Deep Learning

Abir Das
Assistant Professor
Computer Science and Engineering Department
Indian Institute of Technology Kharagpur

http://cse.iitkgp.ac.in/~adas/

Agenda

- Introduce the concepts of
  - Regularization
  - Dropout
  - Batch normalization

- Resource: Goodfellow Book (Chapter 7)

Regularization

- Machine learning is concerned more about the performance on the test data than on the training data

- According to the Goodfellow book, chapter 7 – “*Many strategies used in Machine Learning are explicitly designed to reduce the test error, possibly at the expense of increased training error. These strategies are collectively known as Regularization*”.

- Also – in the book, regularization is defined as – “*Any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error*”.

Regularization Strategies

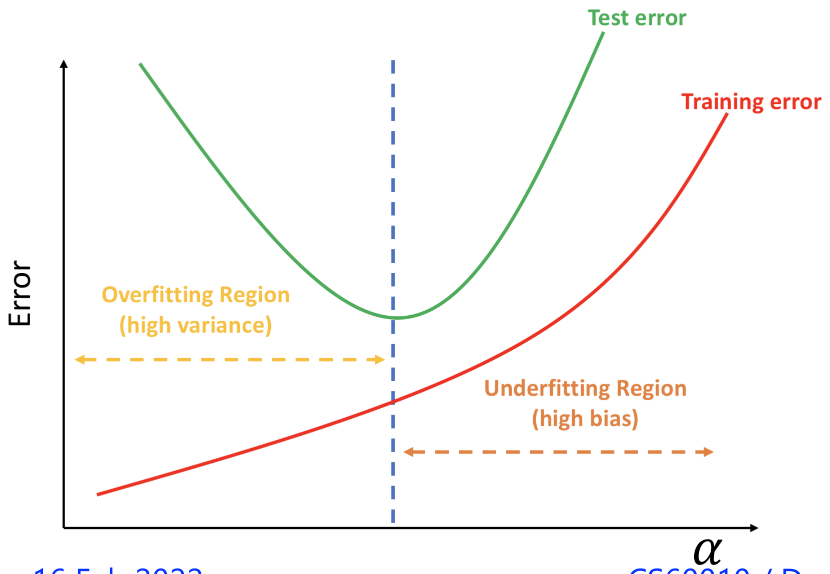
- Adding restrictions on parameter values
- Adding constraints that are designed to encode specific kinds of prior knowledge
- Use of ensemble methods/dropout
- Dataset augmentation
- In practical Deep Learning scenarios, we almost do find – the best fitting model (in the sense of minimizing generalization error) is a large model that has been regularized appropriately

Parameter Norm Penalties

- The most traditional form of regularization to deep learning is adding penalties for high norm of parameters
- This approach limits the capacity of the model by adding penalty $\Omega(\theta)$to the objective function resulting in

$$\tilde{J}(\theta) = J(\theta) + \alpha\Omega(\theta)$$

- When the optimization procedure tries to minimize the objective function, it will also limit the parameters to grow in an unbounded manner, thus restricting the complexity



Parameter Norm Penalties

- Two most common choices are L2 (also known as weight decay in deep learning community) and L1 norms as penalties

$$\Omega(\theta) = \frac{1}{2}||\theta||_2^2 = \frac{1}{2}\theta^T\theta$$
$$\Omega(\theta) = ||\theta||_1 = \sum_i \theta_i$$

- In neural networks, we typically, choose the w's as the θ's to regularize – not the biases

- Regularizing bias parameters can introduce significant amount of underfitting

- Thus for neural networks,

$$\tilde{J}(w) = J(w) + \alpha\Omega(w)$$

L-2 Parameter Norm Regularization

- L-2 parameter norm penalty is commonly known as **Weight Decay**

- We can gain some insight into the behavior of weight decay regularization by studying the gradient of the regularized objective function.

- $\tilde{J}(w) = J(w) + \frac{\alpha}{2} w^T w$
- The gradient is $\nabla_w \tilde{J}(w) = \nabla_w J(w) + \alpha w$

- So, the update step is

$$\begin{aligned} w &\leftarrow w - \epsilon(\nabla_w J(w) + \alpha w) \\ &= (1 - \epsilon\alpha)w - \epsilon \nabla_w J(w) \end{aligned}$$

- The addition of weight decay term modifies the learning rule to shrink the weight vector further before performing the usual gradient update

L-2 Parameter Norm Regularization

- Further simplification of the analysis will be made by making a quadratic approximation to the unregularized objective function in the neighborhood of the optimum weights w^*, to the unregularized objective function.

- $\hat{J}(w) = J(w^*) + \frac{1}{2}(w - w^*)^T H(w - w^*)$
- H is the Hessian Matrix of J w.r.t. w evaluated at w*.

- What rule/formula is used to get this approximation?
- Taylor series expansion

- Where is the first order term?
- w^* being the minimizing value, $\nabla_w J(w^*)$ is 0

L-2 Parameter Norm Regularization

- With this approximation, the regularized objective is given by

$$\hat{J}(w) + \frac{\alpha}{2} w^T w$$

- Computing the gradient of the above and equating it to 0, we get the minimizing w of the regularized and approximated objective as,

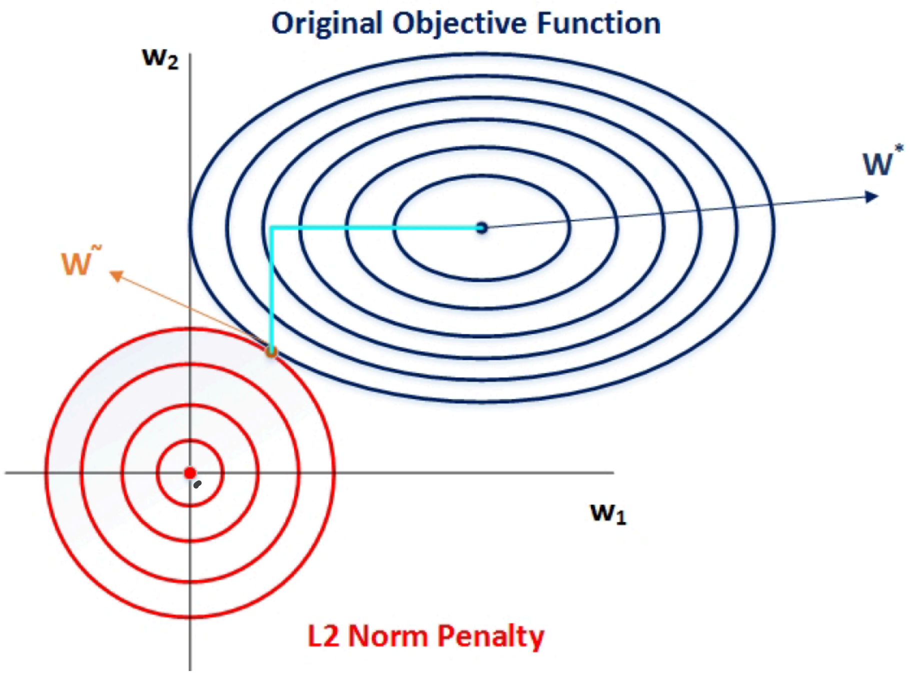
$$\widetilde{w} = (H + \alpha I)^{-1} H w^*$$

- As $\alpha \to 0, \widetilde{w} = w^*$

- As α grows, we can see the effect by using eigendecomposition of H

L-2 Parameter Norm Regularization

- $H = Q\Lambda Q^T$
- Then $\tilde{w} = (Q\Lambda Q^T + \alpha I)^{-1} Q\Lambda Q^T w^* = Q(\Lambda + \alpha I)^{-1}\Lambda Q^T w^*$
- The effect of weight decay is to rescale w^* along the axes defined by the eigenvectors of H. Specifically, the component of w^* that is aligned with the i^{th} eigenvector of H is rescaled by a factor $\frac{\lambda_i}{\lambda_i+\alpha}$



Regularization Strategies: Dataset Augmentation

- One way to get better generalization is to train on more data.
- But under most circumstances, data is limited. Furthermore, labelling is an extremely tedious task.
- Dataset Augmentation provides a cheap and easy way to increase the amount of training data.

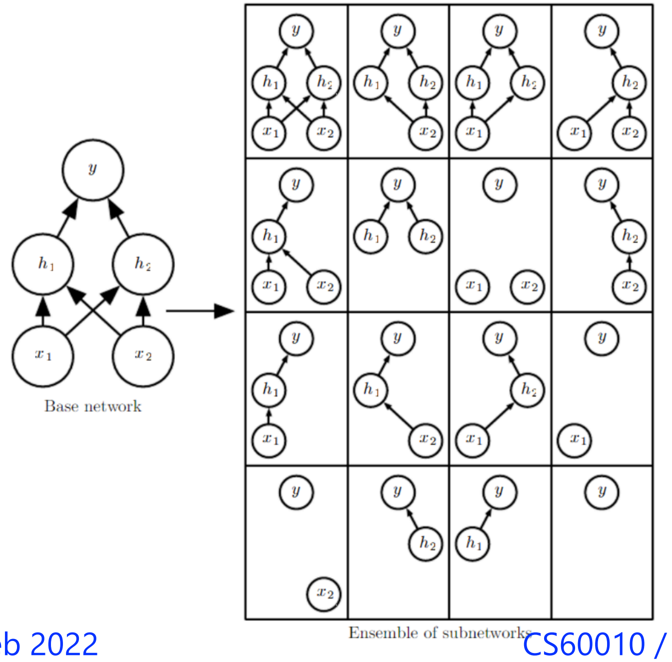
Color Jitter

Horizontal Flip

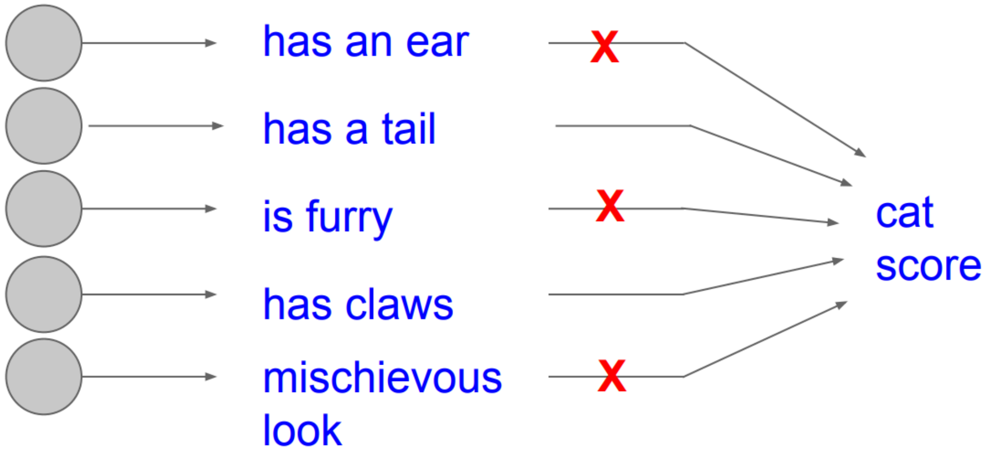
And many many more

Regularization Strategies: Dropout

- Bagging is a technique for reducing generalization error through combining several models (Breiman, 1994)
- Bagging: (1) Train k different models on k different subsets of training data, constructed to have the same number of examples as the original dataset through random sampling from that dataset with replacement
- Bagging: (2) Have all of the models vote on the output for test examples
- Dropout is a computationally inexpensive but powerful extension of Bagging
- Training with dropout consists of training sub-networks that can be formed by removing non-output units from an underlying base network

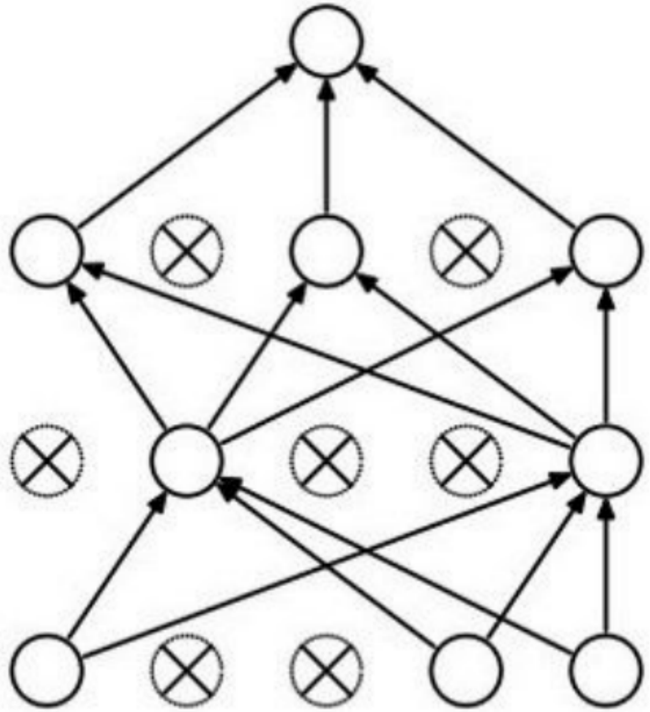


Forces the network to have a redundant representation.



Images courtesy: Goodfellow et. al., Karpathy et. al.

Dropout – At Test Time



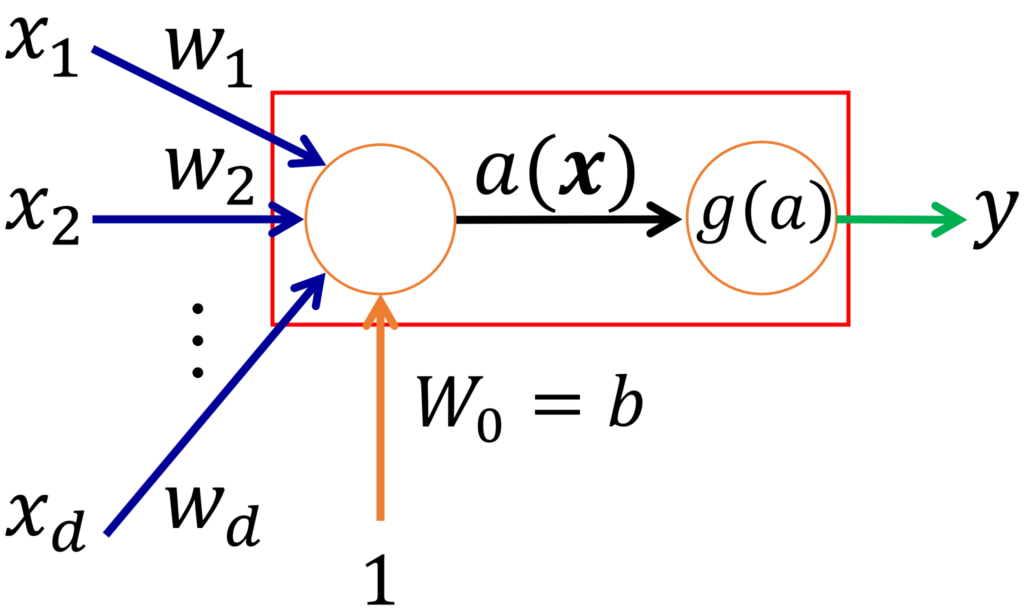
- Ideally, the randomness would have to be integrated out.
- Monte Carlo approximation: Do many forward passes with different random neurons dropped out. Then average out all predictions.
- An approximation to this approximation:
  - Can this be done in a single forward pass!
  - Can this be done without dropping out any neuron during forward pass at test time!
  - 1st way: Get the output of the network at test time with all neurons on. Scale down this by multiplying it with the probability value with which neurons are dropped during training.
  - 2nd way: During training compute the output of the network that you get after dropping out neurons with probability ‘p’. During training itself, scale up this by multiplying it with (1/p). At test time, get the output as what is coming by keeping all the neurons on.

Images courtesy: Goodfellow et. al., Karpathy et. al.

Dropout (Fun Intuition)

Dropout (Srivastava et al., 2014) may be the first instance of a human curated artisanal regularization technique that entered wide scale use in machine learning. Dropout, simply described, is the concept that if you can learn how to do a task repeatedly whilst drunk, you should be able to do the task even better when sober. This insight has resulted in numerous state of the art results and a nascent field dedicated to preventing dropout from being used on neural networks.

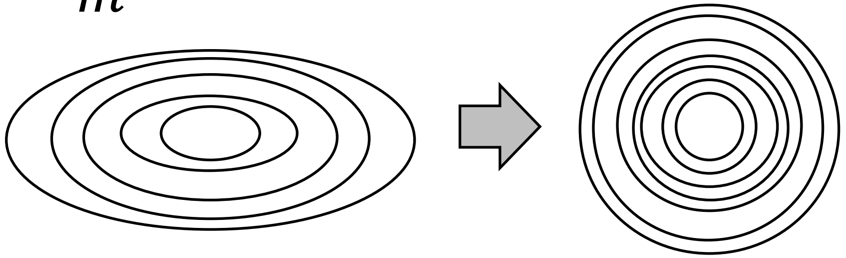
Batch Normalization

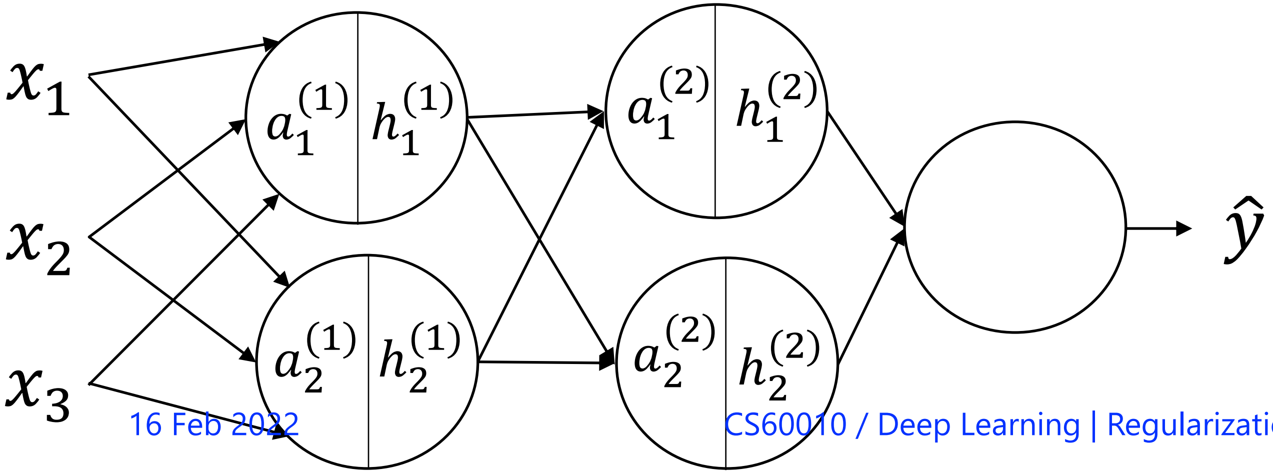


$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

$$X = X - \mu$$

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} x^{(i)^2} \text{ (elementwise)}$$





Can we normalize $h^{(2)}$ so as to train $w^{(3)}$, $b^{(3)}$ faster ?

Normalize $a^{(2)}$

Implementing BatchNorm

Given some intermediate values in NN, $a^{[l](i)} : a^{(1)}, a^{(2)}, \dots, a^{(m)}$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} a^{(i)}$$

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} (a^{(i)} - \mu)^2$$

$$a_{norm}^{(i)} = \frac{a^{(i)} - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

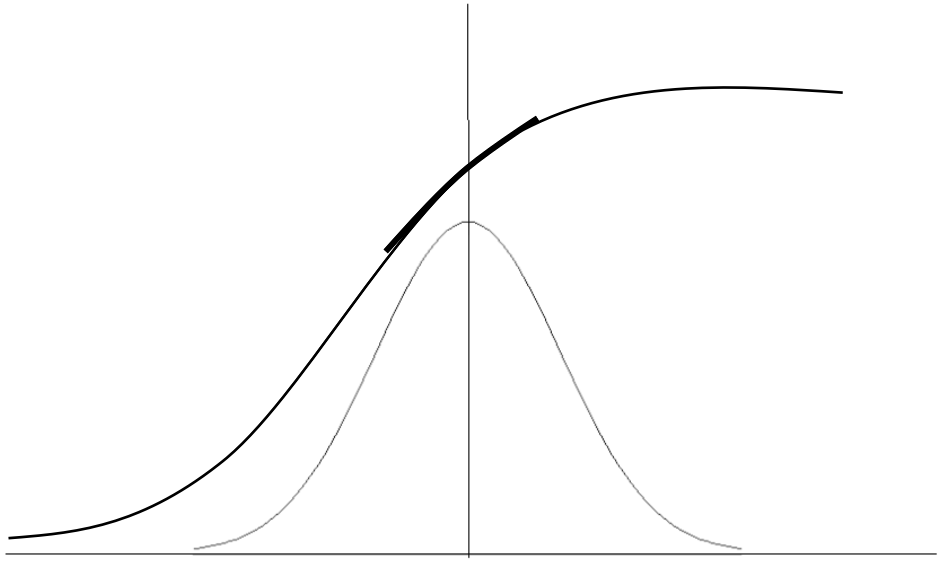
If,

$$\gamma = \sqrt{\sigma^2 + \epsilon}$$

$$\beta = \mu$$

then,

$$\tilde{a}^{(i)} = a^{(i)}$$



$\tilde{a}^{(i)} = \gamma a_{norm}^{(i)} + \beta$, γ and β are the learnable parameters of the model

Use $\tilde{a}^{[l](i)}$ instead of $a^{[l](i)}$ in further calculations

Effect of Batch Normalization on Biases

$$a^{(l)} = w^{(l)} h^{(l-1)} + b^{(l)}$$

We know, $\mu = \frac{1}{m}\sum_{i=1}^{m} a^{(l)}$

$$= \frac{1}{m}\sum_{i=1}^{m} w^{(l)} h^{(l-1)} + \frac{1}{m}\sum_{i=1}^{m} b^{(l)}$$

$$= \frac{1}{m}\sum_{i=1}^{m} w^{(l)} h^{(l-1)} + b^{(l)}$$

So, $a_{norm}^{(i)} = \frac{a^{(i)} - \mu}{\sqrt{\sigma^2 + \epsilon}} = \frac{w^{(l)} h^{(l-1)} + \cancel{b^{(l)}} - \left(\frac{1}{m}\sum_{i=1}^{m} w^{(l)} h^{(l-1)} + \cancel{b^{(l)}}\right)}{\sqrt{\sigma^2 + \epsilon}}$

$$= \frac{w^{(l)} h^{(l-1)} - \frac{1}{m}\sum_{i=1}^{m} w^{(l)} h^{(l-1)}}{\sqrt{\sigma^2 + \epsilon}}$$

$$\tilde{a}^{(i)} = \gamma a_{norm}^{(i)} + \beta$$