# TTIC 31230, Fundamentals of Deep Learning

David McAllester, April 2017

## Regularization

## Last Lecture

Controlling Vanishing and Exploding Gradients.

Initialization, Batch Normalization and Highway Architectures.

Highways:

Pure (Resnet): $L_{i+1} = L_i + D_i$

Forget Gated (LSTM): $L_{i+1} = F_i * L_i + D_i$

Exclusively Gated (GRU): $L_{i+1} = F_i * L_i + (1 - F_i) * D_i$

# Weight Norm Regularization

## Regularization as Model Bias

Many regularization methods can be viewed as imposing a learning bias — some models or parameter values are preferred over others.

Different biases yield different results and regularization often helps.

## $L_2$ Regularization (Tikhonov Regularization)

$$p(\Theta) \propto e^{-\frac{1}{2}\lambda||\Theta||^2}$$

$$\Theta^* = \underset{\Theta}{\text{argmin}} \quad \ell_{\text{train}}(\Theta) \quad + \quad \frac{1}{2}\lambda||\Theta||^2$$

$$\Theta \texttt{ -= } \eta \nabla_\Theta \, \ell_{\text{train}}(\Theta)$$

$$\Theta \texttt{ -= } \eta\lambda\Theta \qquad \text{(shrinkage)}$$

At equilibrum these two updates must sum to zero giving

$$\Theta = \frac{-1}{\lambda} \nabla_\Theta \, \ell_{\text{train}}(\Theta)$$

# $L_1$ Regularization and Sparsity

$$p(\Theta) \propto e^{-||\Theta||_1} \qquad ||\Theta||_1 = \sum_i |\Theta_i|$$

$$\Theta^* = \underset{\Theta}{\text{argmin}} \quad \ell_{\text{train}}(\Theta) \; + \; \lambda||\Theta||_1$$

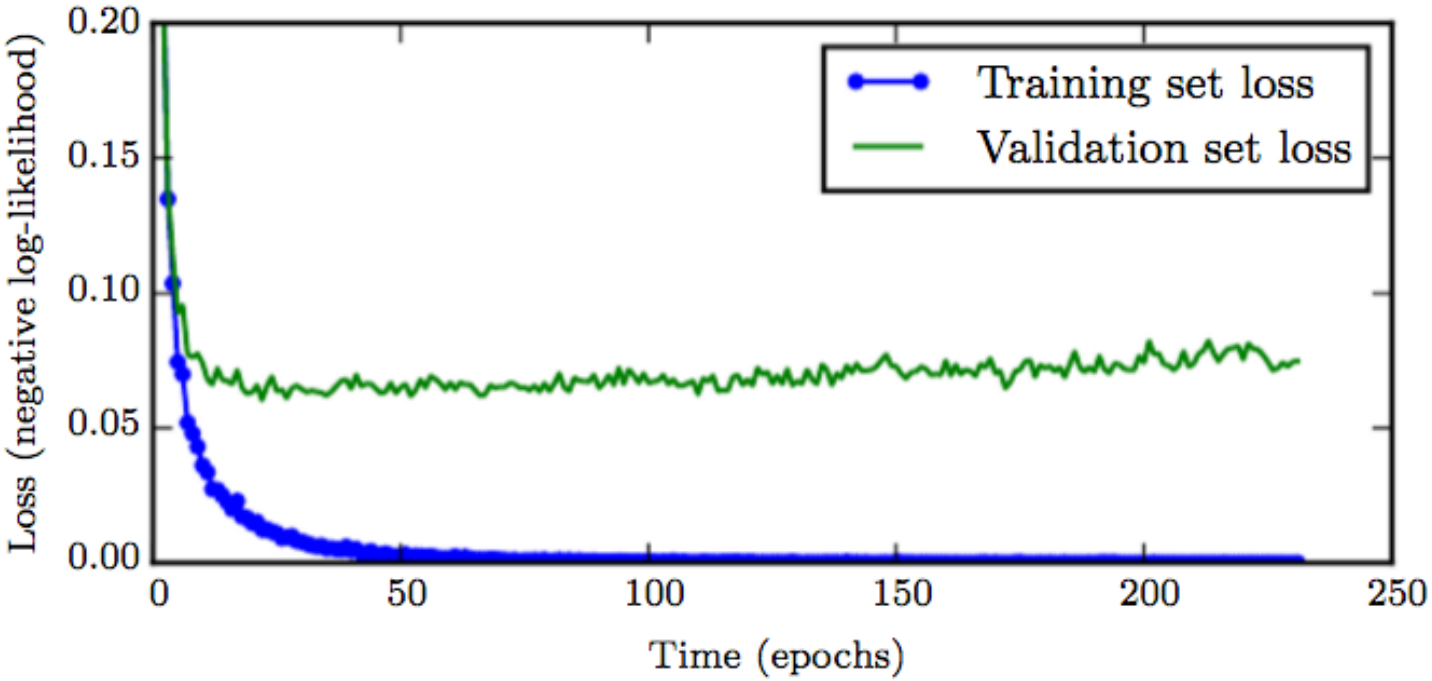
$$\Theta \texttt{ -= } \eta \nabla_\Theta \, \ell_{\text{train}}(\Theta)$$
$$\Theta_i \texttt{ -= } \eta\lambda \, \text{sign}(\Theta_i) \qquad \text{(shrinkage)}$$

At equilibrium (sparsity is difficult to achieve with SGD)

$$\begin{aligned} \Theta_i &= 0 && \text{if } |\partial\ell/\partial\Theta_i| < \lambda \\ \partial\ell/\partial\Theta_i &= -\lambda \text{sign}(\Theta_i) && \text{otherwise} \end{aligned}$$

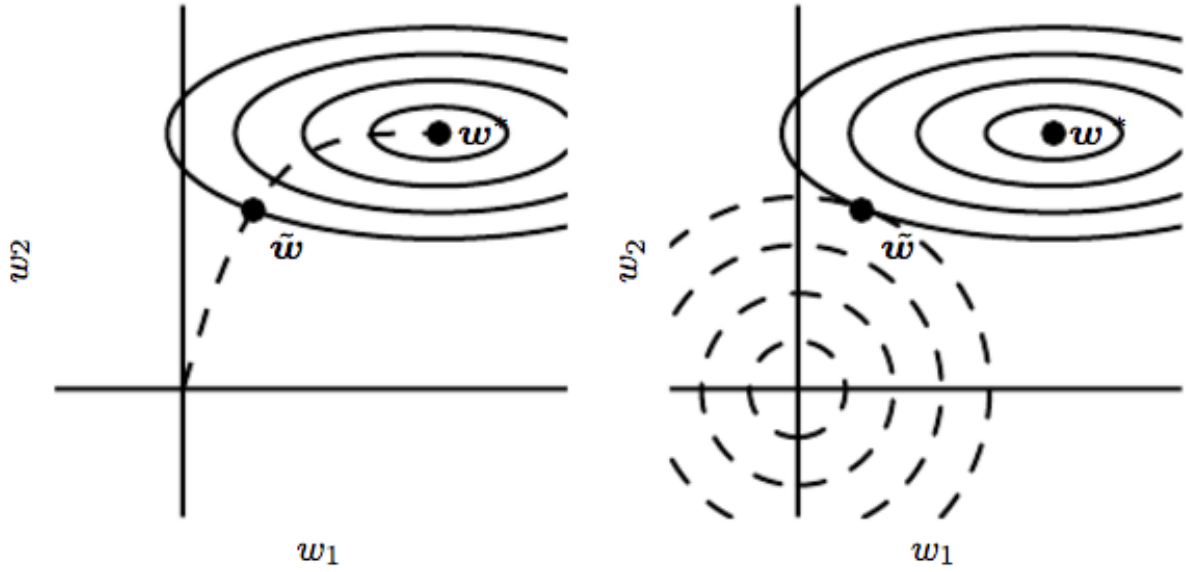
# Early Stopping

# Early Stopping




[Goodfellow et al.]

# Early Stopping



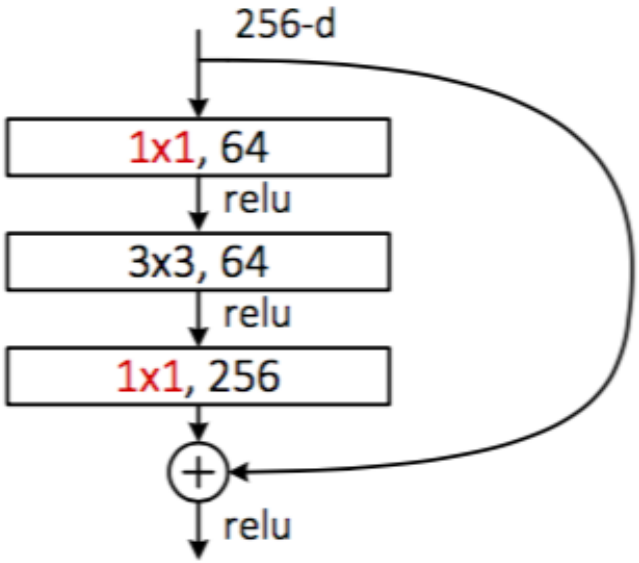

[Goodfellow et al.]

For differential Newton updates on a quadratic loss function, early stopping and $L_2$ regularization are equivalent.

# Using Fewer Parameters

## Using Fewer Parameters

We prefer models with fewer parameters. (Occam's Razor)




$$2\times256\times64 + 9\times64\times64 = 69,632$$

$$9\times256\times256 = 589,824$$

- **bottleneck**
  (for ResNet-50/101/152)

[Kaiming He]

# Sparse Activation

## Sparse Activation

We can impose an $L_1$ regularization on the activations of the network (the output of the activation function of each neuron).

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}}\, \ell(\Theta) + \lambda||h||_1$$

where $h$ is the vector of neuron activation outputs.

This will tend to make activations sparse.

## $k$-Sparse Coding (Orthogonal Matching Pursuit)

Let $W$ be a matrix where we view $W_{.,i}$ is the $i$th "dictionary vector".

For input $x$ we can construct a $k$-sparse representation $h(x)$.

$$h(x) = \underset{h,||h||_0=k}{\text{argmin}} \; ||x - Wh||^2$$

Note

$$Wh = \sum_{i \in I(x)} h_i \, W_{.,i} \qquad |I(x)| = k$$

We can now replace $x$ by its sparse code $h(x)$.

# Ensembles

## Ensembles under Square Loss

We average $k$ regression models

$$f(x) = \frac{1}{k}\sum_{i=1}^{k} f_i(x)$$

$$f(x) - y = \frac{1}{k}\sum_{i=1}^{k} (f_i(x) - y)$$

$$\epsilon = \frac{1}{k}\sum_{i=1}^{k} \epsilon_i, \quad \epsilon_i = f_i - y \quad \text{(residuals)}$$

## Ensembles

Assume that $\mathrm{E}\left[\epsilon_i^2\right] = \sigma^2$ and $\mathrm{E}\left[\epsilon_i \epsilon_j\right] = \sigma^2 \rho$ for $i \neq j$.

$$\mathrm{E}\left[\left(\frac{1}{k}\sum_i \epsilon_i\right)^2\right] = \frac{1}{k^2}\mathrm{E}\left[\sum_i \left(\epsilon_i^2 + \sum_{j \neq i} \epsilon_i \epsilon_j\right)\right]$$

$$= \frac{1}{k}\sigma^2 + \frac{k-1}{k}\sigma^2\rho = \sigma^2\left(\frac{1}{k} + \left(1 - \frac{1}{k}\right)\rho\right)$$

If Pearson's correlation $\rho = \mathrm{E}\left[\epsilon_i \epsilon_j\right] / \sigma^2 < 1$ we win.

## Ensembles Under Log Loss

For log loss we average the probability vectors.

$$P(y|x) = \frac{1}{k} \sum_i P_i(y|x)$$

$-\log P$ is a convex function of $P$. For any convex $\ell(P)$ Jensen's inequality states that

$$\ell\left(\frac{1}{k}\sum_i P_i\right) \leq \frac{1}{k}\sum_i \ell(P_i)$$

This implies that the loss of the average model cannot be worse (can only be better) than the average loss of the models.

## Finding Strong Diverse Models

In deep learning we typically get a variety of models by training under different random initializations.

"Bagging" gets diverse models by training under different random subsets of the training data.

"Random Forests" are decision trees learned under different random sets of available features for each decision tree split.

# Dropout

## Dropout

Dropout can be viewed as an ensemble method.

To draw a model from the ensemble we randomly select a mask $\mu$ with

$$\begin{cases} \mu_i = 0 \text{ with probability } \alpha \\ \\ \mu_i = 1 \text{ with probability } 1 - \alpha \end{cases}$$

Then we use the model $(\Theta,\ \mu)$ with weight layers defined by

$$y_i = \text{Relu}\left(\sum_j W_{i,j}\ \mu_j x_j\right)$$

## Dropout Training

Repeat:

- Select a random mask $\mu$

$$y_i = \text{Relu}\left(\sum_j W_{i,j}\ \mu_j x_j\right)$$

- $\Theta$ -= $\nabla_\Theta\ \ell(\Theta, \mu)$

Backpropagation must use the same mask $\mu$ used in the forward computation.

## The Weight Scaling Rule

At train time we have

$$y_i = \text{Relu}\left(\sum_j W_{i,j}\ \mu_j x_j\right)$$

At test time we have

$$y_i = \text{Relu}\left((1-\alpha)\ \sum_j W_{i,j}\ x_j\right)$$

At test time we use the "average network".

## How to Average

It is not clear whether the weight scaling rule is superior to standard model averaging defined by

$$P(y|x) = \mathrm{E}_{\mu}\left[P(y|x, \Theta, \mu)\right]$$

Goodfellow et al. (2013) found that the weight scaling rule outperformed standard model averaging.

Gal and Ghahramani (2015) found the opposite.

It seems to be model dependent.

## The Case of Least Squares Regression

Consider simple least square regression

$$\begin{aligned}
\Theta^* &= \underset{\Theta}{\operatorname{argmin}} \;\; \mathrm{E}_{(x,y)} \, \mathrm{E}_{\mu} \left[ (y - \Theta \cdot (\mu * x))^2 \right] \\
&= \mathrm{E} \left[ (\mu * x)(\mu * x)^\top \right]^{-1} \; \mathrm{E} \left[ y(\mu * x) \right] \\
&= \underset{\Theta}{\operatorname{argmin}} \;\; \mathrm{E}_{(x,y)} (y - (1-\alpha)\Theta \cdot x)^2 + \sum_i \frac{1}{2} (\alpha - \alpha^2) \mathrm{E} \left[ x_i^2 \right] \Theta_i^2
\end{aligned}$$

In this case dropout is equivalent to a form of $L_2$ regularization — see Wager et al. (2013).

## Some Claims

According to Goodfellow et al., Srivistava (2014) has shown (presented evidence?) that dropout is more effective than norm regulrization, filter norm constraints, and sparse activation regularization.

However, combinations of dropout with other methods can still yield improvements over dropout alone.

## Dropout

Dropout regularization training allows a larger model (more parameters) to be trained.

If training a larger model is not computationally feasible, then there may be no point in using dropout.

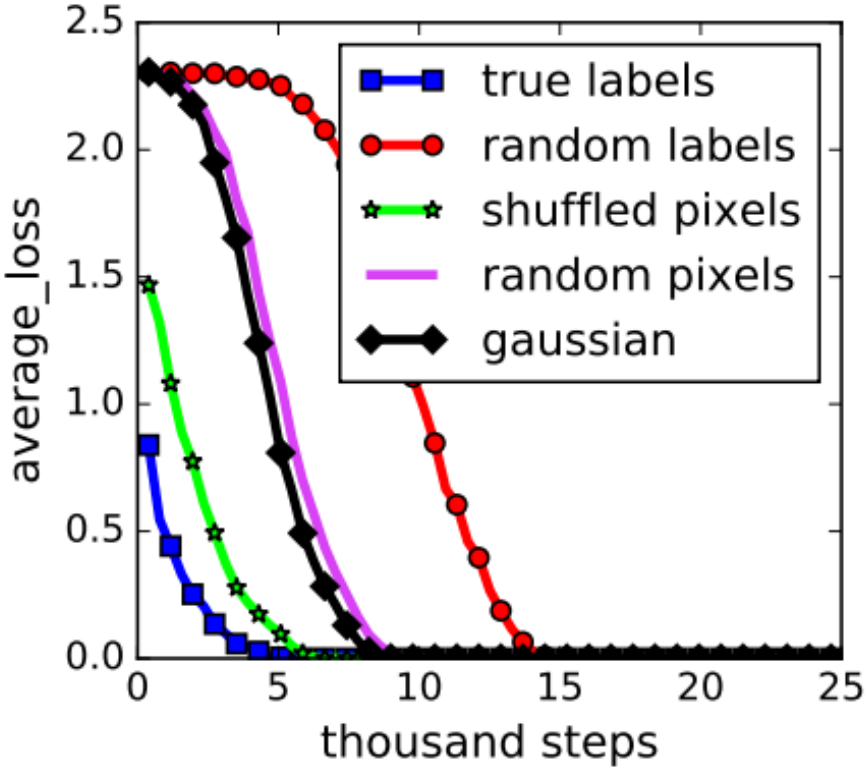
## Implicit Regularization

"Understanding deep learning requires rethinking generalization", Zhang et al. (November 2016).

## Troubling Experiments

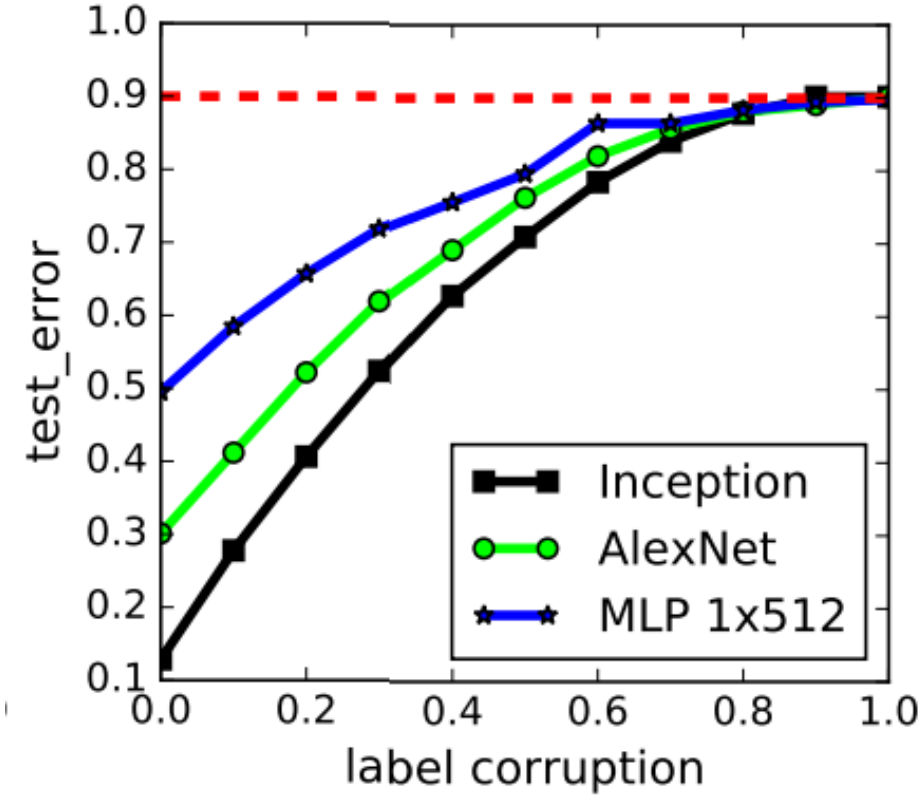
"Our experiments establish that state-of-the-art convolutional networks for image classification trained with stochastic gradient methods easily fit a random labeling of the training data."

# Training on Corrupted Data




Inception on CIFAR10

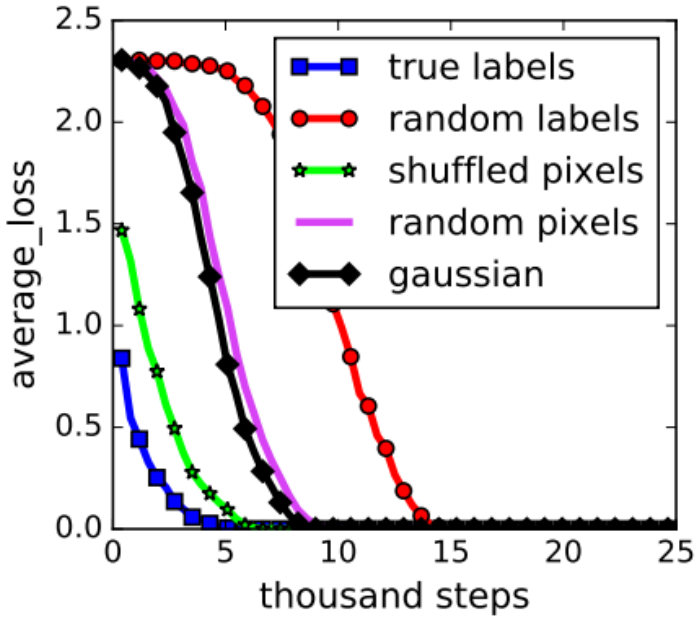
# Test Error as a Function of Training Label Corruption




(c) generalization error growth

# Implicit Regularization

One can modify the PAC-Bayesian bound (and other bounds) to replace $||\Theta||^2$ with $||\Theta - \Theta_{\text{init}}||^2$.



# END