DS 4440 Computation Graphs

So far:

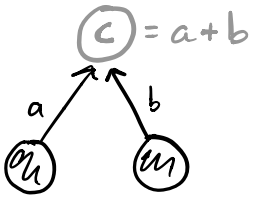
- Supervised learning
- SGD (for linear models)
- MLPs & non-linear models

Today Generalize from MLPs to Computation Graphs, the core abstraction for NNs.

- Intro Computation Graphs as a formalism
- Work through "forward passes"
- Review in Torch

The Big Idea (TM)

Models as Directed Acyclic Graphs (DAGs)

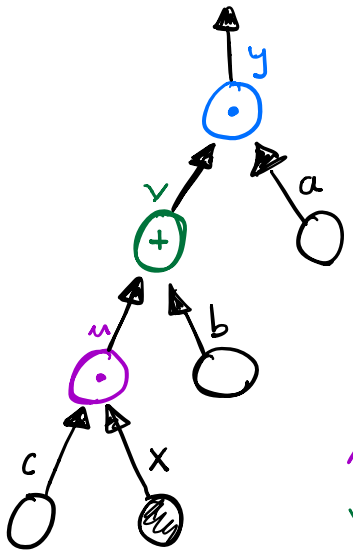


$c = a + b$

- Nodes correspond to operations and often labeled as variables
- Tensors "flow" along edges
- We care about forward and backward passes through the graph

Consider $y = a \cdot (b + c \cdot x)$

$u = c \cdot x$, $v = u + b$



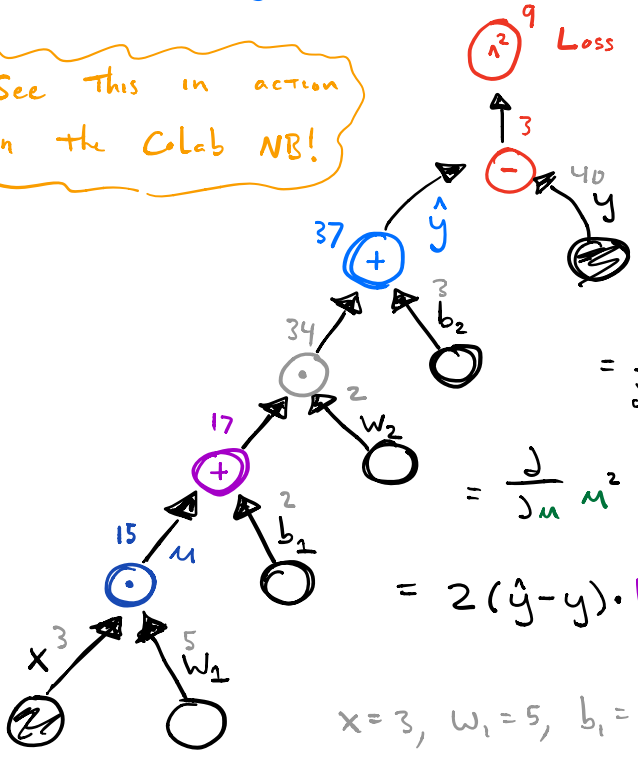
In a forward pass we traverse nodes in topological order, passing variables along edges.

Let: $x = 3$, $c = 5$, $b = 2$, $a = 3$

$u = c \cdot x = 15$ $\quad$ $v = u + b = 17$ $\quad$ $y = a \cdot v = 51$

Let's build a computation graph for a regression model: $\hat{y} = w_2 \cdot (w_1 \cdot x + b_1) + b_2$ [Exercise]

See this in action in the Colab NB!



$$\frac{\partial}{\partial w_2} Loss = \frac{\partial}{\partial w_2} (\hat{y} - y)^2$$

$$= \frac{\partial}{\partial w_2} (\underbrace{(w_2 \cdot [w_1 \cdot x + b_1] + b_2) - y}_{u})^2$$

$$= \frac{\partial}{\partial u} u^2 \; \frac{\partial}{\partial w_2} (\underbrace{w_2 \cdot [w_1 \cdot x + b_1]}_{h} + b_2 - y)$$

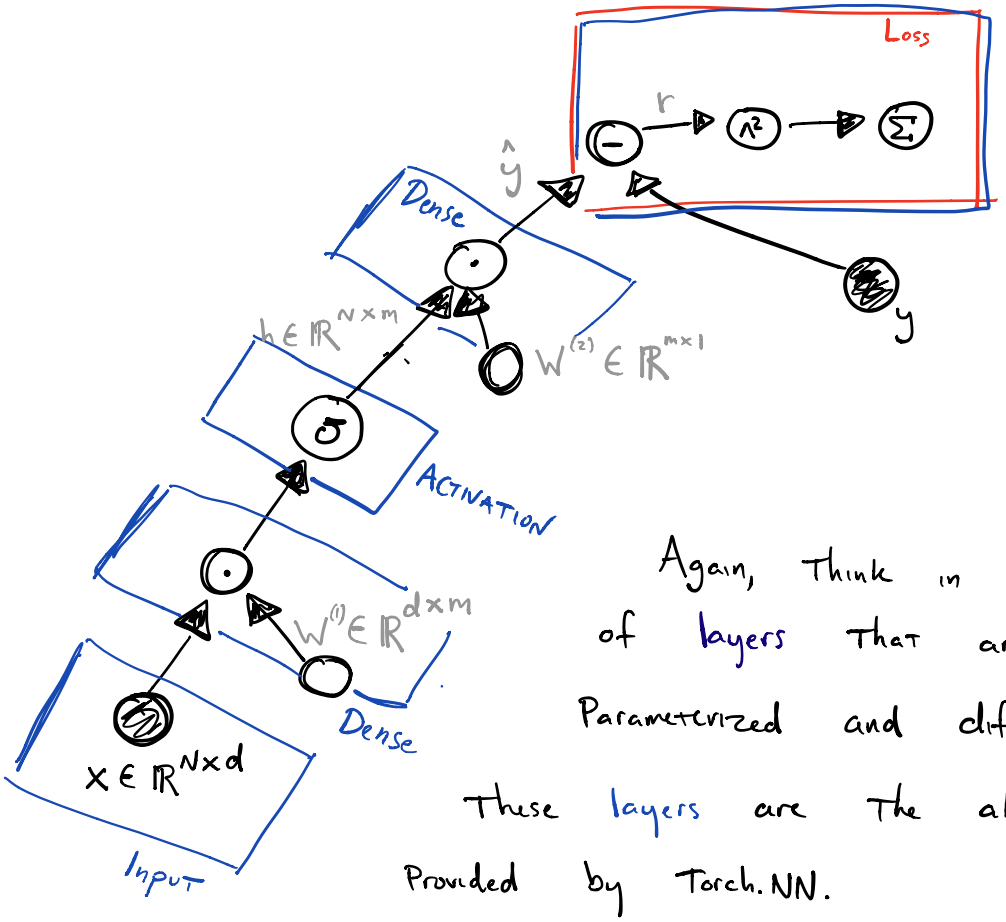
$$= 2(\hat{y} - y) \cdot h$$

$x = 3$, $w_1 = 5$, $b_1 = 2$, $w_2 = 2$, $b_2 = 3$ $\quad$ $y = 40$

So far we've been looking at univariate functions; let's consider a multivariable MLP with a non-linear activation. We'll stick with regression.

$$\hat{y} = W^{(2)} \cdot \sigma(W^{(1)} \cdot x)$$

$$\mathcal{L}(y, \hat{y}) = \sum_i (y_i - \hat{y}_i)^2$$

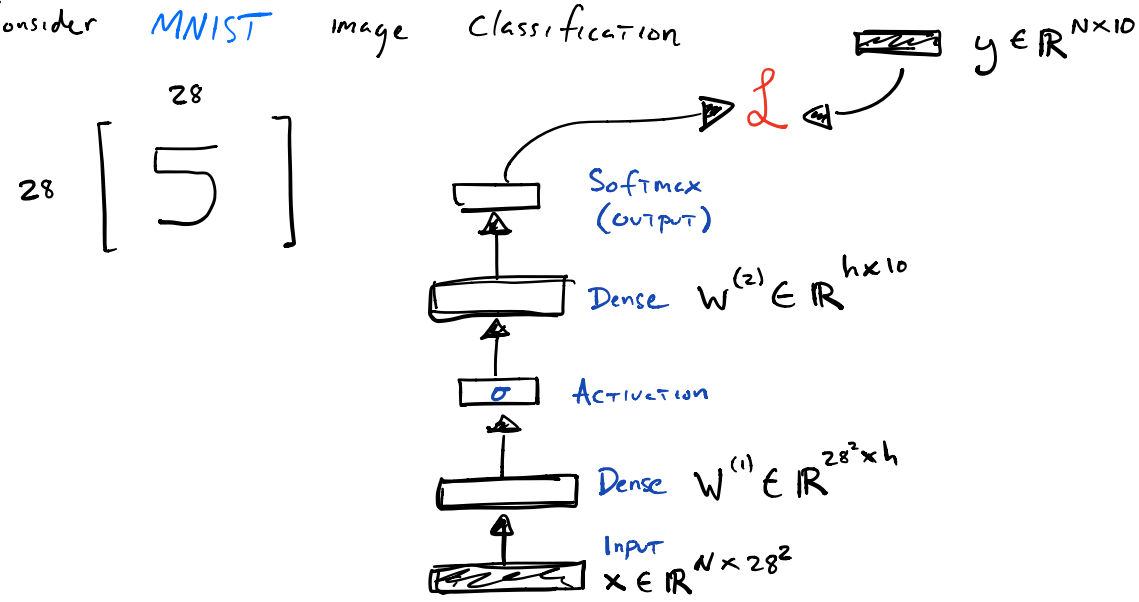


Again, think in terms of layers that are parameterized and differentiable. These layers are the abstraction provided by Torch.NN.

Why layers?

- Suppose we want to change to a classification: What needs to change?
- Complexity: Layers can be complicated (RNNs, ConvNets ...); we abstract away from this

Consider MNIST image classification

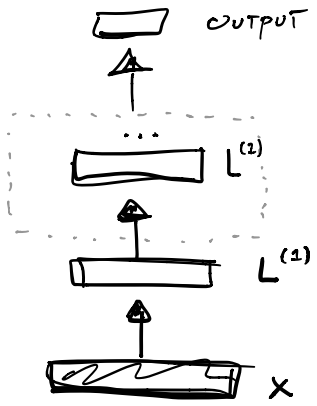


Static computation graphs → Define and Compile model, Pass all batches Through

Dynamic computation graphs → Assemble graph as needed i.e., before each forward Pass

Dynamic graphs can be particularly useful for variable length inputs like text or sequences generally.

Can also do wacky (cool!) things like have stochastic structures



Let's see in CoLab.