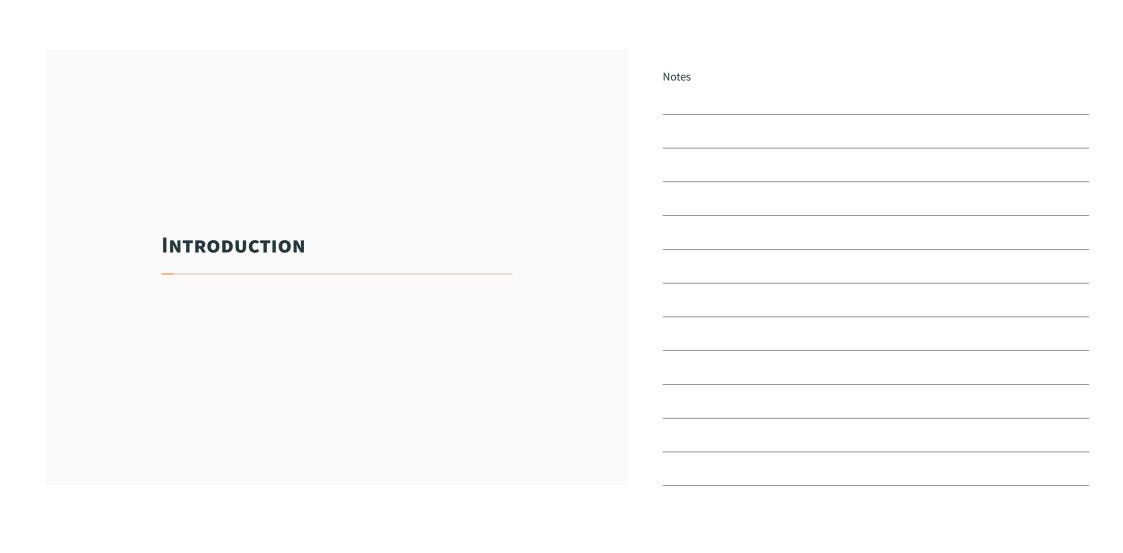
SIMULATING RECURRENT NEURAL NETWORKS ON GRAPHIC PROCESSING UNITS

SUMMER PROJECT

Zhangsheng Lai September 28, 2017

Notes			

1



INTRODUCTION

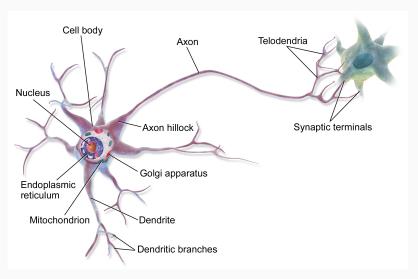


Figure 1: Anatomy of a neuron¹

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

FEEDFORWARD NEURAL NETWORK "Dog" L1 L2 L3 L4 L5 Figure 2: Feedforward Neural Network 3

Notes			

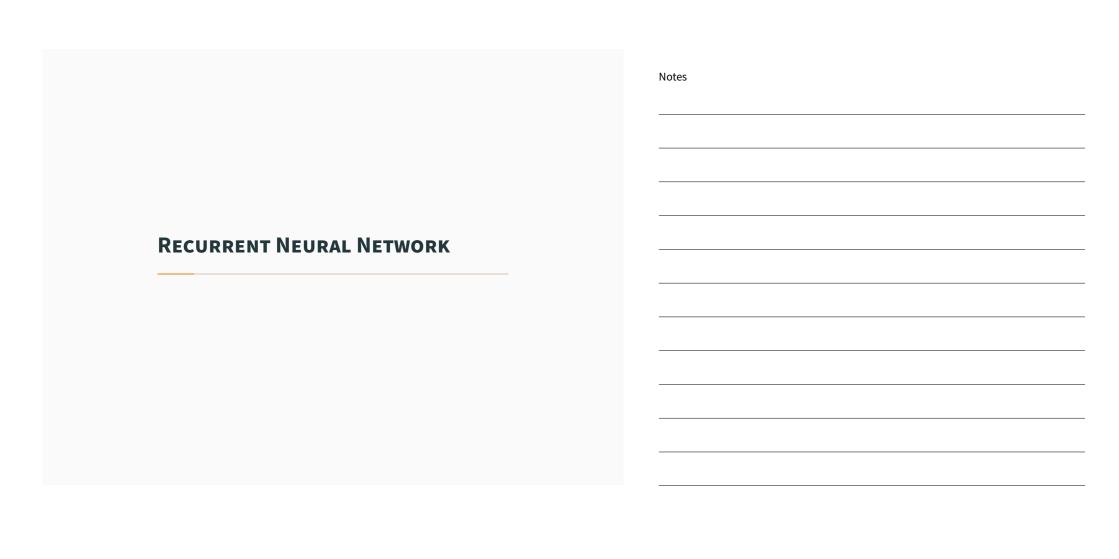
Simulating Recurrent Neural Networks on Graphic Processing Units
Introduction
Feedfo

Motos

Feedforward Neural Network

- the features that define a neuron are electrical excitability, the presence of synapses, which are complex membrane junctions that transmit signals to other neurons
- there are approximately 10¹⁴ neurons in the human brain
- artifical neuron networks are inspired by these biological neurons
- for example we have feedforward neural networks where connections between the units do not form a cycle
- the connections between the neurons in our brain are much more complex than those in the feedforward neural networks
- we have managed to use feedforward neural networks, to classify images very well
- however, the methodology used to do classification is based on learning parameters of the model and then do matrix multiplication to obtain a probability of it being classified as a particular class.
- supervised: MNIST digits classification, unsupervised: autoencoders
- these neural networks are the more popular and mainstream ones, but today we are going to look at RNNs and how to simulate them.
- we will look at some RNNs where their architecture is closer to our brains and by building such neural networks with the neurons matching the number of neurons in the brain, we hope to possibly arrive at some learning theories that is close to learning in the brain, if not as good as the brain
- to construct such a big network of neurons, we have to rely on hardware that are more suitable to dealing with large numbers of computation, thus we would want to simulate these RNNs on GPUs

Notes			
-			



Simulating Recurrent Neural Networks on Graphic Processing Units Recurrent Neural Network Recurrent Neural Network

RECURRENT NEURAL NETWORK

- Long short term memory (LSTM) is a RNN architecture that remembers values over arbitrary intervals
- LSTMs able to connect previous information to the present task, such as using previous video frames might inform the understanding of the present frame

votes		

Notes			

Simulating Recurrent Neural Networks on Graphic Processing Units
Recurrent Neu
Roltzm

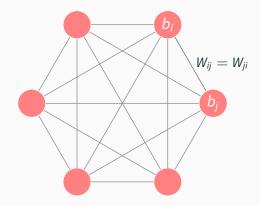
Recurrent Neural Network

Boltzmann Machines

- composed of primitive computing elements called units
- units has two states, on or off, represented by {1, 0}
- connected to each other by bi-directional links
- adopts these states as a function of the states of its neighbouring units and weights of its links to them, it is a probabilistic function for a Boltzmann machine.
- weights can take on any real value
- a unit being on or off is taken to mean that the system currently accepts or rejects some elemental hypothesis of the domain
- weight on a link represents a weak pairwise constrain between two hypothesis
- positive (negative) weights indicate that two hypothesis support (contradict) one another with other things being equal
- link weights are symmetric, having the same strength in both directions

Notes		

BOLTZMANN MACHINES



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

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

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

5

Simulating Recurrent Neural Networks on Graphic Processing Units
Recurrent Net Recurrent Neural Network

Notes

Boltzmann Machines

- temperature variable controls the amount of noise; higher temperature means more noise and also gives us a higher probability of transiting to a higher energy state and hence avoids local minimum
- ullet when au o 0 we get back the Hopfield network
- for $\tau_1 > \tau_2$, we are less likely to go to a lower energy state compared to in τ_1 compared to τ_2 , i.e. more likely to go to a higher energy state when the temperature is higher. This allows us to escape from local minimum and arrive at the global minimum
- we simulate the BM in the similar way to how it was in the Hopfield networks, but replace the binary threshold units with binary stochastic units

-		

BOLTZMANN MACHINES

Algorithm 1 Boltzmann Machine Simulation.

1: Initialize W, b

2: Initialize $\mathbf{x}^{(0)}$

3: **for** *i* from 1 to *N* **do**

Random $k \in \{1, \dots, d\}$, where d is the number of neurons

5: Compute $p_k = \frac{1}{1+e^{-\Delta E_k/\tau}}$ 6: $\boldsymbol{x}^{(i)} \leftarrow \text{flip}(\boldsymbol{x}^{(i-1)})$

7: end for

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

Notes			

BOLTZMANN MACHINES

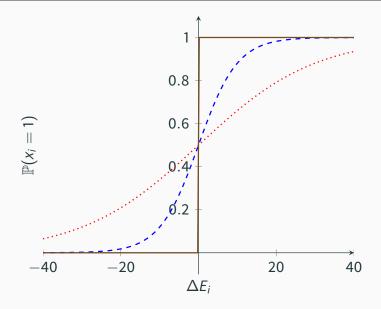
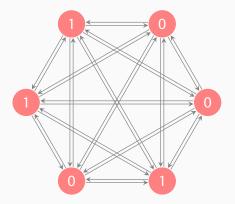


Figure 3: au=0 (solid), au=5 (dashed), au=15 (dotted)

Notes			





Notes			

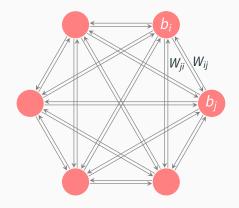
8

Simulating Recurrent Neural Networks on Graphic Processing Units

McCulloch-Pit -McCulloch-Pitts Machines

- state 1 is the refractory state, the neuron just fired and is unable to fire till it recovers
- state 0 is the armed state, the neuron just recovered and is waiting to fire
- here we model the units with the Nossenson-Messer neuron model, which explains biological firing rates in response to external stimuli

Notes			



Transition Energy,
$$E(y, x|\theta) = -\sum_{ji \in E} W_{ji}y_jx_i - \sum_{j \in V} b_js_j - \sum_{i \in V} b_is_i$$

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

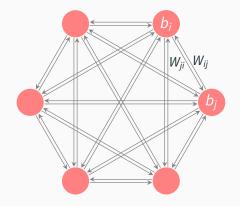
Notes			

Simulating Recurrent Neural Networks on Graphic Processing Units

McCulloch-Pit McCulloch-Pitts Machines

- digraph, G=(V,E), weights $W:V\to\mathbb{R}$, biases $b:E\to\mathbb{R}$, binary states $\mathbb{B}=\{0,1\}$ with an initial distribution $\mathbb{B}^{|V|}\to\delta$ and a temperature τ
- allow transitions where y and x differ by only one bit
- here the W matrix need not be symmetrical with zero diagonals like what we had in the Hopfield network and Boltzmann machine models
- for each $y \neq x$, start a Poisson process with rate Γ_{yx}
- as such, we can talk about the interarrival timings of the Poisson process and our simulation of the McCulloch-Pitts machine not only gives us a binary tuple, but also the time taken from it to transit from its earlier state

Notes			



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

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

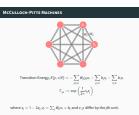
where $s_j = 1 - 2x_j$, $z_j = \sum_j W_{ji}x_i + b_j$ and x, y differ by the jth unit.

Notes

Simulating Recurrent Neural Networks on Graphic Processing Units

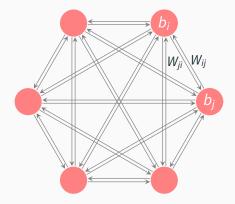
McCulloch-Pit

McCulloch-Pitts Machines



- when doing the updates we can just update the linear responses z_i and apply softmax on the λ_i 's to get the probability distribution of the transitions.
- it seems counter-intuitive to think of 0 as armed and 1 as refractory, but it is in fact the most natural thinking
- ullet a transition from 0 o 1 is a act of firing and a transition from 1 o 0 is the act of recovery
- when a neuron transit from $0 \rightarrow 1$, it changes the value of the linear response; for a transiting neuron i, if $W_{ii} > 0$, then such a transition increases the linear response of neuron j and if $W_{ii} < 0$ it decreases the linear response of neuron *j*
- the sign s depends on the state of the neuron, it preserves the sign of the linear response if it is armed and flips the sign of the linear response if it is refractory

Notes			



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

Notes			

Simulating Recurrent Neural Networks on Graphic Processing Units

McCulloch-Pit

McCulloch-Pitts Machines

McCulloch-Pitts Machines

• when doing the updates we can just update the linear responses z_j and apply softmax on the λ_j 's to get the probability distribution of the transitions.

Notes			

Simulating Recurrent Neural Networks on Graphic Processing Units McCulloch-Pit McCulloch-Pitts Machines

☐ McCulloch-Pitts Machines

• McCulloch-Pitts machine is continuous time; we have a

Notes			

Algorithm 2 CTMC Simulation.

1: Initialize **W**, **b**

2: Initialize $\mathbf{x}^{(0)}$

3: **for** *i* from 1 to *N* **do**

Compute Γ_{yx} , P_{yx} for each \mathbf{y}

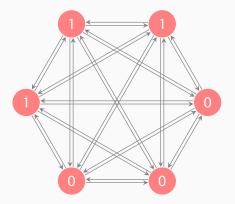
5: Compute $a_x = \sum_{i} \Gamma_{yx}$ 6: $\mathbf{x}^{(i)} \leftarrow \text{flip}(\mathbf{x}^{(i-1)})$

Sample holding time $T_{i-1} \sim \mathsf{Exp}(a_x)$

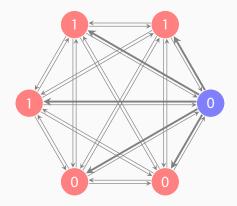
8: end for

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

Notes			

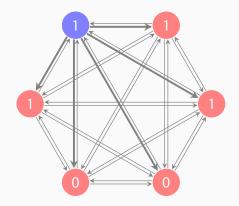


Notes		



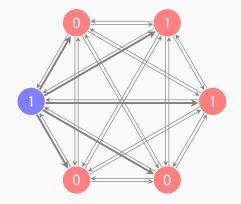
 $(T_0, (1, 0, 0, 0, 1, 1))$

14



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

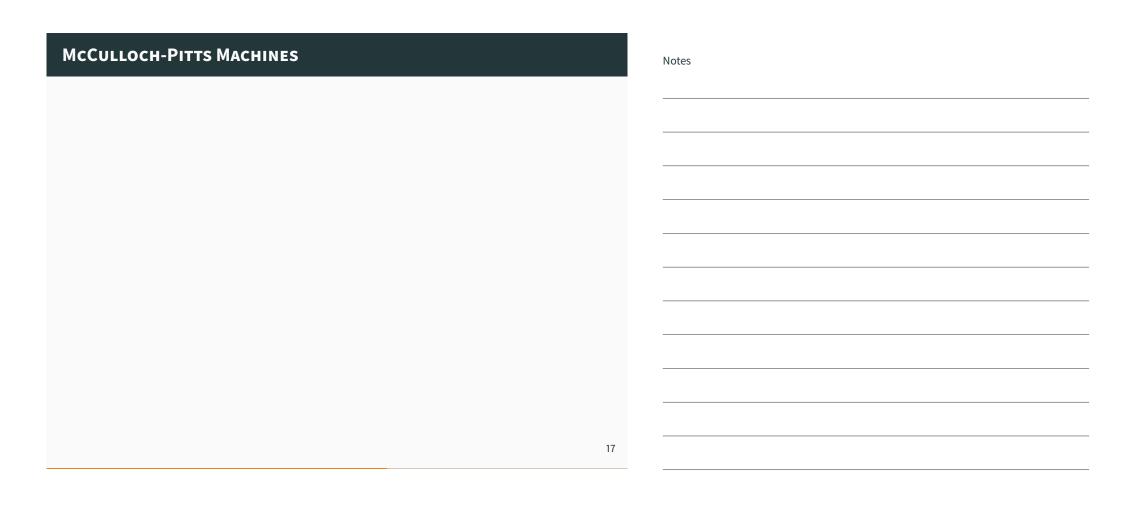
Notes		

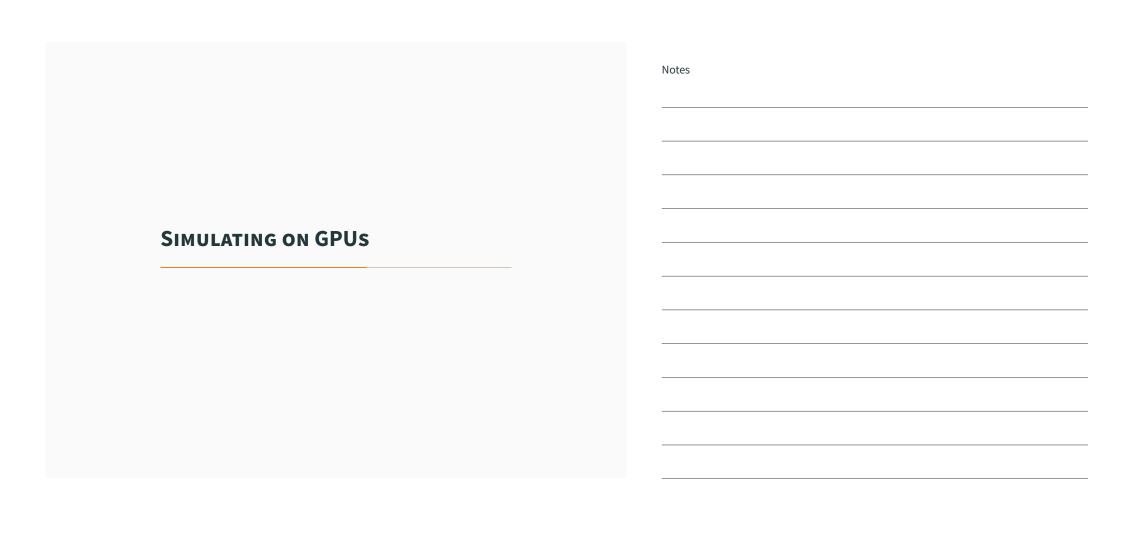


 $(T_0, (1, 0, 0, 0, 1, 1))$ $(T_1, (1, 0, 0, 1, 1, 1))$

 $(T_2,(1,0,0,1,1,0))$

Notes			





Control ALU ALU ALU Cache DRAM (a) CPU (b) GPU Figure 4: Comparison between the amount of transistors devoted to different functions inside a CPU and a GPU.

Notes

Simulating Recurrent Neural Networks on Graphic Processing Units
Simulating on

Simulating on GPUs

☐ Simulating on GPUs



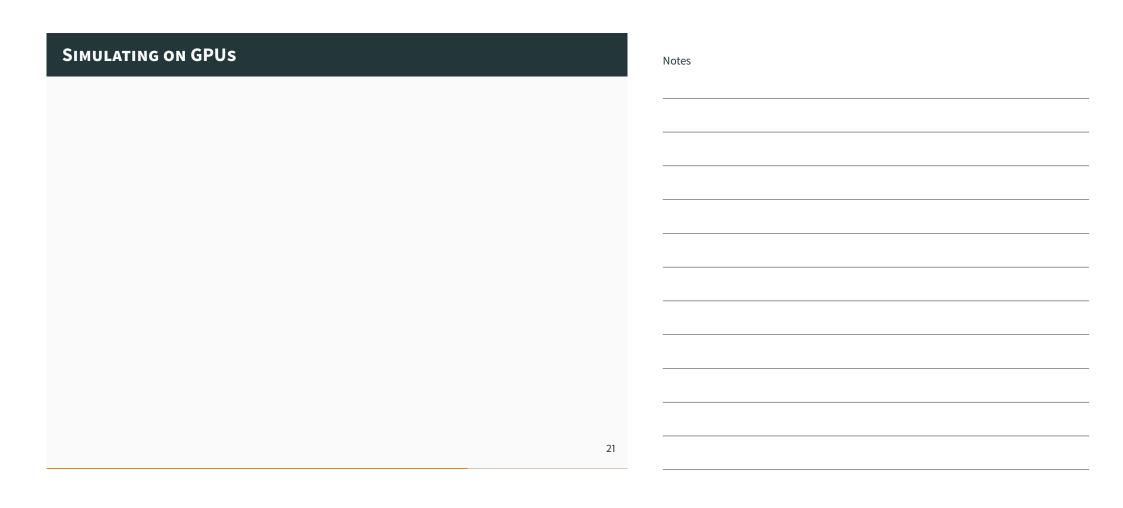
- To simplify quite a bit, think of a GPU as a factory and a CPU as Steven Hawking. Factory workers, each represented by a core, can complete lots of easy, similar tasks with incredible efficiency?tasks like geometry and shading. On the other hand Mr. Hawking, while incredibly smart and only occasionally baffled, is just one man. His skill set is better used on singular, complex problems like artificial intelligence.
- DRAM: dynamic random access memory, ALU: arithmetic logic unit, Cache, Control
- trade off control for compute in the form of lots of simple compute units
- GPUs have an explicit programming model; we have to write programs in the way that we utilise as much of the parallel processing as much as possible
- GPUs optimize for throughput, not latency; they are willing to accept increase latency of any single individual computation in exchange for more computation being performed per second, the computation performed per second is measured by floating point operations per second (FLOPS)

Notes			
-			



SIMULATING ON GPUS block 0 block 1 block 2 block 3 20

Notes		



REFERENCES



Biological Plausible Deep Learning for Recurrent Spiking Neural Networks

CSC321: Introduction to Neural Networks and machine Learning

Hopfield nets and simulated annealing.

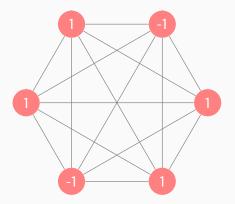
https://www.cs.toronto.edu/ hinton/csc321/notes/lec16.pdf

CSC321: Introduction to Neural Networks and machine Learning

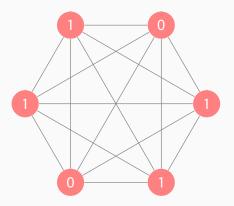
Boltzmann Machines as Probabilistic Models.

https://www.cs.toronto.edu/ hinton/csc321/notes/lec17.pdf

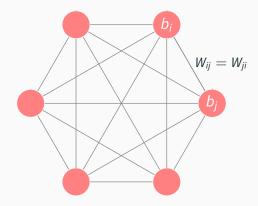
Notes			



Notes			



Notes			

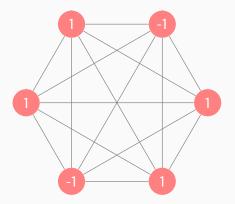


Energy configuration,
$$E = -\sum_{i < j} W_{ij} x_i x_j - \sum_i b_i x_i$$

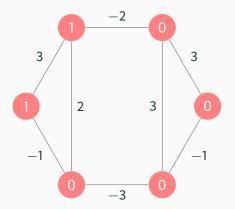
Energy gap,
$$\Delta E_i = E(x_i = 0) - E(x_i = 1) = \sum_j W_{ij}x_j + b_i$$

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

Notes

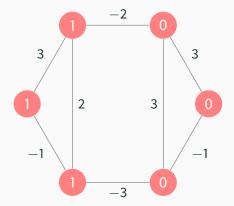


Notes			



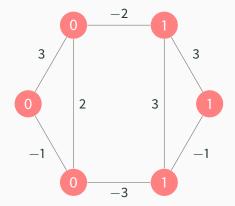
(1,0,0,0,0,1)

Notes			



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

Notes		



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

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

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

Notes			