

Neural networks in julia

Simple neural net

There is a thing for nets in LaTeX

NN set up

```
Dense(16, 32, relu),  
    Dense(32, 64, relu),  
    Dense(64, 150),  
    Dense(150, 30),  
    Dense(30, 1),
```

Trained of ~16000 randomly selected exaples for 20 epochs on the same set

Results

Tested on all available data

The lopsided error at extrema is explained by 32 or -32 not being included into the training set due to random selection

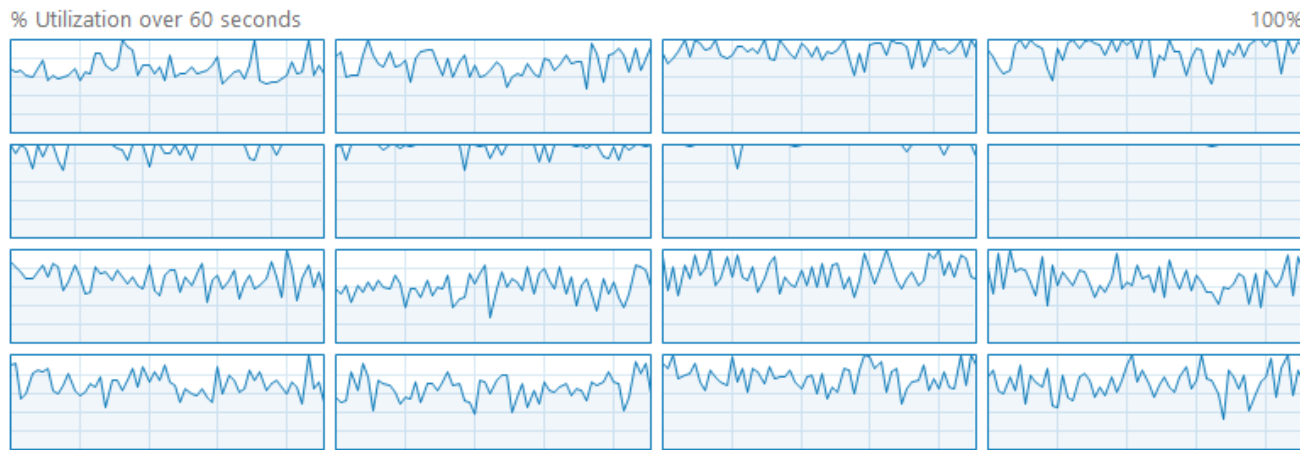
Noteworthy peculiarity when running

```
model = build_model() |> gpu  
@epochs 20 Flux.train!(loss, ps, data, opt )
```

and gpu isn't set up cpu runns very fast i.e.

CPU

AMD Ryzen 7 4800H with Radeon Graphics



Utilization	Speed		Base speed:	2.90 GHz
82%	4.08 GHz		Sockets:	1
			Cores:	8
Processes	Threads	Handles	Logical processors:	16
260	3830	115216	Virtualization:	Enabled
Up time			L1 cache:	512 KB
0:07:48:53			L2 cache:	4.0 MB
			L3 cache:	8.0 MB

This doesn't happen when running a CNN
even though the code is virtually the same

CPU

AMD Ryzen 7 4800H with Radeon Graphics



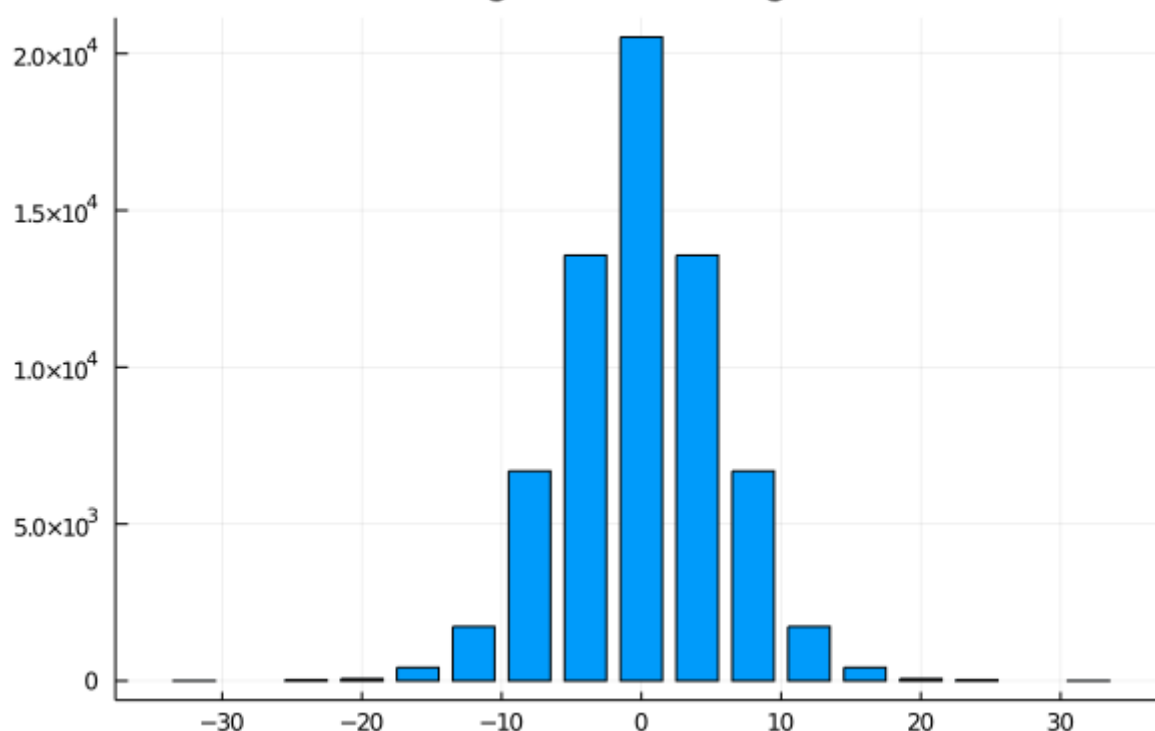
Utilization	Speed	Base speed:	2.90 GHz
19%	3.55 GHz	Sockets:	1
Processes	Threads	Cores:	8
257	3559	Logical processors:	16
Handles		Virtualization:	Enabled
114334		L1 cache:	512 KB
Up time		L2 cache:	4.0 MB
0:08:08:47		L3 cache:	8.0 MB

CNN

```
Conv((2,2),1=>5,relu),
Conv((2,2), 5=>3,pad=(1,1), relu),
Conv((2,2), 3=>3,pad=(1,1), relu),
Conv((2,2), 3=>5,relu),
Conv((2,2), 5=>3,pad=(1,1), relu),
Conv((2,2), 3=>3,pad=(1,1), relu),
Conv((2,2), 3=>5,relu),
Flux.flatten,
Dense(125, 1),
```

Trained on **Flat** energy distribution distribution of data
for 20 epochs

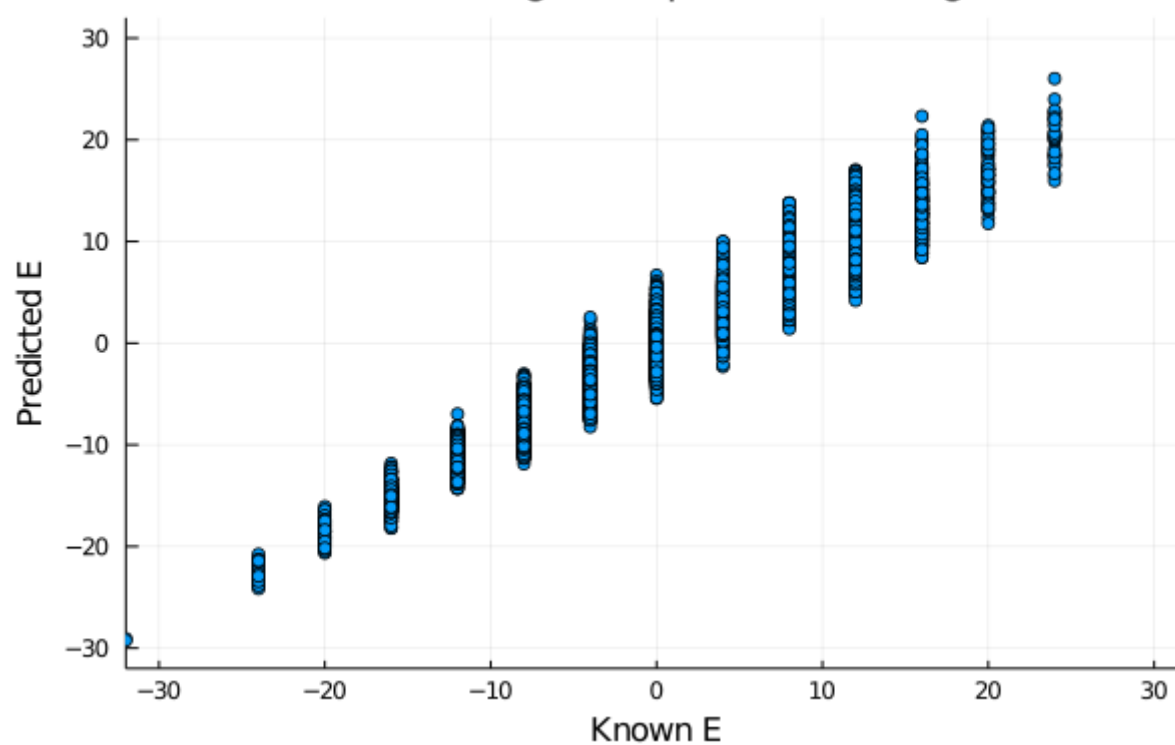
Histogram of training Data



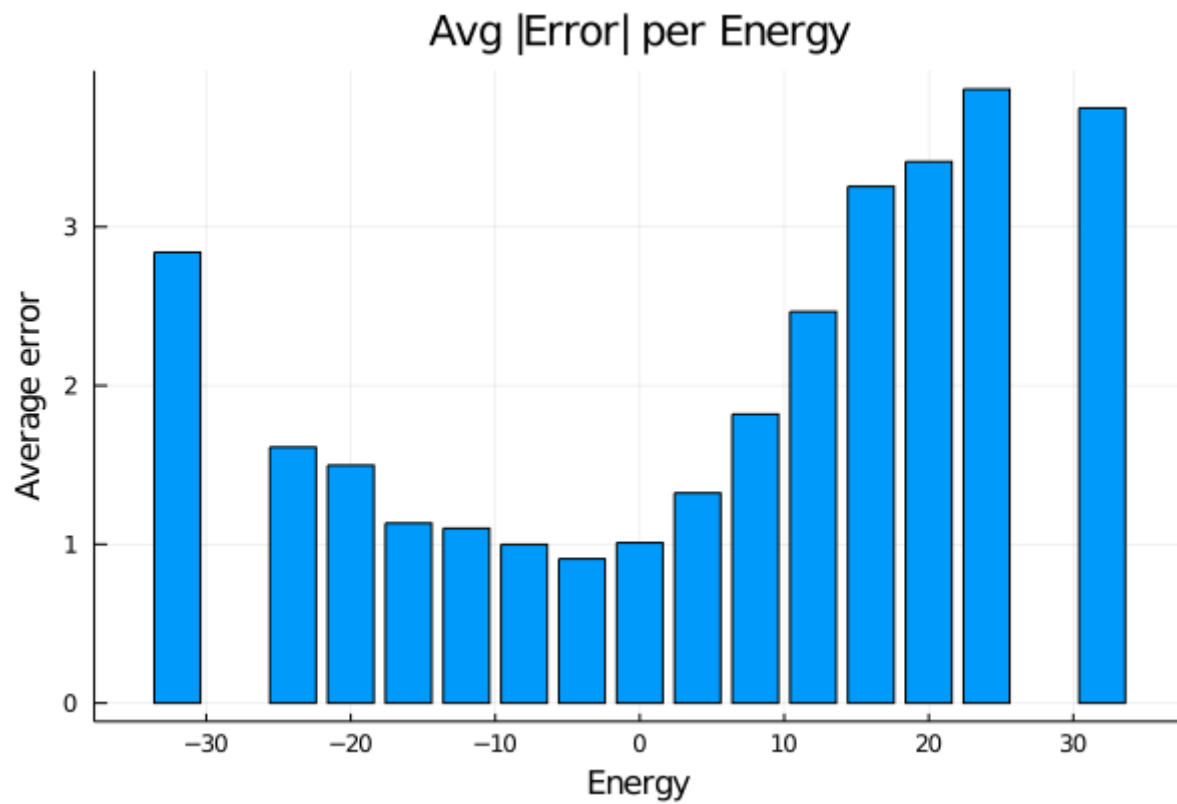
When evaluated over all data

the results are much worse

Known energies vs predicted energies



There is clearly less error on the boundaries where there were a lot of examples 32 and -32



Then repeated over **Random** data distribution

Finally, NN on random vs NN on random vs CNN on random vs CNN on flat