Simulating Recurrent Neural Networks on Graphic Processing Units

Summer Project

Zhangsheng Lai

September 29, 2017

Introduction

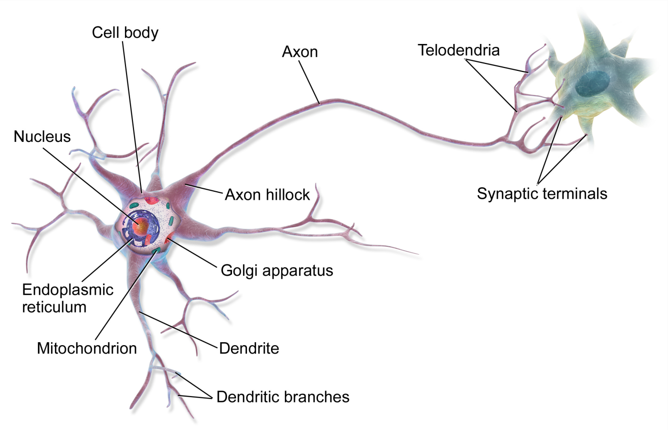


Figure 1: Anatomy of a neuron[1]

[1] By BruceBlaus - Own work, CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=28761830

Feedforward Neural Network

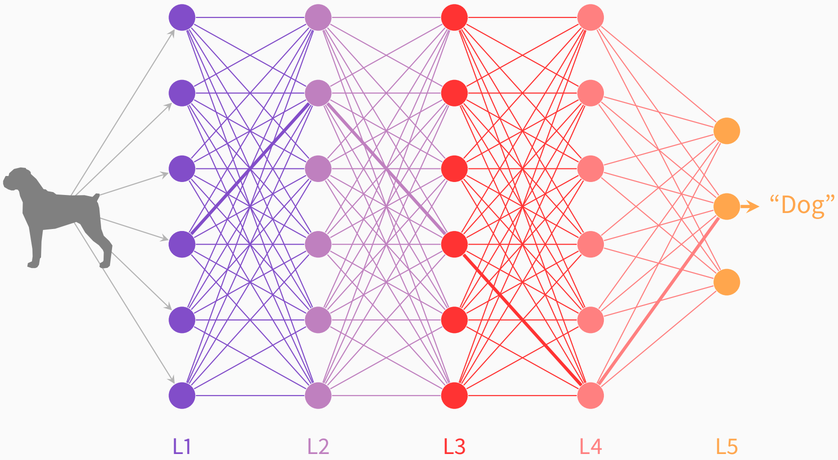
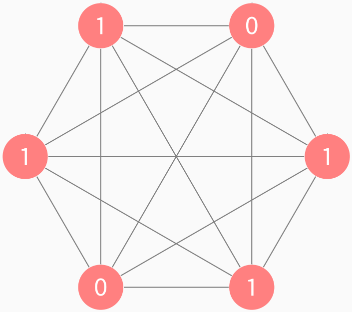


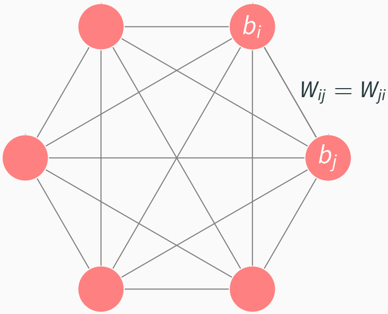
Figure 2: Feedforward Neural Network

Recurrent Neural Networks

Boltzmann Machines



Boltzmann Machines



$$\text{Energy configuration, } E = -\sum_{i<j} W_{ij} x_i x_j - \sum_i b_i x_i$$

$$\text{Energy gap, } \Delta E_i = E(x_i = 0) - E(x_i = 1) = \sum_j W_{ij} x_j + b_i$$

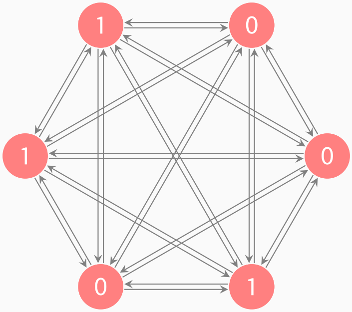
$$p_i := \mathbb{P}(x_i = 1) = \frac{1}{1 + e^{-\Delta E_i / \tau}}$$

Boltzmann Machines

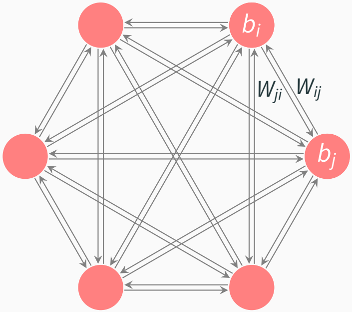
Algorithm 1 Boltzmann Machine Simulation.

Initialize $\boldsymbol{W}, \boldsymbol{b}$
Initialize $\boldsymbol{x}^{(0)}$
for i from 1 to N **do**
    Random $k \in \{1, \ldots, d\}$, where d is the number of neurons
    Compute $p_k = \frac{1}{1+e^{-\Delta E_k / \tau}}$
    $\boldsymbol{x}^{(i)} \leftarrow \text{flip}(\boldsymbol{x}^{(i-1)})$
end for

where flip($x^{(i-1)}$) flips the state of the chosen neuron k.


1
0
1
0
0
1

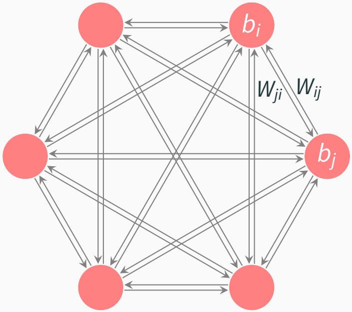
McCulloch-Pitts Machines



$$\text{Transition Energy, } E(y, x|\theta) = -\sum_{ji \in E} W_{ji} y_j x_i - \sum_{j \in V} b_j s_j - \sum_{i \in V} b_i s_i$$

$$\Gamma_{yx} = \exp\left(-\frac{1}{2\tau} E(y, x|\theta) + \frac{1}{2\tau} E(x, x|\theta)\right)$$

McCulloch-Pitts Machines

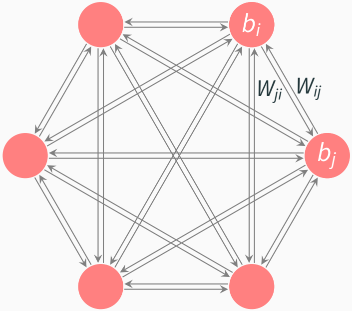


$$\text{Transition Energy, } E(y, x|\theta) = -\sum_{ji \in E} W_{ji} y_j x_i - \sum_{j \in V} b_j s_j - \sum_{i \in V} b_i s_i$$

$$\Gamma_{yx} := \exp\left(\frac{1}{2\tau} s_j z_j\right)$$

where $s_j = 1 - 2x_j$, $z_j = \sum_j W_{ji} x_i + b_j$ and x, y differ by the jth unit.

McCulloch-Pitts Machines



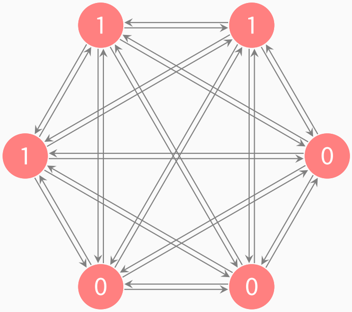
Transition probability from x to y, $p_{yx} = \dfrac{\lambda_j}{\sum_{j'} \lambda_{j'}}$

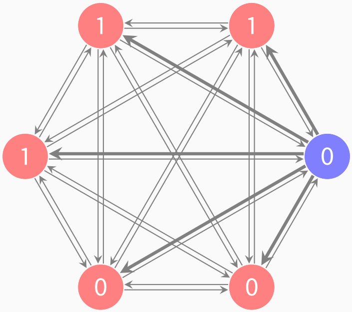
McCulloch-Pitts Machines

Algorithm 2 CTMC Simulation.

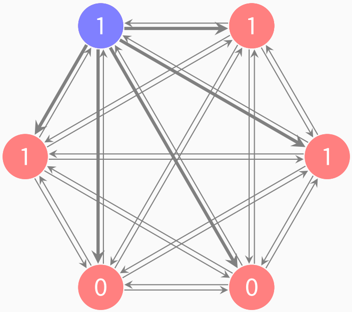
```
Initialize W, b
Initialize x^(0)
for i from 1 to N do
    Compute Γ_yx, p_yx for each y
    Compute a_x = ∑ Γ_yx
    x^(i) ← flip(x^(i−1))
    Sample holding time T_{i−1} ∼ Exp(a_x)
end for
```

where $\mathrm{flip}(x^{(i-1)})$ flips the state of the transiting neuron.


1
1
1
0
0
0

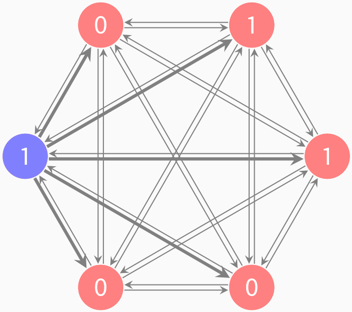


$$(T_0, (1, 0, 0, 0, 1, 1))$$



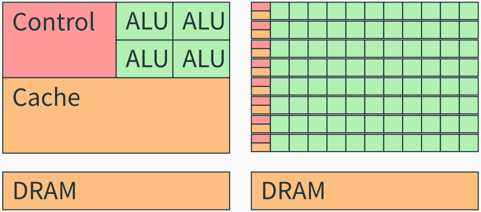
$$(T_0, (1, 0, 0, 0, 1, 1))$$

$$(T_1, (1, 0, 0, 1, 1, 1))$$



$$(T_0, (1, 0, 0, 0, 1, 1))$$
$$(T_1, (1, 0, 0, 1, 1, 1))$$
$$(T_2, (1, 0, 0, 1, 1, 0))$$

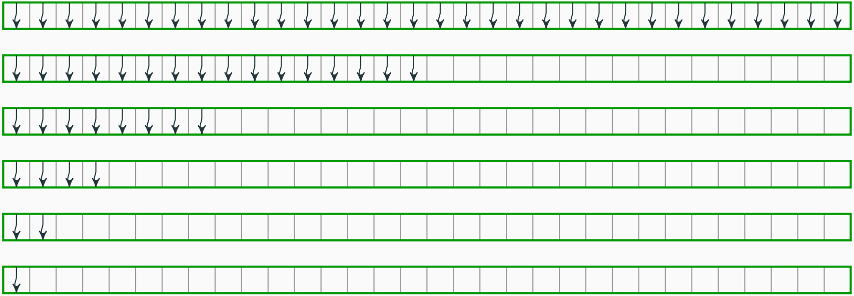
Simulating on GPUs



(a) CPU (b) GPU

Figure 3: Comparison between the amount of transistors devoted to different functions inside a CPU and a GPU.

GPU Algorithm: Reduction



```
mod = SourceModule("""
        __global__ void reduce_kernel(float *d_out, float *d_in)
        {
            int myId = threadIdx.x + blockDim.x * blockIdx.x;
            int tid = threadIdx.x;
            // do reduction in global memory
            for (unsigned int s = blockDim.x / 2; s > 0; s >>= 1)
            {
                if (tid < s)
                {
                    d_in[myId] += d_in[myId + s];
                }
                __syncthreads(); // make sure all adds at one stage are
                     done
            }
            // only thread 0 writes result for this block back to global
                 memory
            if (tid == 0)
            {
                d_out[blockIdx.x] = d_in[myId];
            }
        }
        """, arch='sm_60')
```

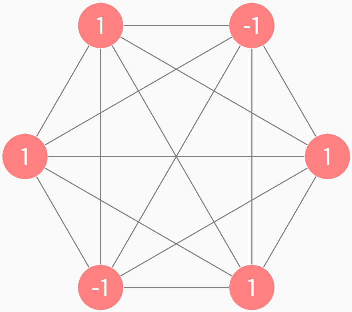
Importance to Simulating on GPUs

- Faster matrix multiplication
- Larger neural networks
- Larger function space
- Energy efficiency

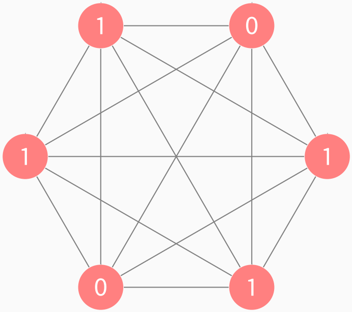
References

- Biological Plausible Deep Learning for Recurrent Spiking Neural Networks S. Lin.
- CSC321: Introduction to Neural Networks and machine Learning ***Hopfield nets and simulated annealing*.**
  https://www.cs.toronto.edu/ hinton/csc321/notes/lec16.pdf
- CSC321: Introduction to Neural Networks and machine Learning ***Boltzmann Machines as Probabilistic Models*.**
  https://www.cs.toronto.edu/ hinton/csc321/notes/lec17.pdf

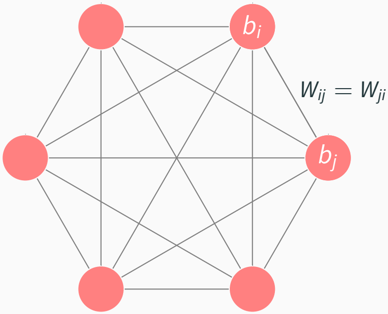
Hopfield Networks



Hopfield Networks



Hopfield Networks

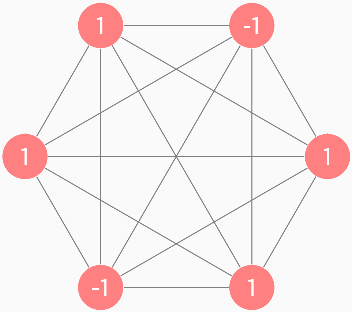


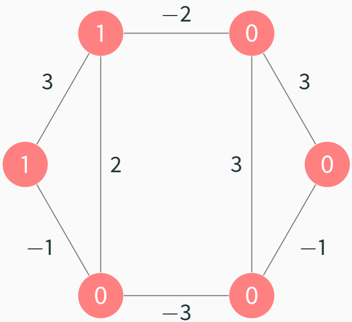
Energy configuration, $E = -\sum_{i<j} W_{ij} x_i x_j - \sum_i b_i x_i$

Energy gap, $\Delta E_i = E(x_i = 0) - E(x_i = 1) = \sum_j W_{ij} x_j + b_i$

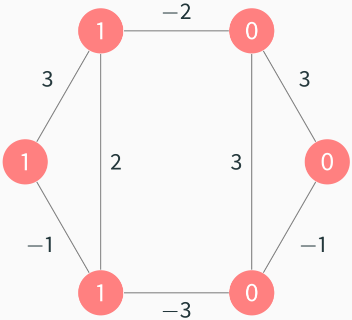
Update rule, $x_i := \begin{cases} 1 & \sum_j W_{ij} x_j + b_i \geq 0 \\ 0 & \text{otherwise} \end{cases}$

Hopfield Networks





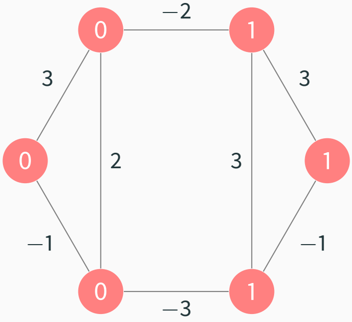
$(1, 0, 0, 0, 0, 1)$



$(1, 0, 0, 0, 0, 1)$

$(1, 1, 0, 0, 0, 1)$

Hopfield Networks



$$(1, 0, 0, 0, 0, 1)$$

$$(1, 1, 0, 0, 0, 1)$$

$$(0, 0, 1, 1, 1, 0)$$