CIO NN

CIO NN stands for **C**ontroller **I**nput **O**utput **N**erual **N**etwork

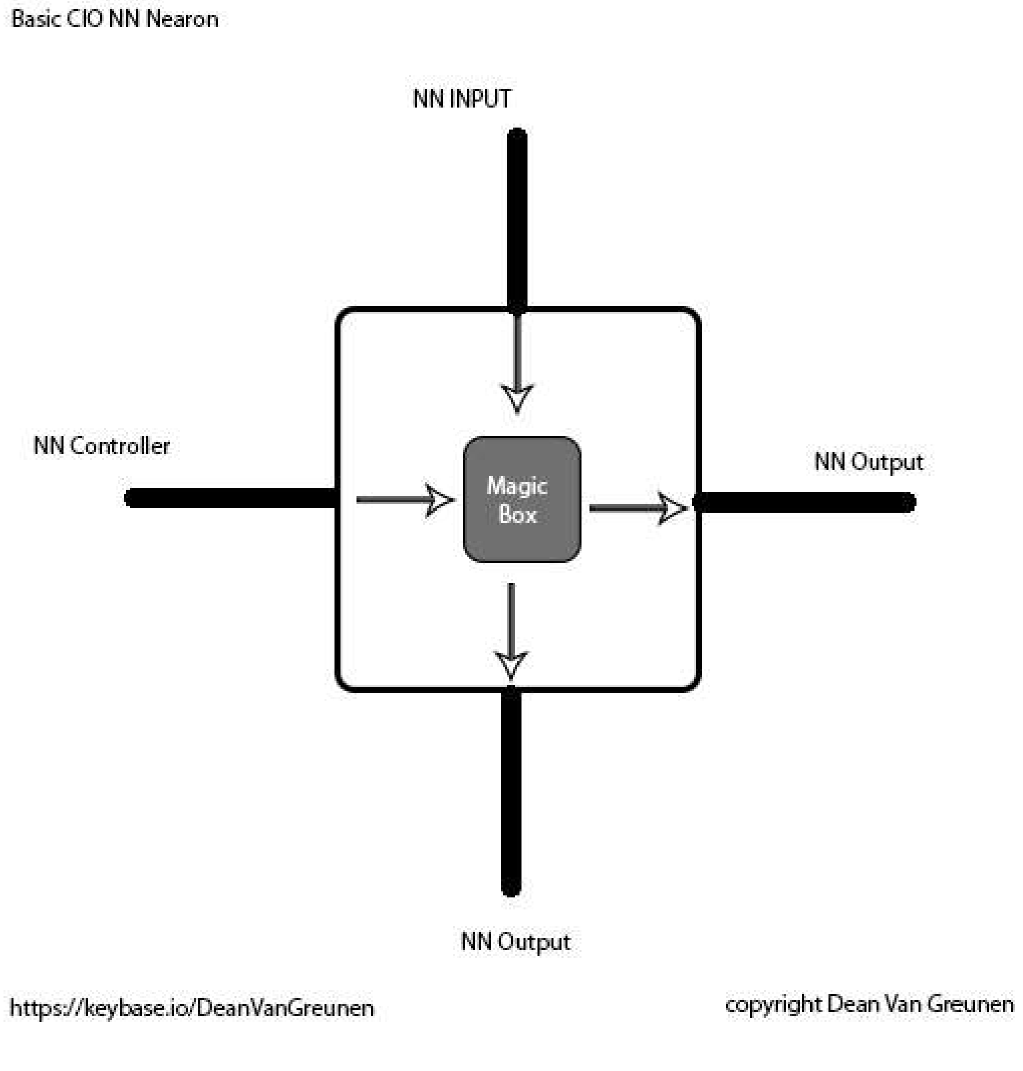
note due to a typo the "**nearon**" means "**neron**"

For this we have to redefine the Nearon

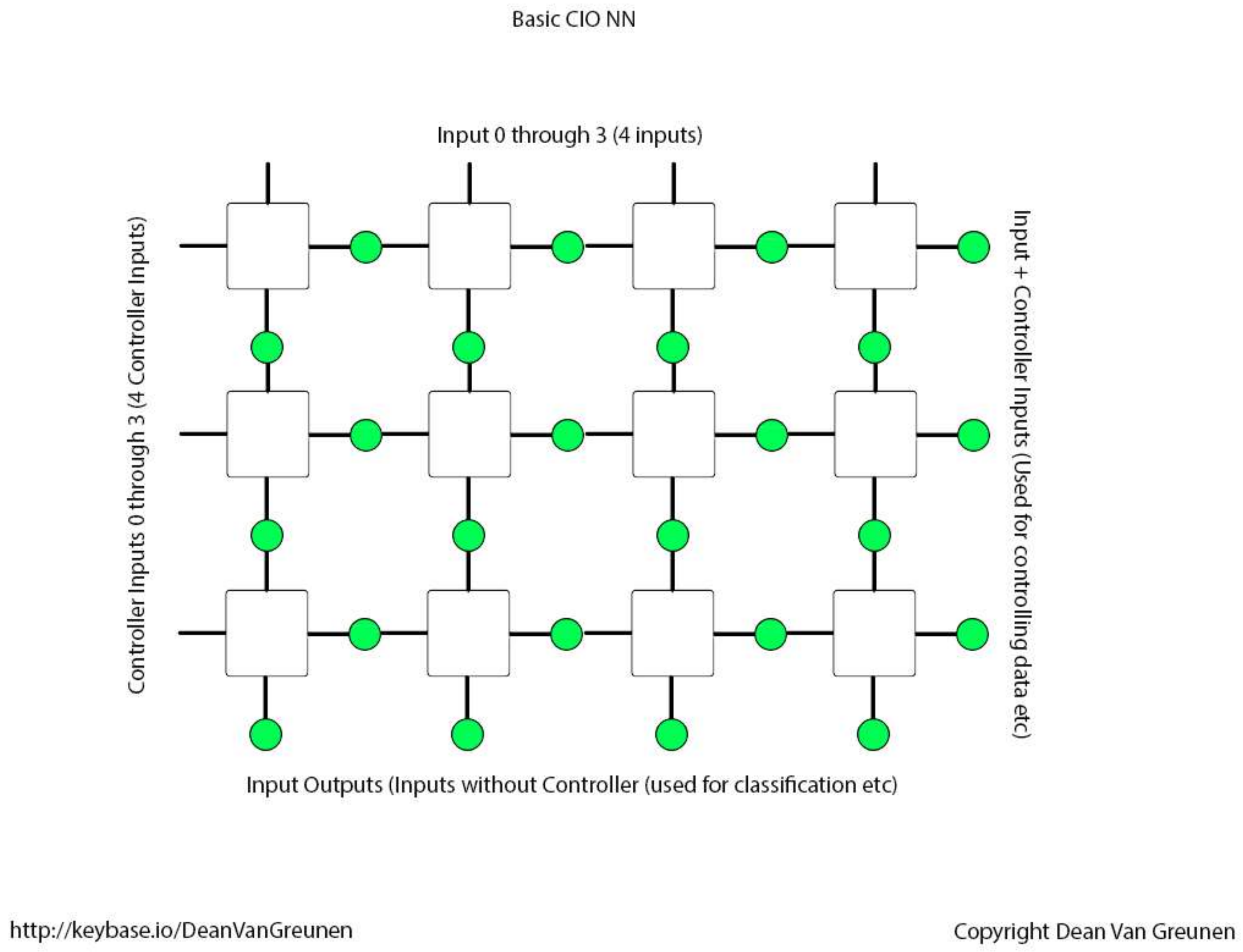
- 2 Inputs
- 2 Outputs
- 4 Weights (each input and output have their own weights)
- Internal Memory Cell (any byte or bit or block size with variable size)
- Activation Function (Defines what weights and what inputs activate this Nearon)
- Memory Storage Function (Defines what and when this cell should store said memory or memory stream)
- Memory Transpose Function (Once activated any stored memory that the activation function can trigger will be played/pushed into the Nerual Network)
- Forget Function (Defines when and/or how and/or why these memories can be destroyed/removed based on Activation function with Memory states and any input stateses itself)

How would I implement this in the form of Code? please take note of the spec. (this is non profit/GNU v3)

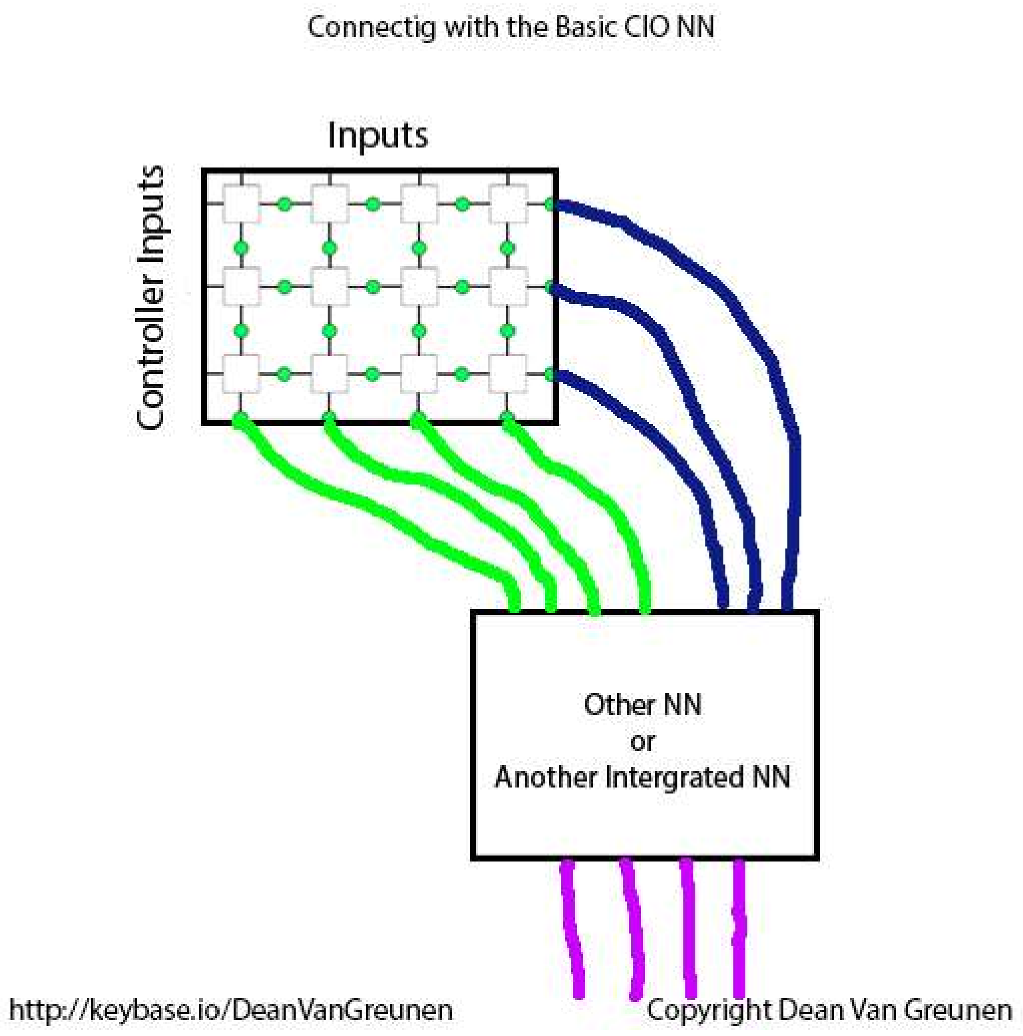
This would look something like these:



These would be arranges like this:


Basic CIO NN
Input 0 through 3 (4 inputs)
Controller Inputs 0 through 3 (4 Controller Inputs)
Input + Controller Inputs (Used for controlling data etc)
Input Outputs (Inputs without Controller (used for classification etc)
http://keybase.io/DeanVanGreunen
Copyright Dean Van Greunen

Which we can build it into this:


Connectig with the Basic CIO NN
Inputs
Controller Inputs
Other NN
or
Another Intergrated NN
http://keybase.io/DeanVanGreunen
Copyright Dean Van Greunen

it can be trained like this:

- do normal NN training from the inputs to the input outputs like a hidden layer NN

then trained to be controlled like this:

- then set the known NN dataset (inputs) to be corrected to actual or correct values via the CI (Controller Input) which will be outputed on the "Input + Controller Input" Output

by training like this:

- you have a normal NN with hidden layers which can be trainned (this can be done with CNNs) with backpropergation, and a cost function (use least amount of nerons) etc
- now you can allow the CIO NN to retrain / teach itself with supervised or unsupervised learning.
- you can combine the "Input Output" and the "Input + Controller Input" Output with another NN which can then connect with this NN in a similiar way that a Neron connects to a Neron