What kind of a graphical model is the brain?

Vrushali Pandit
vrushali.1@iitj.ac.in

Indian Institute of Technology
Jodhpur

The big dream of Artificial Intelligence is to mathematically model the human brain. With a gross simplification, we can say that our brains are unsupervised generative models. This is because true learning is being able to self-learn and extrapolate the knowledge to incomplete or corrupted situations. We have 86 million neurons, and to learn a network with these many neurons we need thousands of billions of parameters. We also know that any model which models the brain has to be graphical because of the "neurons that wire together fire together" concept.

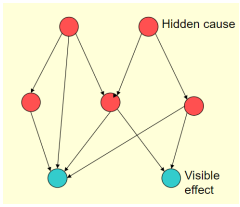
Without getting into the depth of the architecture, we first try to understand whether it is a directed graphical model or an undirected one. In previous works of the author (add references) it was assumed that the brain is either of the two. But these models simply didn't work well enough. This paper introduces a new hybrid generative model. The top two layers of this model form an undirected graph (**restricted boltzmann machine**) which can extract associative memory* from the data. This memory is used by a directed cyclic graph (**sigmoid belief network**) to convert it to observables such as pixels of an image. The paper discusses models with increasing complexity leading up to this hybrid model.

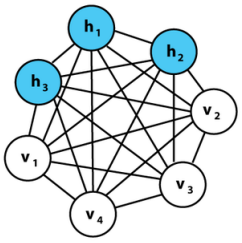
Each neuron of our brain can be modeled as a binary stochastic neuron which can either be "on" or "off" depending on the bias terms and its neighbouring neurons or nodes.

$$p(s_i = 1) = \frac{1}{1 + exp(-b_i + \sum_j s_j * w_{ij})}$$

Where s_i is the ith node, b_i its bias term, j iterates over all the nodes connected to it with w_{ij} weights.

When such neurons are arranged in a directed graph it becomes a **SIGMOID BELIEF NETWORK**. Also called bayesian networks, SBN are nets where each node is a random variable and edges denote the conditional probability of the child node given that the parent nodes take some value. Hence each edge is a factor of the joint probability of the entire network. Although the SBN has advantages in terms of interpretability, there wasn't a single efficient algorithm to solve it. Additionaly, being a directed model it had the issue "explaining away" which induces dependencies in indepent nodes.





When the stochastic binary units are placed in an undirected manner we get a **BOLTZMANN MACHINE** . It is made of hidden units and visible units which are all interconnected. The machine doesn't produce an output, instead it learns the probability distribution of all the units of the network, and while testing predicts beforehand whether it'll go in a similar state or not. It is an energy based model meaning that there is an overall energy associated with every configuration of the model. In every step of weight update, the model attains a configuration which lowers this overall energy.

$$E(v,h) = \sum_{i=vis} v_i * b_i + \sum_{k=hid} h_k * b_k + \sum_{i<j} v_i * v_j * w_{ij} + \sum_{k<l} h_k * h_l * w_{kl}$$

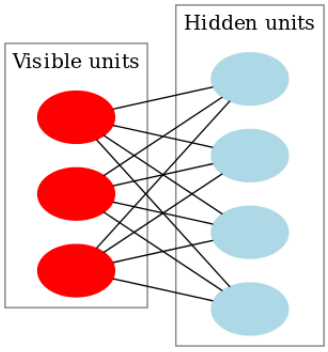
This equation captures visible bias, hidden bias, visible-visible, visible-hidden and hidden-hidden interactions (in the same order).

The probability of the model attaining a certain configuration, and because for a particular kind of data, the number of visible nodes are fixed, we have:

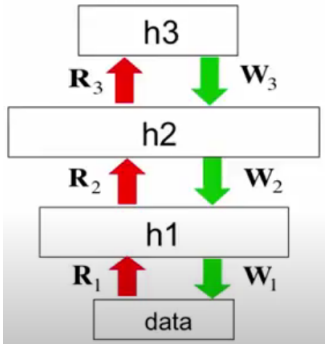
$$P^{v,h} = \frac{\sum_{h'} exp(-E^{vh'})}{\sum_{v'h'} exp(-E^{v'h'})}$$

which sums up over a fixed visible configuration. The learning algorithm has to maximize these probabilities and hence drive the energy function to equilibrium. In the learning phase, hidden units are chosen randomly and updated using eq 1 in such a way that eqn 3 increases every time. However, it takes forever for this to reach thermal equilibrium because it has to iterate over all configurations of hidden units in every single weight update.

To improve this, we tweak the model by having only one hidden layer, keeping all hidden units independent and all visible units independent. This gives us a **RESTRICTED BOLTZMANN MACHINE**. There are two sets of weights- hidden and visible weights. In the first forward pass, the hidden biases are used to train the visible biases to create some output activations. These activations are used to reconstruct the input, where now the visible biases are used to train the hidden biases. We calculate the error as the dissimilarity between the input and the reconstructed input. Because of these independencies, we can update all the hidden units at one go and won't have to randomly choose them to update. RBM's use Kullback-Leibler divergences as the cost and contrastive divergence for learning.



"Restricted Boltzmann Machine"



"Wake sleep Algorithm"

Till now, the problem of learning an undirected graphical model of the brain has been slightly solved. But there still isn't even a somewhat efficient algorithm for solving a SBG. To do so, the **WAKE SLEEP ALGORITHM** is introduced. Similar to the learning in RBMs, there are two sets of weights; generative weights (W1,W2,W3) and recognition weights (R1,R2,R3). In the WAKE phase (the forward pass) we use to recognition weights to get activations for all the hidden units. The activation of each hidden layer belongs to a particular approximate distribution of that layer. Now perform maximum likelihood learning to fit the approximate distribution to the original distribution. These best parameters are the generative weights. Now you sample vectors for hidden units from these approximate distributions and treat them as the activations. Then in the SLEEP phase do the exact opposite and learn the recognition weights.