Energy-Based Learning Primer with a Focus on RBM's and DBN's

CM & BP

Tufts University

170411

Energy-Based Models

Definition

An energy-based model (EBM) is a model that associates a scalar energy value between all configurations of the model inputs

This association is by way of an *Energy Function*, $E(\cdot)$

Example of an Energy Function

Let $X \in \mathbb{R}^n$ be some set of observed data and $Y = \{-1, +1\}$ be the predictand of X, an example energy function could be defined as:

$$E(X, Y) = Y \sum_i X_i$$

One Important Property of $E(\cdot)$

In general $E(\cdot)$ can have any form; however, there is at least one behavior $E(\cdot)$ should have in order to be useful.

Define $E(\cdot)$ such that high compatibility between variables maps to a small values and vice-versa.

That way, inference can be viewed as finding the $y \in Y$ that minimizes the energy function, i.e.,

$$y^* = \underset{y \in Y}{\operatorname{argmin}}\, E(X, y)$$

Energy-Based Learning

Learning an EBM involves finding the best $E(\cdot)$ that exhibits this behavior.

This learning process makes use of a loss function which measures the quality of energy functions still left in the search.

From EBM's to Probabilistic Models

Up to this point, we're only interested in the minimizing values of $E(\cdot)$; however, *it's possible that the values of* $E(\cdot)$ *may need to be combined with results of, or added to, other models* (as we'll see later).

Enter Gibb's Distribution (think Softmax function):

$$Pr(Y|X) = \frac{1}{Z} e^{-\beta E(X,Y)}$$

where $\beta > 0$ and $Z = \sum_{y \in Y} e^{-\beta E(X,y)}$ is called the **Partition Function**

How About Adding Latent (hidden) Variables?

Oftentimes, adding unobserved variables to a model will increase it's expressive power. One can also measure the energy of this configuration between observed and latent information:

$$Pr(X, H) = \frac{1}{Z} e^{-\beta E(X,H)}, \quad Z = \sum_{x} \sum_{h} e^{-\beta E(x,h)}$$

In this way, unsupervised energy-based learning is also possible.

Restricted Boltzmann Machines

Definition

A Restricted Boltzmann Machine (RBM) is an EBM with the following energy function:

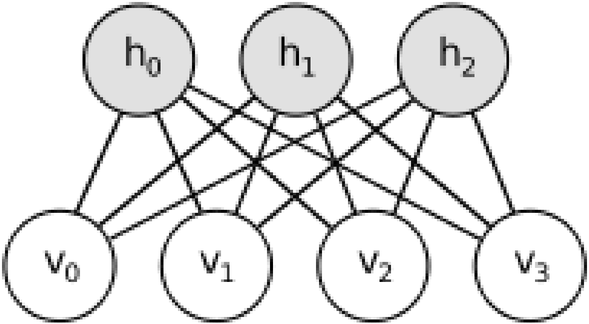
$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^\top \mathbf{v} - \mathbf{c}^\top \mathbf{h} - \mathbf{v}^\top \mathbf{W} \mathbf{h}$$

where $\mathbf{v} \in \mathbb{R}^n$, $\mathbf{h} \in \{0,1\}^m$, $\mathbf{W} \in \mathbb{R}^{nxm}$, $\mathbf{b} \in \mathbb{R}^n$, and $\mathbf{c} \in \mathbb{R}^m$.

Note

In the paper, $\mathbf{v} \in \{0,1\}^n$ (except for the first input), and in the upcoming slides, it's assumed $\mathbf{v}$ is in this sets.

Restricted Boltzmann Machines



v is typically called the "Visible Vector/Layer" and **h** is the called the "Hidden Vector/Layer"

Formulation as a Probabilistic Model

Apply Gibb's Distribution to obtain the joint density:

$$p(\mathbf{v}, \mathbf{h}) = \frac{\tilde{p}(\mathbf{v}, \mathbf{h})}{Z}$$

where

$$\tilde{p}(\mathbf{v}, \mathbf{h}) = e^{-E(\mathbf{v},\mathbf{h})} \quad \text{and} \quad Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$$

Note

Despite $\mathbf{v} \in \{0, 1\}^n$ and $\mathbf{h} \in \{0, 1\}^m$, *$Z$ is still an intractable quantity* having $O(2^{n+m})$ quantities in the sum! The marginal distributions are similarly exponential.

Properties of Binary RBM's

Visible and hidden units are conditionally independent, i.e.,

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}) \quad \text{and} \quad p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h})$$

where

$$p(h_j = 1|\mathbf{v}) = \sigma\left(\sum_i v_i W_{ij} + c_j\right), \quad p(v_i = 1|\mathbf{h}) = \sigma\left(\sum_j h_j W_{ij} + b_i\right)$$

Convenient gradients:

$$\nabla_{\mathbf{W}} E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}\mathbf{h}^\top, \quad \nabla_{\mathbf{b}} E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}, \quad \nabla_{\mathbf{c}} E(\mathbf{v}, \mathbf{h}) = -\mathbf{h}$$

What's the Goal?

When using an RBM the goal is to learn a probability distribution over the inputs, i.e., we want to learn:

$$p(\mathbf{v}) = \frac{\sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h})}{Z}$$

where

$$\tilde{p}(\mathbf{v}, \mathbf{h}) = e^{-E(\mathbf{v},\mathbf{h})} \quad \text{and} \quad Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$$

Learning with Maximum Likelihood

Let $\mathbf{\Theta} = \{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$ be the parameters of $p(\mathbf{v})$, can we use Maximum Likelihood to update these parameters?

$$p(\mathbf{v}; \Theta) = \frac{\sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}{Z}$$

$$\log p(\mathbf{v}; \Theta) = \log \sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) - \log Z$$

$$\nabla_{\Theta} \log p(\mathbf{v}; \Theta) = \nabla_{\Theta} \log \sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) - \nabla_{\Theta} \log Z$$

Yes! But we may need to calculate the intractable Z as well as the intractable sum $\sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)$

Taking a Closer Look at the Gradient

Use SGD to update the parameters, Θ, of the model:

$$\Theta \leftarrow \Theta - \epsilon \nabla_\Theta \log p(\mathbf{v}; \Theta)$$

$$\nabla_\Theta \underbrace{\log p(\mathbf{v}; \Theta)}_{\text{Objective}} = \underbrace{\nabla_\Theta \log \sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}_{\text{Positive Phase}} - \underbrace{\nabla_\Theta \log Z}_{\text{Negative Phase}}$$

Intuition

The Positive Phase seeks to increase the probability of training samples - the data that's observed. The Negative Phase seeks to decrease the probability of samples drawn from the model distribution, $p(\mathbf{v})$, which it believes in strongly.

The Positive Phase

Definition

The term $\nabla_\Theta \log \sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)$ is known as the **Positive Phase**

$$\nabla_\Theta \log \sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) = \frac{\sum_{\mathbf{h}} \nabla_\Theta \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}{\sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}$$

$$= \frac{\sum_{\mathbf{h}} \nabla_\Theta e^{\log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}}{\sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)} = \frac{\sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) \nabla_\Theta \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}{\sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}$$

$$= \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{v}) \nabla_\Theta \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) = \mathbb{E}_{\mathbf{h} \sim p(\mathbf{h}|\mathbf{v})} \left[\nabla_\Theta \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) \right]$$

The Positive Phase

Let's examine the tractability of computing:

$$\mathbb{E}_{\mathbf{h} \sim p(\mathbf{h}|\mathbf{v})} \left[\nabla_\theta \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) \right], \quad \forall \theta \in \Theta$$

Using the definition of gradient

$$\nabla f(x_1 \dots x_n) = \frac{\partial f}{\partial x_1} \mathbf{e_1} + \dots + \frac{\partial f}{\partial x_n} \mathbf{e_n}$$

and linearity of expectation, the positive phase can be rewritten

$$\sum_{\theta_i \in \theta} \mathbb{E}_{\mathbf{h} \sim p(\mathbf{h}|\mathbf{v})} \left[\frac{\partial \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}{\partial \theta_i} \right] \mathbf{e_i}$$

For $\theta_i = W_{ij}$; Positive Phase

Recall by assumption, $h_j \in \{0, 1\}$.

$$\mathbb{E}_{\mathbf{h} \sim p(\mathbf{h}|\mathbf{v})} \left[\frac{\partial \log \tilde{p}(\mathbf{v}, \mathbf{h}; \mathbf{W})}{\partial W_{ij}} \right] = \mathbb{E}_{\mathbf{h} \sim p(\mathbf{h}|\mathbf{v})} \left[v_i h_j \right] = \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{v}) v_i h_j$$

$$= v_i \sum_{h_j} \sum_{\mathbf{h_{-j}}} p(h_j, \mathbf{h_{-j}}|\mathbf{v}) h_j = v_i \sum_{h_j} p(h_j|\mathbf{v}) h_j \sum_{\mathbf{h_{-j}}} p(\mathbf{h_{-j}}|\mathbf{v})$$

$$= v_i \sum_{h_j} p(h_j|\mathbf{v}) h_j = v_i p(h_j = 1|\mathbf{v})$$

$$= v_i \sigma \left(\sum_i v_i W_{ij} + c_j \right)$$

Return to the Gradient

Use SGD to update the parameters, Θ, of the model:

$$\Theta \leftarrow \Theta - \epsilon \nabla_\Theta \log p(\mathbf{v}; \Theta)$$

$$\nabla_\Theta \underbrace{\log p(\mathbf{v}; \Theta)}_{\text{Objective}} = \underbrace{\nabla_\Theta \log \sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}_{\text{Positive Phase}} - \underbrace{\nabla_\Theta \log Z}_{\text{Negative Phase}}$$

For any $W_{ij} \in \mathbf{W}$

$$\frac{\partial \log p(\mathbf{v}; \mathbf{W})}{\partial W_{ij}} = \underbrace{v_i \sigma \left(\sum_i v_i W_{ij} + c_j \right)}_{\text{Positive Phase}} - \underbrace{\nabla_\Theta \log Z}_{\text{Negative Phase}}$$

The Negative Phase

Definition

The term $\nabla_\Theta \log Z$ is known as the **Negative Phase**

$$\nabla_\Theta \log Z = \frac{\sum_{\mathbf{v}} \sum_{\mathbf{h}} \nabla_\Theta \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}{Z} = \frac{\sum_{\mathbf{v}} \sum_{\mathbf{h}} \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) \nabla_\Theta \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta)}{Z}$$

$$= \sum_{\mathbf{v}} \sum_{\mathbf{h}} p(\mathbf{v}, \mathbf{h}) \nabla_\Theta \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) = \mathbb{E}_{\mathbf{v},\mathbf{h} \sim p(\mathbf{v},\mathbf{h})} \left[\nabla_\Theta \log \tilde{p}(\mathbf{v}, \mathbf{h}; \Theta) \right]$$

Negative Phase

Let's examine the tractability of computing:

$$\mathbb{E}_{\mathbf{v},\mathbf{h}\sim p(\mathbf{v},\mathbf{h})}\left[\nabla_\theta \log \tilde{p}(\mathbf{v},\mathbf{h};\Theta)\right], \quad \forall \theta \in \Theta$$

Using the definition of gradient

$$\nabla f(x_1 \dots x_n) = \frac{\partial f}{\partial x_1}\mathbf{e_1} + \cdots + \frac{\partial f}{\partial x_n}\mathbf{e_n}$$

and linearity of expectation, the negative phase can be rewritten

$$\sum_{\theta_i \in \theta} \mathbb{E}_{\mathbf{v},\mathbf{h}\sim p(\mathbf{v},\mathbf{h})}\left[\frac{\partial \log \tilde{p}(\mathbf{v},\mathbf{h};\Theta)}{\partial \theta_i}\right]\mathbf{e_i}$$

For $\theta_i = W_{ij}$; Negative Phase

$$\mathbb{E}_{\mathbf{v},\mathbf{h}\sim p(\mathbf{v},\mathbf{h})}\left[\frac{\partial \log \tilde{p}(\mathbf{v},\mathbf{h};\mathbf{W})}{\partial W_{ij}}\right] = \mathbb{E}_{\mathbf{v},\mathbf{h}\sim p(\mathbf{v},\mathbf{h})}\left[v_i h_j\right] = \sum_{\mathbf{v}}\sum_{\mathbf{h}} p(\mathbf{v},\mathbf{h}) v_i h_j$$

$$= \sum_{\mathbf{v}} v_i \sum_{\mathbf{h}} p(\mathbf{v}) p(\mathbf{h}|\mathbf{v}) h_j = \sum_{\mathbf{v}} v_i p(\mathbf{v}) \sum_{h_j}\sum_{\mathbf{h_{-j}}} p(h_j, \mathbf{h_{-j}}|\mathbf{v}) h_j$$

$$= \sum_{\mathbf{v}} v_i p(\mathbf{v}) \sum_{h_j} p(h_j|\mathbf{v}) h_j \sum_{\mathbf{h_{-j}}} p(\mathbf{h_{-j}}|\mathbf{v})$$

$$= \sum_{\mathbf{v}} v_i p(\mathbf{v}) \sum_{h_j} p(h_j|\mathbf{v}) h_j = \sum_{\mathbf{v}} v_i p(\mathbf{v}) p(h_j = 1|\mathbf{v})$$

Unfortunately, the sum over $\mathbf{v}$ still makes this calculation intractable!

Return to the Gradient

Use SGD to update the parameters, Θ, of the model:

$$\Theta \leftarrow \Theta - \epsilon \nabla_{\Theta} \log p(\mathbf{v};\Theta)$$

$$\nabla_{\Theta} \underbrace{\log p(\mathbf{v};\Theta)}_{\text{Objective}} = \underbrace{\nabla_{\Theta} \log \sum_{\mathbf{h}} \tilde{p}(\mathbf{v},\mathbf{h};\Theta)}_{\text{Positive Phase}} - \underbrace{\nabla_{\Theta} \log Z}_{\text{Negative Phase}}$$

For any $W_{ij} \in \mathbf{W}$

$$\frac{\partial \log p(\mathbf{v};\mathbf{W})}{\partial W_{ij}} = \underbrace{v_i \sigma\left(\sum_i v_i W_{ij} + c_j\right)}_{\text{Positive Phase}} - \underbrace{\sum_{\mathbf{v}} v_i p(\mathbf{v}) p(h_j = 1|\mathbf{v})}_{\text{Negative Phase}}$$

Dealing with the Intractable Negative Phase

Enter Gibb's Sampling and Contrastive Divergence

1. Select a mini-batch of size M, $\{\mathbf{v^{(1)}} \dots \mathbf{v^{(M)}}\}$
2. Generate $\mathbf{h^{(i)}} = \sigma(\mathbf{v^{(i)\top}} \mathbf{W} + \mathbf{c})$
3. Force elements of $\mathbf{h^{(i)}}$ to be binary by: $h_j = \mathbb{I}(h_j \geq X \sim Uniform([0, 1]))$
4. Generate $\tilde{\mathbf{v}}^{\mathbf{(i)}} = \sigma(\mathbf{W}\mathbf{h^{(i)}} + \mathbf{b})$
5. Goto 2 and repeat k times replacing $\mathbf{v^{(i)}}$ with $\tilde{\mathbf{v}}^{\mathbf{(i)}}$
6. Generate $\tilde{\mathbf{h}}^{\mathbf{(i)}} = \sigma(\mathbf{v^{(i)\top}} \mathbf{W} + \mathbf{c})$ (note that this is not binarized)

Replace:

$$\mathbb{E}_{\mathbf{v},\mathbf{h} \sim p(\mathbf{v},\mathbf{h})} [v_i h_j] \approx \frac{1}{M} \sum_{m=1}^{M} \tilde{v}_i^{(m)} \tilde{h}_j^{(m)}$$

Putting it all Together

For any $W_{ij} \in \mathbf{W}$

$$\frac{\partial \log p(\mathbf{v}; \mathbf{W})}{\partial W_{ij}} \approx \frac{1}{M}\sum_{m=1}^{M} v_i^{(m)} p(h_j^{(m)} = 1|\mathbf{v^{(m)}}) - \frac{1}{M}\sum_{m=1}^{M} \tilde{v}_i^{(m)} \tilde{h}_j^{(m)}$$

For any $b_i \in \mathbf{b}$

$$\frac{\partial \log p(\mathbf{v}; \mathbf{b})}{\partial b_i} \approx \frac{1}{M}\sum_{m=1}^{M} v_i^{(m)} - \frac{1}{M}\sum_{m=1}^{M} \tilde{v}_i^{(m)}$$

For any $c_j \in \mathbf{c}$

$$\frac{\partial \log p(\mathbf{v}; \mathbf{c})}{\partial c_j} \approx \frac{1}{M}\sum_{m=1}^{M} p(h_j^{(m)} = 1|\mathbf{v^{(m)}}) - \frac{1}{M}\sum_{m=1}^{M} \tilde{h}_j^{(m)}$$

Deep Belief Networks

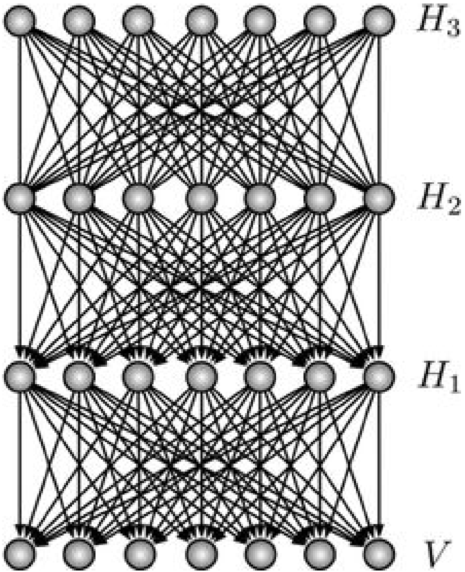
Definition

A *Deep Belief Network* (DBN) is a hybrid graphical model consisting of k RBM's where the hidden layer of the ith RBM becomes the visible layer of the $i + 1th$ RBM.

Note

It is hybrid in the sense that all edges are directed toward the starting visible layer expect for the edges in the topmost RBM.

Deep Belief Networks



Training DBN's

1. Train an RBM using the method described above, i.e., maximize $\log p^{(0)}(V)$.
2. Generate $H_1 \sim p^{(0)}(H|V)$.
3. Change the edges between layers into directed edges facing the conditioning layer
4. Create another hidden layer, H_2, and connect it to H_1 initializing new parameters. (Note the bias on the previous layer (H_1) does not get reinitialized)
5. Maximize $\log p^{(1)}(H_1)$
6. Repeat 2-5 to desired depth incrementing the layer number each time.

References

1. Carreira-Prepinan, M.A. and Hinton, G.E. (2005). On Contrastive Divergence Learning.
2. Goodfellow, I. *et al.* (2016). Deep Learning.
3. Hinton, G.E. (2010). A Practical Guide to Training Restricted Boltzmann Machines.
4. Koller, D. and Friedman N. (2009). Probabilistic Graphical Models: Principles and Techniques.
5. LeCun, Y. *et al.* (2006). A Tutorial on Energy-Based Learning.
6. http://deeplearning.net/tutorial/rbm.html
7. http://www.iro.umontreal.ca/ lisa/twiki/ bin/view.cgi/Public/DBNEquations

Extras

Why does this work?

$$\mathbb{E}_{\mathbf{v},\mathbf{h}\sim p(\mathbf{v},\mathbf{h})}\left[v_i h_j\right] = \mathbb{E}_{\mathbf{v}}\left[v_i p(h_j = 1|\mathbf{v})\right] \approx \frac{1}{M}\sum_{m=1}^{M} \tilde{v}_i^{(m)} \tilde{h}_j^{(m)}$$

Since $p(\mathbf{v}) \approx p_{train}(\mathbf{v})$ is the goal,

- When initializing the RBM with a sample from the training set it's assumed the MC is close to converging on its equilibrium distribution, $p(\mathbf{v})$
- Sampling of k iterations worth is like sampling from $p(\mathbf{v})$
- It can be shown that $\frac{1}{M}\sum_{m=1}^{M} \tilde{v}_i^{(m)} \tilde{h}_j^{(m)}$ is an unbiased estimator for $\mathbb{E}_{\mathbf{v}}\left[v_i p(h_j = 1|\mathbf{v})\right]$. (Recall $\tilde{h}_j^{(m)}$ is not binarized).

Extras

Contrastive Divergence

Minimize: $CN_k = KL(p_{train}(\mathbf{v})||p(\mathbf{v})) - KL(p_k(\mathbf{v})||p(\mathbf{v}))$

Minimized when the distribution of $\mathbf{v}$ ($p_k(\mathbf{v})$) after k Gibb's samples is the same as the distribution of the observed $\mathbf{v}$ ($p_{train}(\mathbf{v})$).

Extras; $\mathbf{v} \in \mathbb{R}^n$?

Rather than the conditional distribution over **v** be Bernoulli, let it be Gaussian as follows:

$$\mathbf{v}|\mathbf{h} \sim \mathcal{N}(\mathbf{v}; \mathbf{W}\mathbf{h}, \beta^{-1})$$

where β^{-1} is the inverse covariance matrix and the RBM's energy function is modified to incorporate this term (not shown).