

Single GPU Classifier

Deep Learning / AI Tasks

- OCR and scene text
- visual object recognition
- speech recognition
- medical classification and prediction
- stock market classification and prediction
- image synthesis / graphics
- control and robotics

Example: Imagenet

- data
 - collection of 1 million images (from the web)
 - 1000 labels (manually labeled)
- task
 - predict the label from the image
 - supervised training
- model
 - deep convolutional neural network

Deep Learning Training Steps

- data collection
- data labeling
- data augmentation
- model training
- model validation
- model deployment

Model Structure

- deep learning models are composed of *layers*
- each layer usually consists of:
 - a linear operation (matrix multiplication, convolution)
 - an element-wise nonlinearity (sigmoid, tanh, ReLU)
- it is the linear operations that are computationally expensive

Simple one hidden layer network:

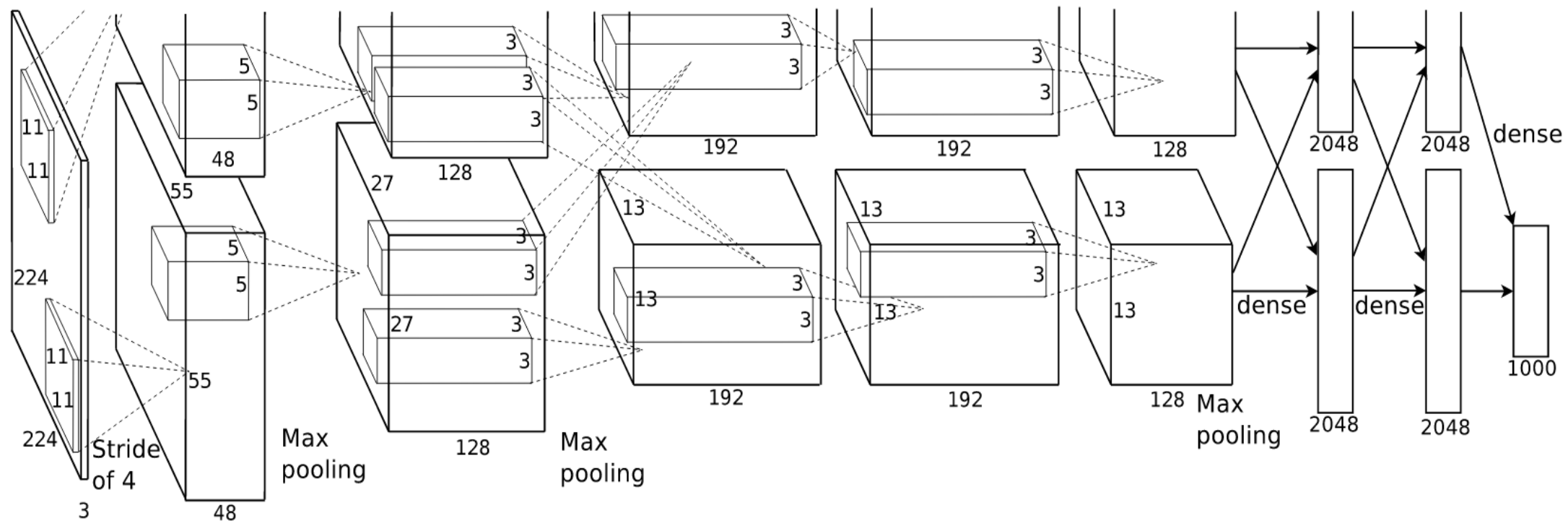
$$f(x) = \sigma(M_2 \cdot \sigma(M_1 \cdot x + b_1) + b_2)$$

Building Models from Modules in PyTorch

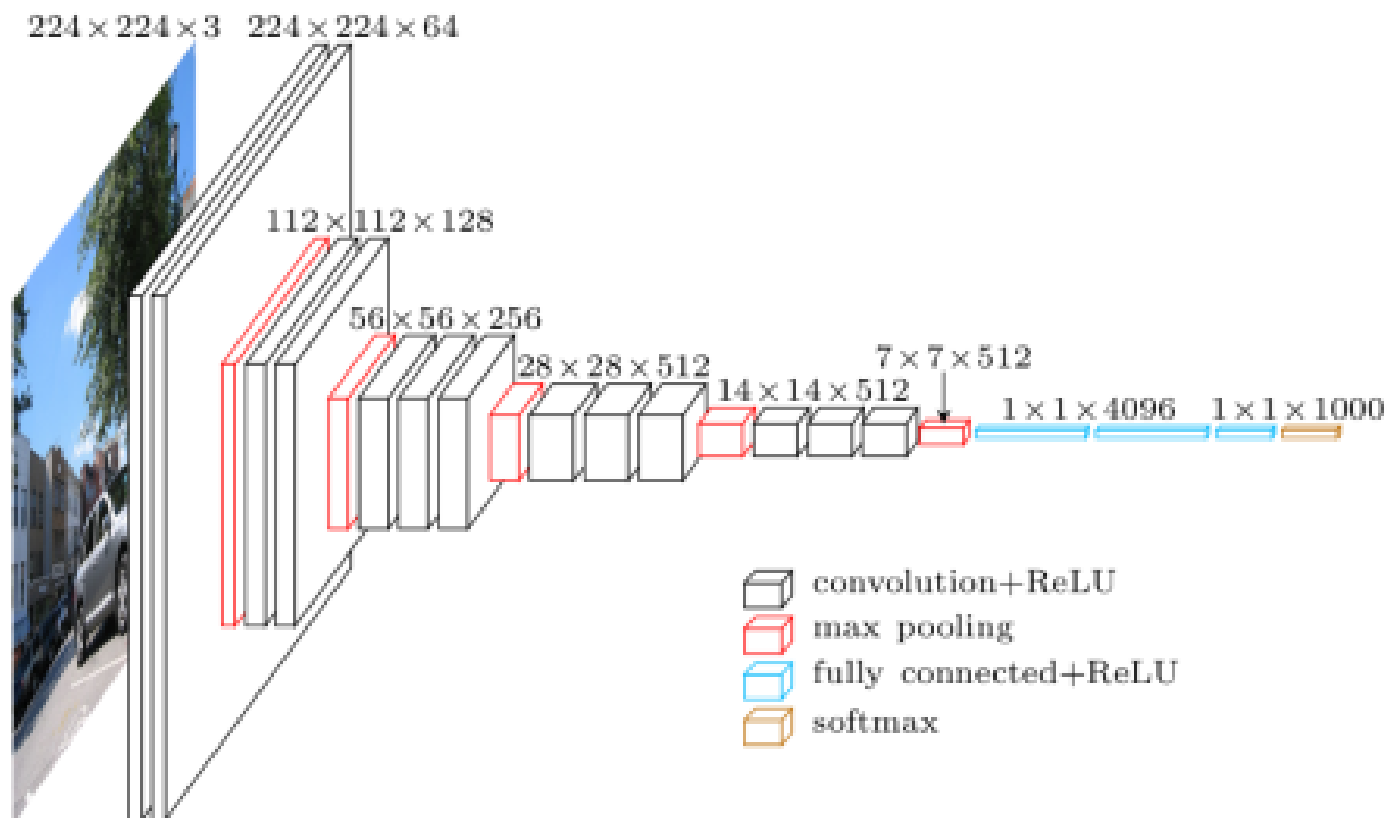
```
model = Sequential(  
    Conv2d(3, 20),  
    ReLU(),  
    Conv2d(20, 40),  
    ReLU(),  
    Flatten(),  
    Linear(40*24*24),  
    ReLU()  
)
```

Other styles: functional computations, mixed, ...

Alexnet

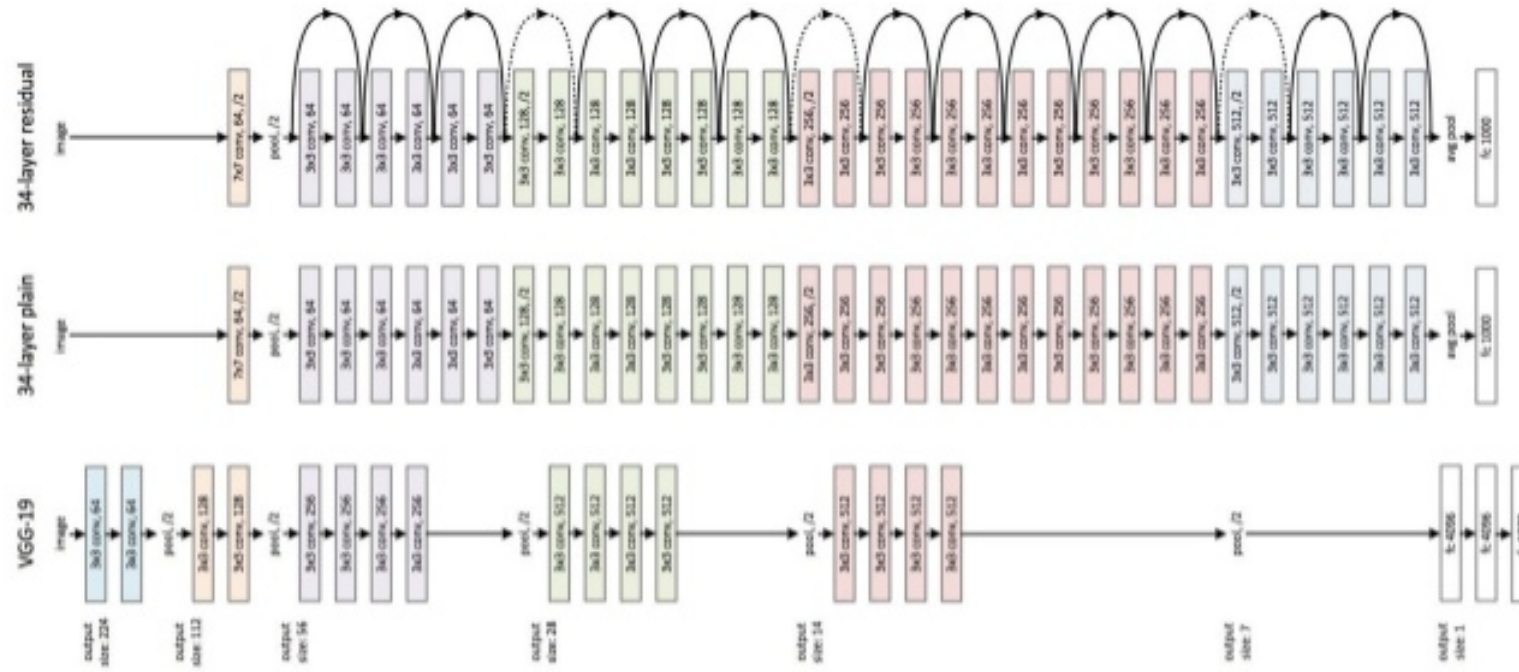


VGG16 Model



Resnet-50 Model

ResNet



Deep Learning = Stochastic Gradient Descent Optimization

- dataset consists of input, target pairs (x, y)
- the model is a function f depending on parameters θ
- repeatedly (by shuffled epoch):
 - create batches of 1-10000 (input, target) pairs
 - compute predicted output \tilde{y}
 - compute the derivative of the error $||y - \tilde{y}||^2$ wrt θ
 - update θ with a small multiple of the derivative

GPU Performance

GeForce RTX 2080 Ti (about \$1000)

- 14.2 TFLOPS¹ of peak single precision (FP32) performance
- 28.5 TFLOPS¹ of peak half precision (FP16) performance
- 14.2 TIPS¹ concurrent with FP, through independent integer execution units
- 113.8 Tensor TFLOPS^{1,2}
- 10 Giga Rays/sec
- 78 Tera RTX-OPS

Forms of Parallelism

- MIMD
 - shared: memory
 - separate: fetch/decode, execute, registers
- SIMT (lockstep threads; roughly, the GPU model)
 - shared: memory, fetch/decode
 - separate: execute, registers
- SIMD
 - shared: memory, fetch/decode, registers
 - separate: execute

PyTorch Data Loading

```
def augment(image): ...

dataset = datasets.ImageNet("/archive/imagenet", augment)
loader = DataLoader(dataset, batch_size=16, shuffle=True, num_workers=4)

for input_batch, target_batch in loader:
    ...
```

PyTorch Training Loop

```
model = make_model()  
model = model.cuda()  
  
optimizer = optim.SGD(model.parameters(), lr=0.01)  
  
for x, target in samples:  
    optimizer.zero_grad()  
    pred = model(x.cuda()).cpu()  
    loss = F.mse_loss(target, pred)  
    loss.backward()  
    optimizer.step()
```

Often we wrap up this functionality into a `Trainer` class.
See the notebook for a full, running implementation.

Docker

- a full AI/DL environment contains dozens of major software libraries, all interacting
- all this can be encapsulated into a single "Docker image"
- Docker images look like virtual machines, but are actually just namespaces
- create anything from fully encapsulated to regular command line
- create standalone executables with tools like Singularity