{a}_{-j})-E(0,\mathbf{a}_{-j})]/T} = e^{-\left(\sum_j w_{ij}a_j+b_i\right)/T}$

Thus update rule converges to Boltzmann distribution

Logistic (and linear transformations thereof) is only function where conditional probs are mutually consistent

Stationary distribution under Hebbian learning

$E(\mathbf{a}) = -\mathbf{a}^{\mathrm{T}}W\mathbf{a} = -\frac{1}{N}\mathbf{a}^{\mathrm{T}}\left(\sum_k \mathbf{a}^k\mathbf{a}^{k\mathrm{T}}\right)\mathbf{a} = -\frac{1}{N}\sum_k\left(\mathbf{a}\cdot\mathbf{a}^k\right)^2$

(complications around {0,1} vs. {-1,1} representation)

Concentrated around attractors

Extremely slow mixing

Simulated annealing: reduce temperature over time

Restricted Boltzmann machine

Bipartite, input and hidden

Visible units v_i, hidden units h_j

Symmetric weights w_{ij}

No intralayer connections: faster convergence

Conditionally independent sampling: $p_{\text{net}}[\mathbf{h}|\mathbf{v}] \propto e^{\mathbf{v}^{\mathrm{T}}W\mathbf{h}+\mathbf{b}_{\mathrm{h}}^{\mathrm{T}}\mathbf{h}}$, $\quad p_{\text{net}}[\mathbf{v}|\mathbf{h}] \propto e^{\mathbf{v}^{\mathrm{T}}W\mathbf{h}+\mathbf{b}_{\mathrm{v}}^{\mathrm{T}}\mathbf{v}}$

Training

Input patterns $\mathbf{v} \sim p_{\text{in}}$ (training distribution)

Goal: network reproduces input distribution

Gibbs/Boltzmann distribution marginalized over $\mathbf{h}$

$p_{\text{net}}(\mathbf{v}) = \sum_{\mathbf{h}} p_{\text{net}}(\mathbf{v},\mathbf{h}) \approx p_{\text{in}}(\mathbf{v})$

Hidden layer learns latent features

Boltzmann machine learning rule

Maximum likelihood of training patterns, by gradient descent

$\prod_k p_{\text{net}}(\mathbf{v}^k) \to \sum_k \ln p_{\text{net}}(\mathbf{v}^k)$

Training set as proxy for true generating distribution: $\sum_{\mathbf{v}} p_{\text{in}}(\mathbf{v}) \ln p_{\text{net}}(\mathbf{v})$

Equivalent to maximizing $\sum_{\mathbf{v}} p_{\text{in}}(\mathbf{v})(\ln p_{\text{net}}(\mathbf{v}) - \ln p_{\text{in}}(\mathbf{v})) = \mathrm{KL}(p_{\text{in}}||p_{\text{net}})$

$$\begin{aligned}\frac{\partial}{\partial w_{ij}}\ln p_{\text{net}}(\mathbf{v}^k) &= \frac{\partial}{\partial w_{ij}}\ln\frac{\sum_{\mathbf{h}} e^{-E(\mathbf{v}^k,\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}} \\ &= \frac{1}{\sum_{\mathbf{h}} e^{-E(\mathbf{v}^k,\mathbf{h})}}\sum_{\mathbf{h}}\frac{\partial}{\partial w_{ij}}e^{-E(\mathbf{v}^k,\mathbf{h})} - \frac{1}{\sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}\sum_{\mathbf{v},\mathbf{h}}\frac{\partial}{\partial w_{ij}}e^{-E(\mathbf{v},\mathbf{h})} \\ &= \frac{\sum_{\mathbf{h}} v_i h_j e^{-E(\mathbf{v}^k,\mathbf{h})}}{\sum_{\mathbf{h}} e^{-E(\mathbf{v}^k,\mathbf{h})}} - \frac{\sum_{\mathbf{v},\mathbf{h}} v_i h_j e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}} \\ &= \mathrm{E}^+[v_i h_j] - \mathrm{E}^-[v_i h_j]\end{aligned}$$

E^+ is expectation relative to $p_{\text{net}}(\mathbf{h}|\mathbf{v}^k)$ (positive phase)

E^- is expectation relative to $p_{\text{net}}(\mathbf{v},\mathbf{h})$ (negative phase)

Contrastive divergence algorithm

Small number of steps for negative phase (e.g., 1 step) rather than convergence

- Sample $\mathbf{v}^+$ from training set
- Sample $\mathbf{h}^+$ from $p_{\text{net}}(\mathbf{h}|\mathbf{v}^+)$
- Calculate $W^+ = \mathbf{v}^{+\text{T}}\mathbf{h}^+$
- Gibbs update: Sample $\mathbf{v}^-$ from $p_{\text{net}}(\mathbf{v}|\mathbf{h}^+)$, and sample $\mathbf{h}^-$ from $p_{\text{net}}(\mathbf{h}|\mathbf{v}^-)$
- Calculate $W^- = \mathbf{v}^{-\text{T}}\mathbf{h}^-$
- Update $\Delta W = \epsilon(W^+ - W^-)$, $\Delta \mathbf{b}_{\text{v}} = \epsilon(\mathbf{v}^+ - \mathbf{v}^-)$, $\Delta \mathbf{b}_{\text{h}} = \epsilon(\mathbf{h}^+ - \mathbf{h}^-)$

Hierarchical Boltzmann machine
- Stacked RBMs
- Trained sequentially
- Hierarchy of features of increasing complexity
- Often back-propagation training for fine-tuning

<u>Cross-validation</u>

Procedure to test model fit

Measures model's ability to generalize to new data

Split data into *train* and *test* sets
- Often multiple splits
- *k*-fold CV: *k* equal sets, each used once as test set with the other *k*-1 for training

Optimize parameters on training set and evaluate fit on test set

Automatically accounts for model flexibility, e.g. free parameters
- Overfitting of training set leads to worse fit on test set
- CV performance is usually nonmonotonic (concave) function of model flexibility

Learning curve
- Test performance as function of training set size (averaged over many splits)
- Steeper for more complex models
- Simpler models win with smaller training sets, complex models win with larger training sets

Psychological parallel
- Generalizing from past experience
- Normative justification for simpler (more constrained) learning mechanisms