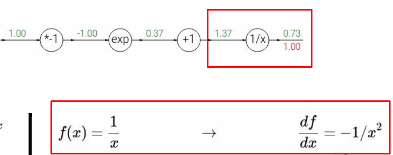
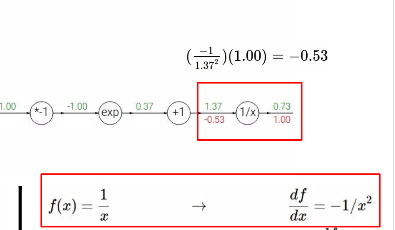
BP 和Neural Networks

BP的基本阐述：

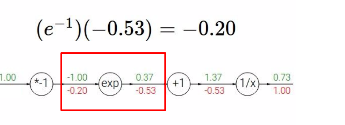
1. 首先计算出各个局部函数的直接梯度表达式。
2. 计算每个梯度的实际值。但是要注意下列计算：



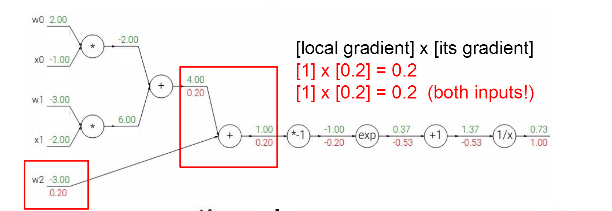


在计算梯度时，我们的x的取值前向传播时传到这个函数里的x的值，而不是这个函数计算得出的值。

1. 运用链式法则，反向向前传播梯度值。



从这里可以看出在储存的时候一个要记录前向传播的值，一个要记录反向传播的梯度。



Pattern in backward flow

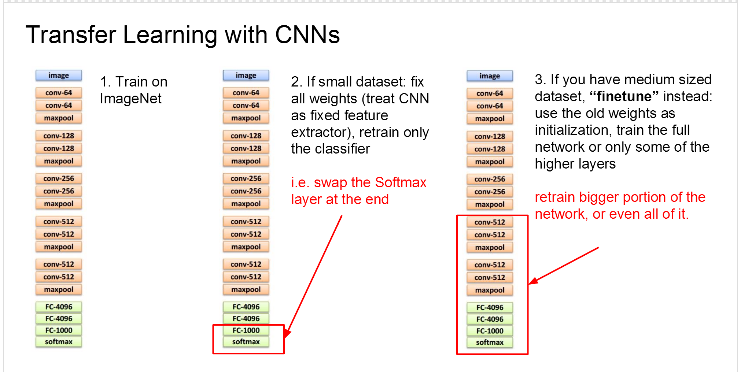
Add gate: gradient distributor

Max gate: gradient router

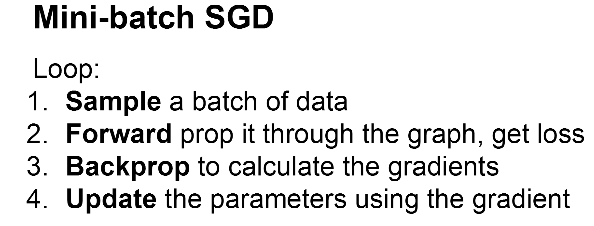
Mul gate: gradient “switcher”

梯度的矩阵运算：

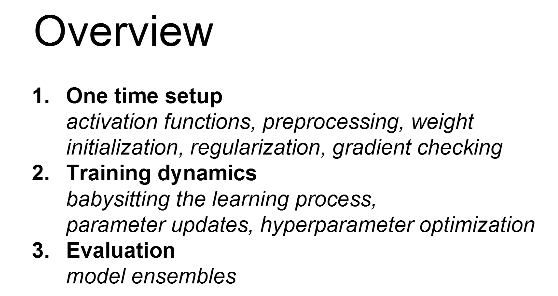
训练处的细节：

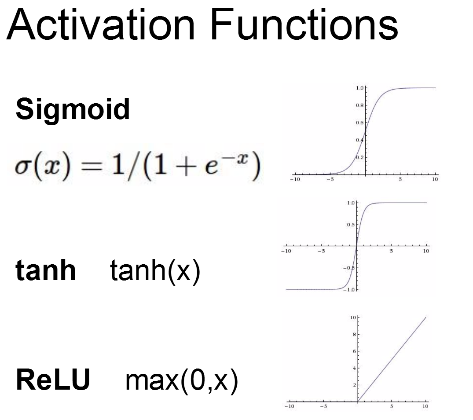


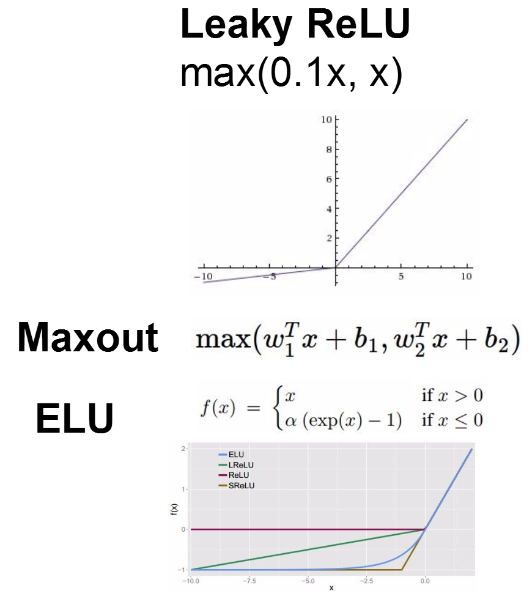
Mini-batch SGD



训练神经网路：







Sigmoid(3 problem):

1. Saturated neurons kill the gradients
2. Sigmoid outputs are not zero-centered
3. Exp() is a bit compute expensive

Tanh(x):

Zero centered(nice)

Still kills gradients when saturated.

ReLU:

Does not saturate(in +region)

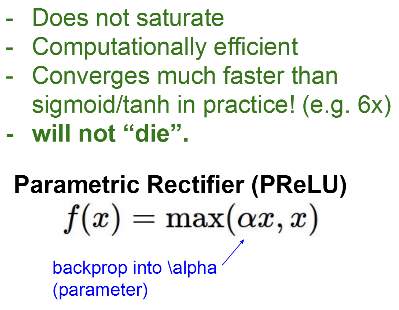
Very computationally efficient

Converges much faster than sigmoid/tanh in practice

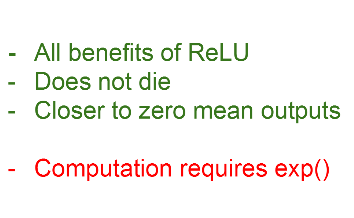
Not zero-centered output

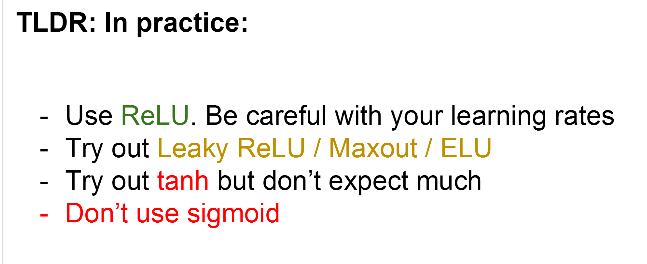
When x < 0,gradient ==0

Leaky ReLU:



Exponential linear units(ELU):





Preprocess the data

1. Zero-centered data(subtract the mean iamge(e.g. alexnet);subtract per-channel mean(vggnet))
2. Normalized data(not common to normalize variance,to do PCA or whitening)
3. Pca
4. Whitening

权重初始化：

First idea:small random numbers(Gaussian with zero mean and x standard deviation)

Works—okay for small networks, but can lead to non-homogeneous distributions of activations across the layers of a network.

Second idea:

Batch normalization

(usually inserted after fully connected/(or convolutional) layers,and before nonlinearity)

：

提高梯度流经整个网络

允许很高的学习率

减小了对初始化的强烈依赖

Regularization in a funny way,and slightly reduces the need for dropout,

**Babysitting the learning process**

Step1: preprocess the data

Step2:choose the architecture

Double check that the loss is reasonable:

Training:

Make sure that you can overfit very small portion of the training data

Loss not going down:

Learning rate too low.

Loss exploding:

Learning rate too high

超参数优化：

CNN(卷积神经网络):