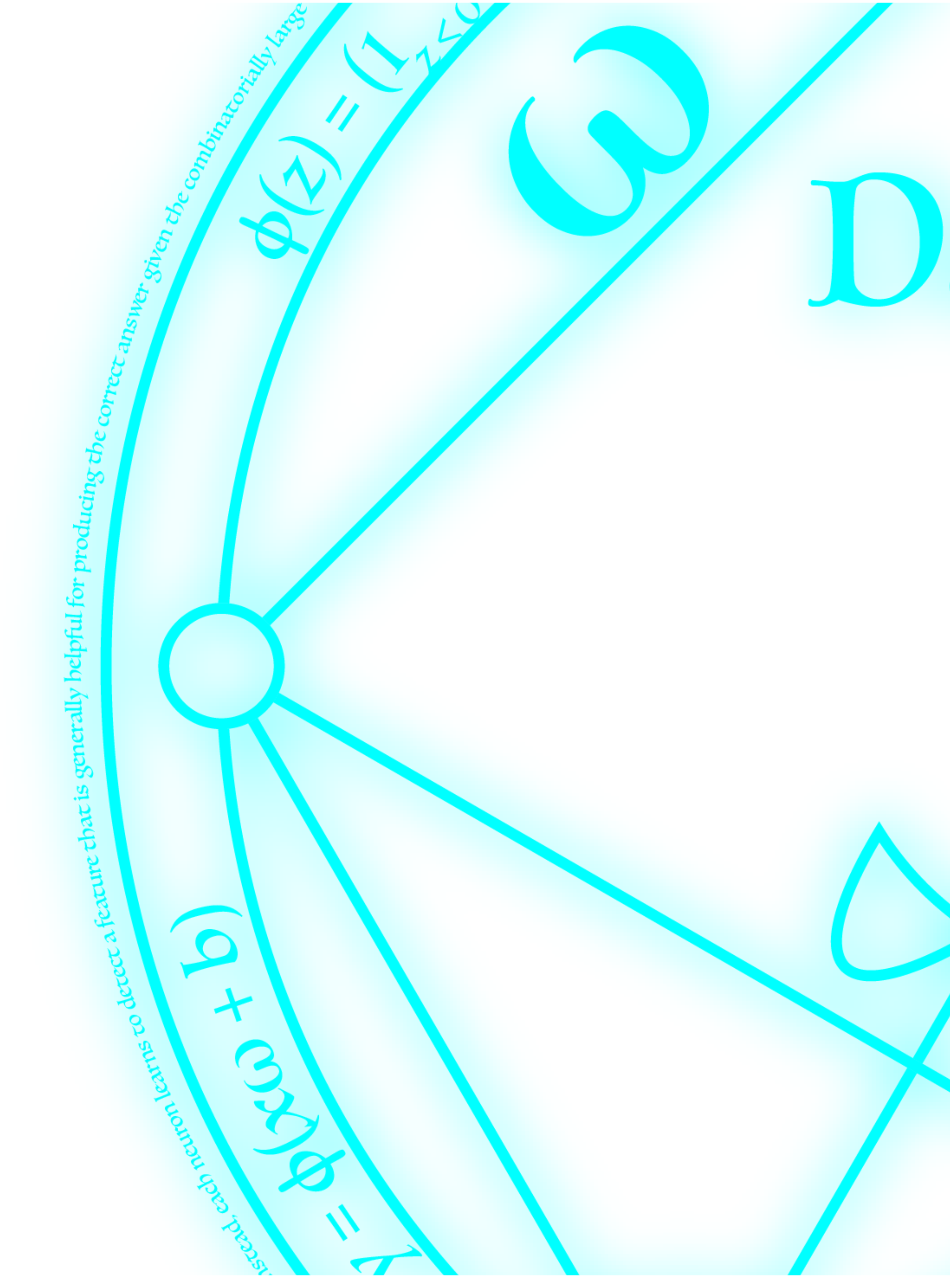
- home
- darknet

- coq tactics
- publications
- projects
- résumé


stead, each neuron learns to detect a feature that is generally helpful for producing the correct answer given the combinatorially large
y = ϕ(xω + b)
ϕ(z) = (1
z<0
D

Tiny Darknet

I've heard a lot of people talking about SqueezeNet.

SqueezeNet is cool but it's JUST optimizing for parameter count. When most high quality images are 10MB or more why do we care if our models are 5 MB or 50 MB? If you want a small model that's actually FAST, why not check out the Darknet reference network? It's only 28 MB but more importantly, it's only 800 million floating point operations. The original Alexnet is 2.3 billion. Darknet is 2.9 times faster and it's small and it's 4% more accurate.

So what about SqueezeNet? Sure the weights are only 4.8 MB but a forward pass is still 2.2 billion operations. Alexnet was a great first pass at classification but we shouldn't be stuck back in the days when networks this bad are also this slow!

But anyway, people are super into SqueezeNet so if you really insist on small networks, use this:

Tiny Darknet

Model	Top-1	Top-5	Ops	Size
AlexNet	57.0	80.3	2.27 Bn	238 MB
Darknet Reference	**61.1**	**83.0**	**0.81 Bn**	28 MB
SqueezeNet	57.5	80.3	2.17 Bn	4.8 MB
Tiny Darknet	58.7	81.7	0.98 Bn	**4.0 MB**

The real winner here is clearly the Darknet reference model but if you *insist* on wanting a small model, use Tiny Darknet. Or train your own, it should be easy!

Here's how to use it in Darknet (and also how to install Darknet):

```
git clone https://github.com/pjreddie/darknet
cd darknet
make
wget https://pjreddie.com/media/files/tiny.weights
./darknet classify cfg/tiny.cfg tiny.weights data/dog.jpg
```

Hopefully you see something like this:

```
data/dog.jpg: Predicted in 0.160994 seconds.
malamute: 0.167168
Eskimo dog: 0.065828
dogsled: 0.063020
standard schnauzer: 0.051153
Siberian husky: 0.037506
```

Here's the config file: tiny.cfg

The model is just some 3x3 and 1x1 convolutional layers:

```
layer     filters    size              input                output
    0 conv     16  3 x 3 / 1   224 x 224 x   3   ->   224 x 224 x  16
    1 max          2 x 2 / 2   224 x 224 x  16   ->   112 x 112 x  16
    2 conv     32  3 x 3 / 1   112 x 112 x  16   ->   112 x 112 x  32
    3 max          2 x 2 / 2   112 x 112 x  32   ->    56 x  56 x  32
    4 conv     16  1 x 1 / 1    56 x  56 x  32   ->    56 x  56 x  16
    5 conv    128  3 x 3 / 1    56 x  56 x  16   ->    56 x  56 x 128
    6 conv     16  1 x 1 / 1    56 x  56 x 128   ->    56 x  56 x  16
    7 conv    128  3 x 3 / 1    56 x  56 x  16   ->    56 x  56 x 128
    8 max          2 x 2 / 2    56 x  56 x 128   ->    28 x  28 x 128
    9 conv     32  1 x 1 / 1    28 x  28 x 128   ->    28 x  28 x  32
   10 conv    256  3 x 3 / 1    28 x  28 x  32   ->    28 x  28 x 256
   11 conv     32  1 x 1 / 1    28 x  28 x 256   ->    28 x  28 x  32
   12 conv    256  3 x 3 / 1    28 x  28 x  32   ->    28 x  28 x 256
   13 max          2 x 2 / 2    28 x  28 x 256   ->    14 x  14 x 256
   14 conv     64  1 x 1 / 1    14 x  14 x 256   ->    14 x  14 x  64
   15 conv    512  3 x 3 / 1    14 x  14 x  64   ->    14 x  14 x 512
   16 conv     64  1 x 1 / 1    14 x  14 x 512   ->    14 x  14 x  64
   17 conv    512  3 x 3 / 1    14 x  14 x  64   ->    14 x  14 x 512
   18 conv    128  1 x 1 / 1    14 x  14 x 512   ->    14 x  14 x 128
   19 conv   1000  1 x 1 / 1    14 x  14 x 128   ->    14 x  14 x1000
   20 avg                       14 x  14 x1000   ->  1000
   21 softmax                                        1000
   22 cost                                           1000
```