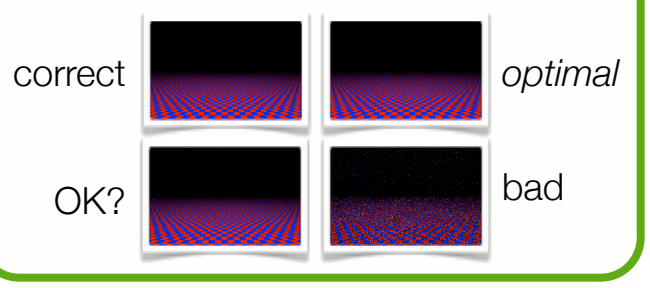
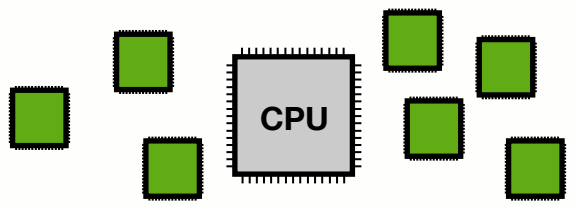
# Neural Acceleration for General-Purpose Approximate Programs

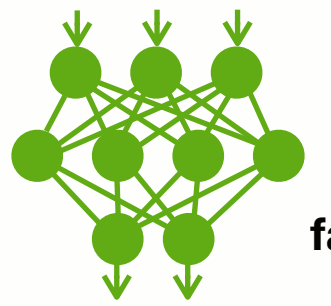
**Approximate computing** research: many applications can tolerate small errors during execution.




Technology trends mean that future chips will benefit from **specialization** and **acceleration**.




Hardware neural networks are:

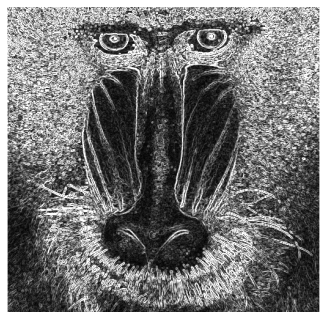


**flexible**
**low power**
**parallel**
**regular**
**fault tolerant**
**parallel**

We show that **neural networks can approximate many functions written in conventional programming languages.** We propose an algorithmic transformation and hardware accelerator that improves programs' performance and energy efficiency with very little accuracy loss.

## programming

edge detection



```
[[NPU]]
float grad(float[3][3] p) {
  …
}

void edgeDetection(Image &src,
                   Image &dst) {
  for (int y = …) {
    for (int x = …) {
      dst[x][y] =
          grad(window(src, x, y));
    }
  }
}
```

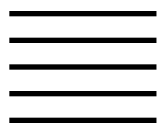
The programmer marks code that is **hot**, **approximable**, and has **well-defined inputs and outputs**. Developers also provide a small set of **representative test inputs**. The rest of the process is automatic.

## compilation

### Code observation:

test inputs

instrumented runs

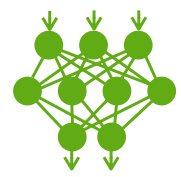


training pairs

### Training:

backpropagation

topology selection

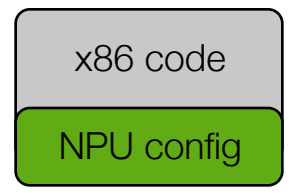


NN config

### Code generation:

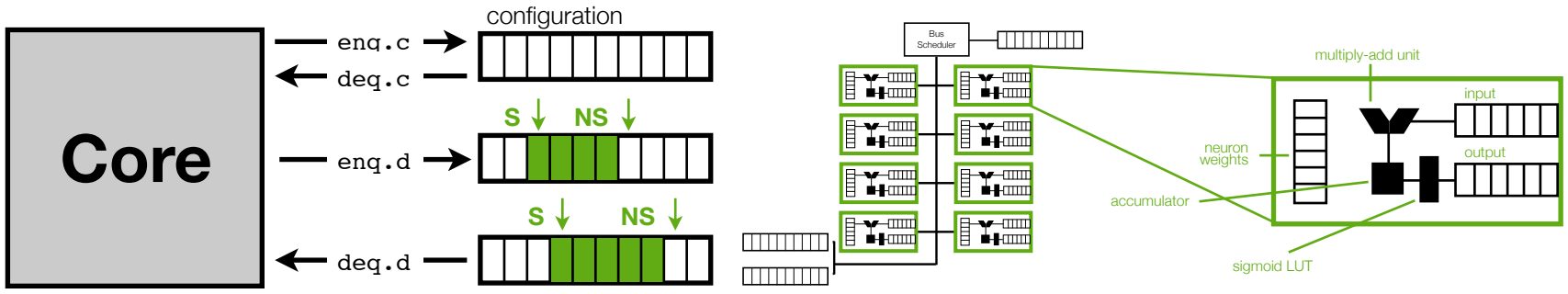
NPU scheduling

insert NPU instructions




hybrid binary

## architecture




Tight coupling allows low-latency communication with the core.

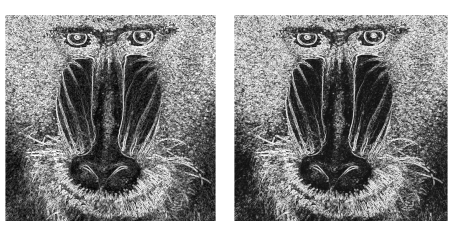
## results

**6**
applications
MARSSx86 simulation
McPAT/CACTI for power

**2.3×**
mean application speedup
0.8× – 11×

**3.0×**
mean energy reduction
1.1× – 21×

**<10%**
output quality loss
3.4% – 9.6%




Hadi Esmaeilzadeh
Adrian Sampson
Luis Ceze
Doug Burger



MICRO 2012
Vancouver, BC


http://sampa.cs.washington.edu/sampa/NPU