



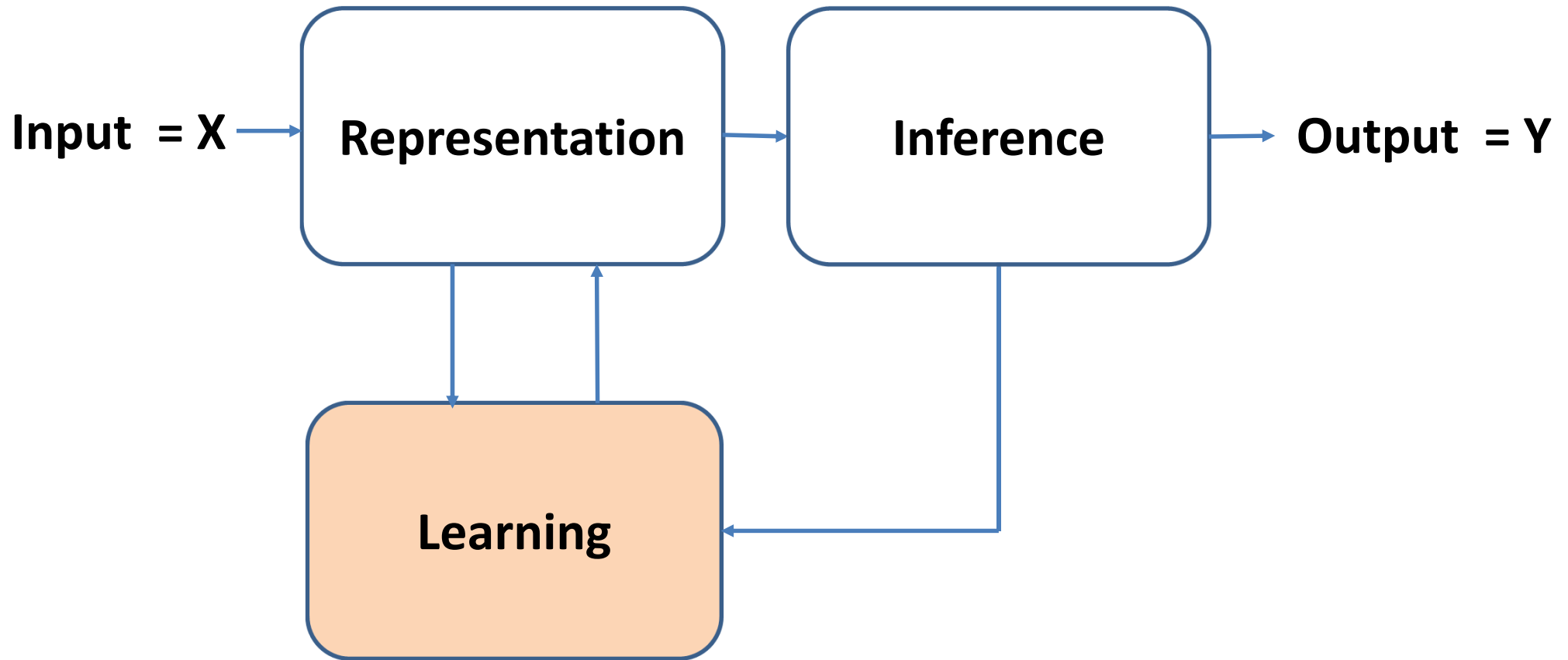
Machine Learning 410

Lesson 2

Regularization for Deep Learning

Steve Elston

Introduction to Regularization for Deep Learning



Introduction to Regularization for Deep Learning

- Deep learning models have very large numbers of parameters which must be learned.
 - Even with large training datasets there may only be a few samples per parameters
- Large number of parameters leads to high chance of **over-fitting** deep learning models
 - Over-fit models do not generalize
 - Over-fit models have poor response to input noise
- To prevent over-fitting we apply **regularization methods**

Introduction to Regularization for Deep Learning

- Bias-variance trade-off
- l_2 regularization
- l_1 regularization
- Early stopping
- Dropout regularization
- Batch normalization

The Bias-Variance Trade-Off

- High capacity models fit training data well
 - Exhibit high variance
 - Do not generalize well; exhibit **brittle behavior**
 - $\text{Error}_{\text{training}} \ll \text{Error}_{\text{test}}$
- Low capacity models have high bias
 - Generalize well
 - Do not fit data well
- Regularization adds bias
 - Strong regularization adds significant bias
 - Weak regularization leads to high variance

The Bias-Variance Trade-Off

- How can we understand the bias-variance trade-off?
- We start with the error:

$$\Delta y = E[Y - \hat{f}(X)]$$

Where:

Y = the label vector.

X = the feature matrix.

$\hat{f}(x)$ = the trained model.

The Bias-Variance Trade-Off

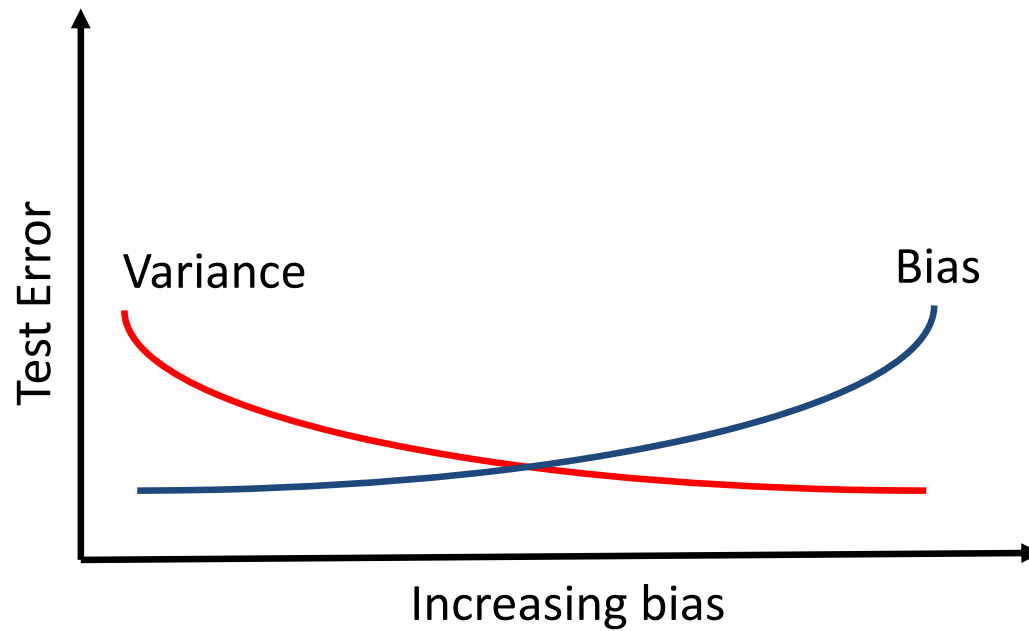
- We can expand the error term

$$\Delta x = \left(E[\hat{f}(X)] - \hat{f}(X)\right)^2 + E\left[(\hat{f}(X) - E[\hat{f}(X)])^2\right] + \sigma^2$$

$$\Delta x = \textit{Bias}^2 + \textit{Variance} + \textit{Irreducible Error}$$

- Increasing bias decreases variance
- Notice that even if the bias and variance are 0 there is still irreducible error

The Bias-Variance Trade-Off



I2 Regularization

- Over-fit models tend to have parameters (weights) with extreme values
- One way to regularize models is to limit the values of the parameters
- We add a small bias term to (greatly) reduce the variance

l2 Regularization

- One way to limit the size of the model parameters is to constrain the **l2** or **Euclidian norm**:

$$\|W\|^2 = \left(w_1^2 + w_2^2 + \dots + w_n^2 \right)^{\frac{1}{2}} = \left(\sum_{i=1}^n w_i^2 \right)^{\frac{1}{2}}$$

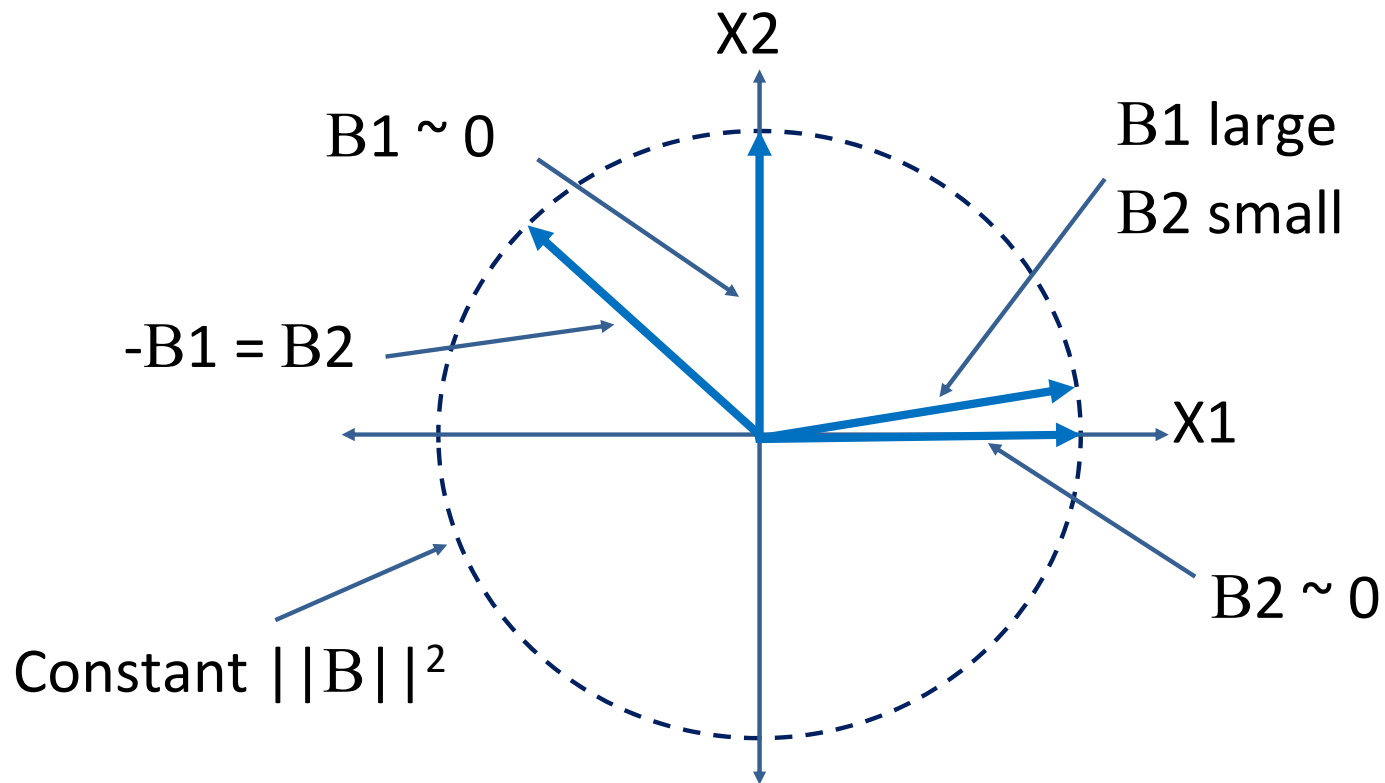
- The regularized loss function is then:

$$J(W) = J_{MLE}(W) + \lambda \|W\|^2$$

- Where λ is the regularization hyperparameter
 - Large λ increases bias but reduces variance
 - Small λ decreases bias and increases variance

L2 Regularization

How can you gain some intuition about L2 regularization?



L2 regularization is considered a **soft constraint**

l2 Regularization

- l2 regularization goes by many names
- Is called Euclidian norm regularization
- First published by Andrey Tikhonov regularization, in late 1940s
 - Only published in English in 1977
 - Is known as **Tikhonov regularization**
- In the statistics literature the method is often called **ridge regression**
- In the engineering literature is referred to as **pre-whitening**

I2 Regularization



Plaque commemorating Andrey Tikhonov at Moscow Institute of Mathematics

Review of Eigenvalues

Eigenvalues are characteristic roots or characteristic values of a linear system

- Start with a square $n \times n$ matrix A , and vector x :

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

- Then, an **eigenvalue** of A has the property:

$$Ax = \lambda x$$

Review of Eigenvalues

Eigenvalues are characteristic roots or characteristic values of a linear system

- Rewrite the eigenvalue relationship:

$$Ax = \lambda x$$

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

- To see that λ is a root of A we can rearrange as follows:

$$\begin{aligned} Ax - \lambda x &= 0 \\ (A - I\lambda)x &= 0 \end{aligned}$$

Review of Eigenvalues

Eigenvalues are characteristic roots or characteristic values of a linear system

- An $n \times n$ matrix A will have **n eigenvalues**
- Find eigenvalues by solving

$$(A - I\lambda)x = 0$$

- Solve n equations involving the **determinant**:

$$\det(A - x) = 0$$

- For more detail on determinants see:

<https://en.wikipedia.org/wiki/Determinant>

Review of Eigenvalues and Eigenvectors

Each eigenvalue of a square matrix is associated with an **eigenvector**

- There is a left and right eigenvector for each eigenvalue
- The **right eigenvector** has the following relationship to the matrix A and eigenvalue λ :

$$Ax_r = \lambda_r x_r$$
$$(A - I\lambda_r)x_r = 0$$

- Notice that the right eigenvector is to the right of the matrix

Review of Eigenvalues and Eigenvectors

Each eigenvalue of a square matrix is associated with an **eigenvector**

- There is also a **left eigenvector** with the following relationship to the matrix A and eigenvalue λ :

$$x_l A = \lambda_l x_l$$

$$(x_l A)^T = \lambda_l x_l^T$$

$$(A^T - I \lambda_l) x_l^T = 0$$

- Notice that the right eigenvector is to the right of the matrix

Review of Eigenvalues and Eigenvectors

Each eigenvalue of a square matrix is associated with an **eigenvector**

- The **l2 norm** of any eigenvector is 1:

$$||x||^2 = (x_1^2 + x_2^2 \dots x_n^2)^{1/2} = 1$$

- This property means that the matrix of n eigenvectors, Q, is **unitary**
- For a **full rank** matrix, each eigenvector is **orthogonal** to every other eigenvector, and **linearly independent**
- Thus, for diagonal matrix of n eigenvalues, Λ , an n x n matrix, A, can be **factored**:

$$A = Q\Lambda Q^{-1}$$

Review of Eigenvalues and Eigenvectors

Each eigenvalue of a square matrix is associated with an **eigenvector**

- Given the eigenvalue-eigenvector **factorization** of the matrix A :

$$A = Q\Lambda Q^{-1}$$

- The **inverse** of A can be found:

$$A^{-1} = Q\Lambda^{-1}Q^{-1}$$

- And, A to the **Nth power** can be computed:

$$A^N = Q\Lambda^N Q^{-1}$$

- You can find more on eigenvalue-eigenvector decomposition:

https://en.wikipedia.org/wiki/Eigendecomposition_of_a_matrix

Eigenvalues and Regularization

ℓ_2 regularization with eigenvalue-eigenvector decomposition

- A linear model with feature matrix, A , labels, x , and parameter vector b is written:

$$x = Ab$$

- We want to find a solution:

$$b = A^{-1}x$$

- The **normal equations** represent a computationally efficient approach:

$$b = (A^T A)^{-1} A^T x$$

Eigenvalues and Regularization

l2 regularization with eigenvalue-eigenvector decomposition

- The n coefficients of the linear model are found with the normal equations:

$$b = (A^T A)^{-1} A^T x$$

- The solution requires finding the inverse of a **symmetric $n \times n$ matrix**
- This matrix can be represented by its eigenvalue-eigenvector decomposition:

$$A^T A = Q \Lambda Q^{-1}$$

Eigenvalues and Regularization

ℓ_2 regularization with eigenvalue-eigenvector decomposition

- We need the inverse of the symmetric matrix with decomposition:

$$A^T A = Q \Lambda Q^{-1}$$
$$(A^T A)^{-1} = Q \Lambda^{-1} Q^{-1}$$

- Where

$$\Lambda^{-1} = \begin{bmatrix} \frac{1}{\lambda_1} & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{\lambda_2} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & \frac{1}{\lambda_n} \end{bmatrix}$$

Eigenvalues and Regularization

l2 regularization with eigenvalue-eigenvector decomposition

- Can compute the inverse of the symmetric matrix with decomposition:

$$(A^T A)^{-1} = Q \Lambda^{-1} Q^{-1}$$

- But, $(A^T A)^{-1}$ can be **rank deficient**!
- In practice, many machine learning problems are rank deficient
- At the eigenvectors of a rank deficient matrix are **not linearly independent**
- A rank deficient matrix has some **zero eigenvalues**
- A matrix with m nonzero eigenvalues is said to have **rank m**

Eigenvalues and Regularization

l2 regularization with eigenvalue-eigenvector decomposition

- For l2 regularization we introduce a small bias term, α^2
- The least-squares, or l2 minimization problem, is then:

$$\min[\| A \cdot x - b \| + \| \alpha^2 \cdot I \|]$$

- With solution

$$b = (A^T A + \alpha^2 \cdot I)^{-1} A^T x$$

Eigenvalues and Regularization

ℓ_2 regularization with eigenvalue-eigenvector decomposition

- For the inverse $(A^T A + \alpha^2 \cdot I)^{-1}$ the eigenvalue matrix is:

$$\Lambda_{Tikhonov}^+ = \begin{bmatrix} \frac{1}{\lambda_1 + \alpha^2} & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{\lambda_2 + \alpha^2} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & \frac{1}{\lambda_m + \alpha^2} \end{bmatrix}$$

- Even for an eigenvalue, λ , of 0, the biased inverse becomes $1/\alpha^2$
- Thus, the bias term, α^2 , creates a 'ridge' of nonzero values on the diagonal ensuring the inverse exists and is stable.

Eigenvalues and Regularization

l2 regularization with eigenvalue-eigenvector decomposition

- Consider the bias-variance trade off of the solution:

$$b = (A^T A + \alpha^2 \cdot I)^{-1} A^T x$$

- For $\alpha^2 = 0$, there is no bias, but variance can be high
- For a large value of α^2 , the inverse is stable and the variance is low, but the bias can be high

Bayesian View of L2 Regularization

How can L2 regularization be interpreted in terms of a prior?

- For data X and label values, Y the posterior distribution of the weights, W is:

$$p(W \mid \{X, Y\}) = \frac{p(W) p(\{X, Y\} \mid W)}{p(\{X, Y\})}$$

- The likelihood of the data and labels given the weights is:

$$p(\{X, Y\} \mid W)$$

- The prior distribution of the weights is $p(W)$

Bayesian View of L2 Regularization

How can L2 regularization be interpreted in terms of a prior?

- For data X and label values, Y the posterior distribution of the weights, W is:

$$p(W \mid \{X, Y\}) = \frac{p(W) p(\{X, Y\} \mid W)}{p(\{X, Y\})}$$

- The likelihood of the data and labels given the weights is:

$$p(\{X, Y\} \mid W)$$

- The prior distribution of the weights is $p(W)$

Bayesian View of L2 Regularization

How can L2 regularization be interpreted in terms of a prior?

- Starting from:

$$p(W \mid \{X, Y\}) = \frac{p(W) p(\{X, Y\} \mid W)}{p(\{X, Y\})}$$

- Take logs of both sides to find the maximum a posteriori (MAP) value of W :
- The prior distribution of the weights is $p(W)$

L1 Regularization

- Regularization can be performed with other norms
- The **L1 (min-max) norm** is another common choice
- Conceptually, L1 norm limits the sum of the absolute values of the weights:

$$\|W\|^1 = (|w_1| + |w_2| + \dots + |w_n|) = \left(\sum_{i=1}^n |w_i| \right)^1$$

- The L1 norm is also known as the **Manhattan distance** or **taxi cab distance**, since it is the distance traveled on a grid between two points.

L1 Regularization

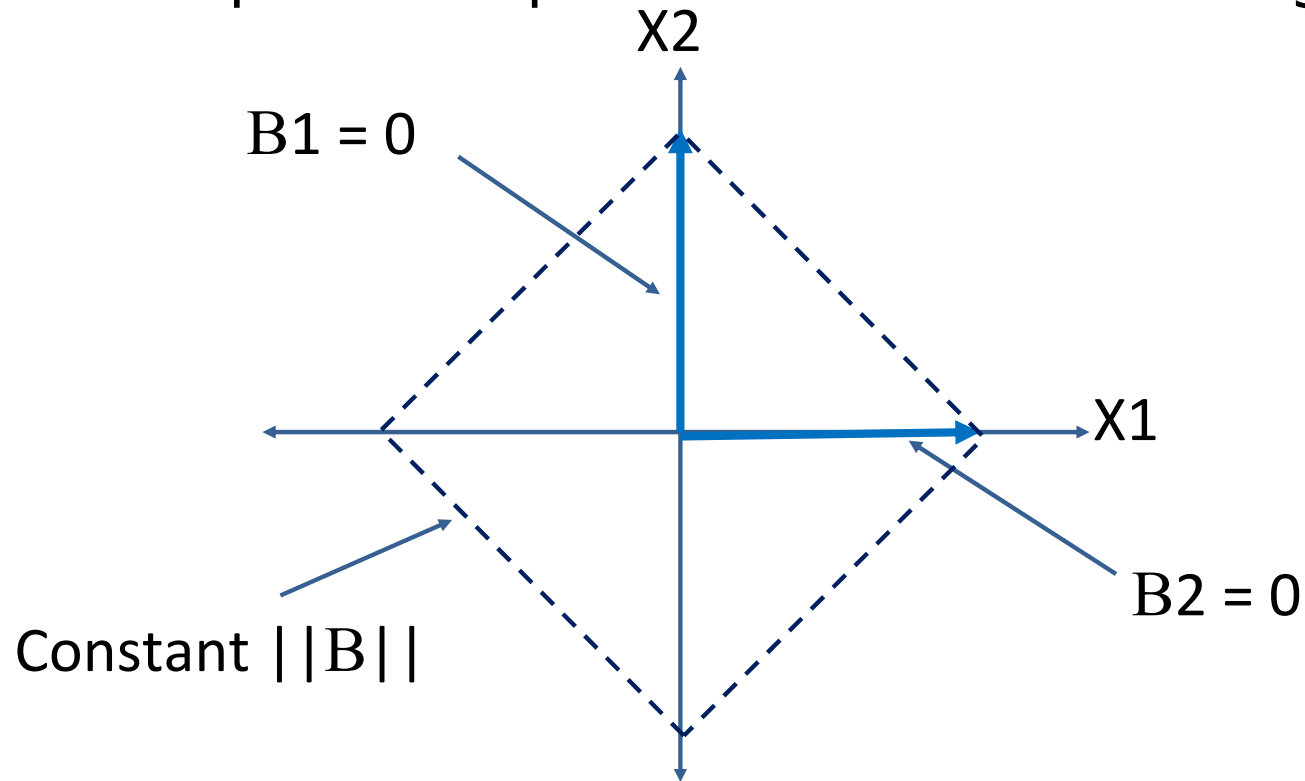
- Given the L1 norm of the weights, the loss function becomes:

$$J(W) = J_{MLE}(W) + \alpha \|W\|^1$$

- Where α is the regularization hyperparameter
 - Large α increases bias but reduces variance
 - Small α decreases bias and increases variance
- The L1 constraint drives some weights to exactly 0
 - This behavior leads to the term **lasso regularization**

L1 Regularization

A diagram helps develop some intuition on L1 regularization:



L1 regularization is a **hard constraint** on the weights

Early Stopping

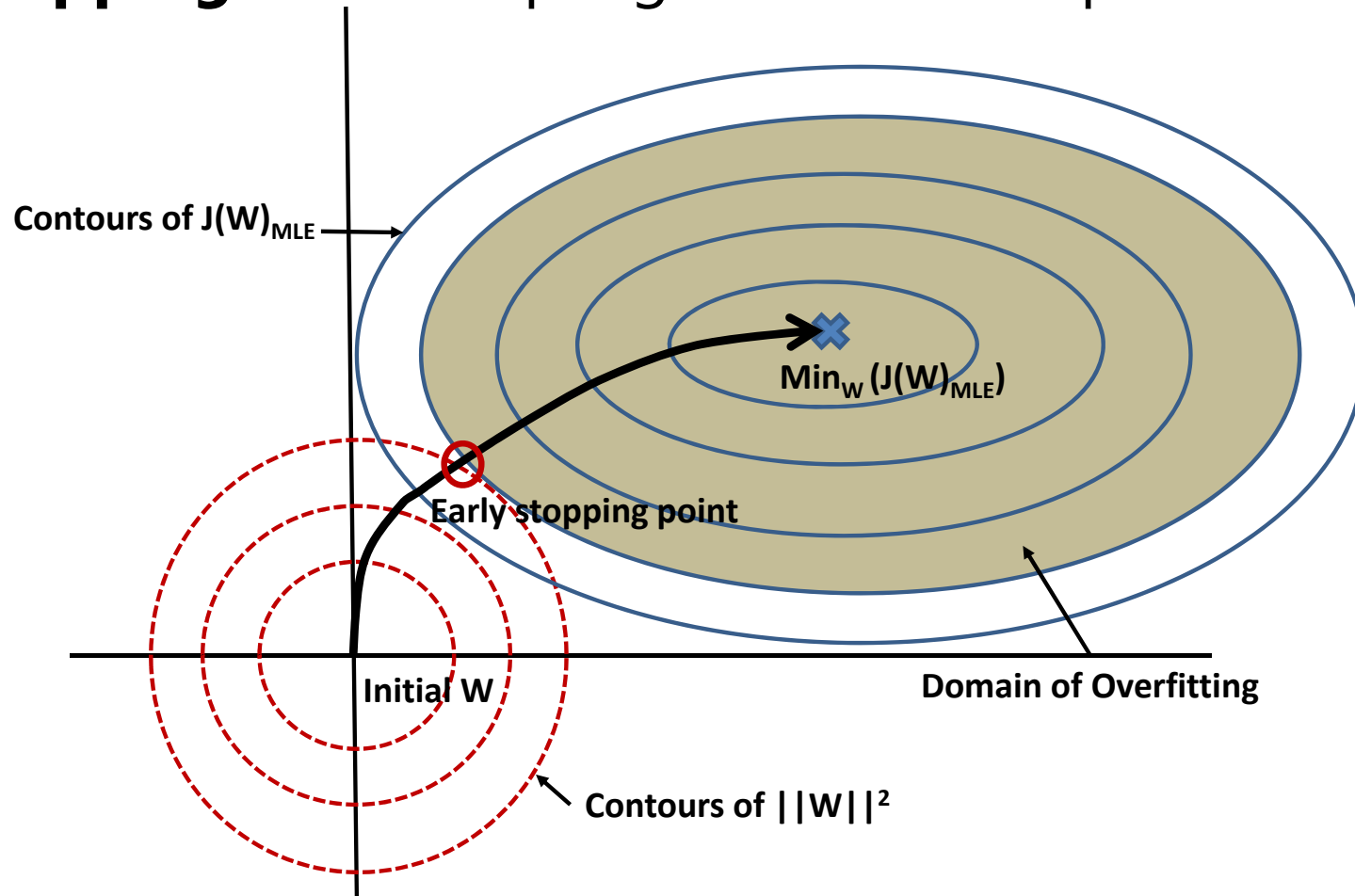
- **Early stopping** is an old and simple idea
- Stop updating the model weights before the model becomes overfit
- Early stopping is analogous to l2 regularization
- We can formulate the regularized loss function as:

$$\operatorname{argmin}_W J(W) = J(W)_{MLE} + \alpha \|W\|^2$$

- Where α is the regularization hyperparameter

Early Stopping

Early stopping has a simple geometric interpretation



Dropout regularization

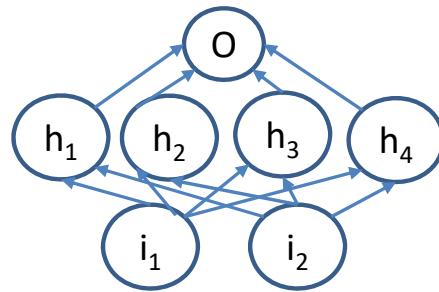
- Overfit deep network models tend to suffer from a problem of co-adaptation
 - With limited training data weight tensors become adapted to the training data
 - Such a model is unlikely to generalize
- We need a way to break the co-adaptation of the weight tensor

Dropout regularization

- **Dropout regularization** is a conceptually simple method unique to deep learning
 - At each step of the gradient decent some **fraction, p , of the weights are dropped-out** of each layer
 - The result is a series of models trained for each dropout sample
 - The final model is a **geometric mean** of the individual models
- Weight values are clipped in a small range as a further regularization
- For full details see the readable paper by Srivastava et. al., 2014
<http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf>

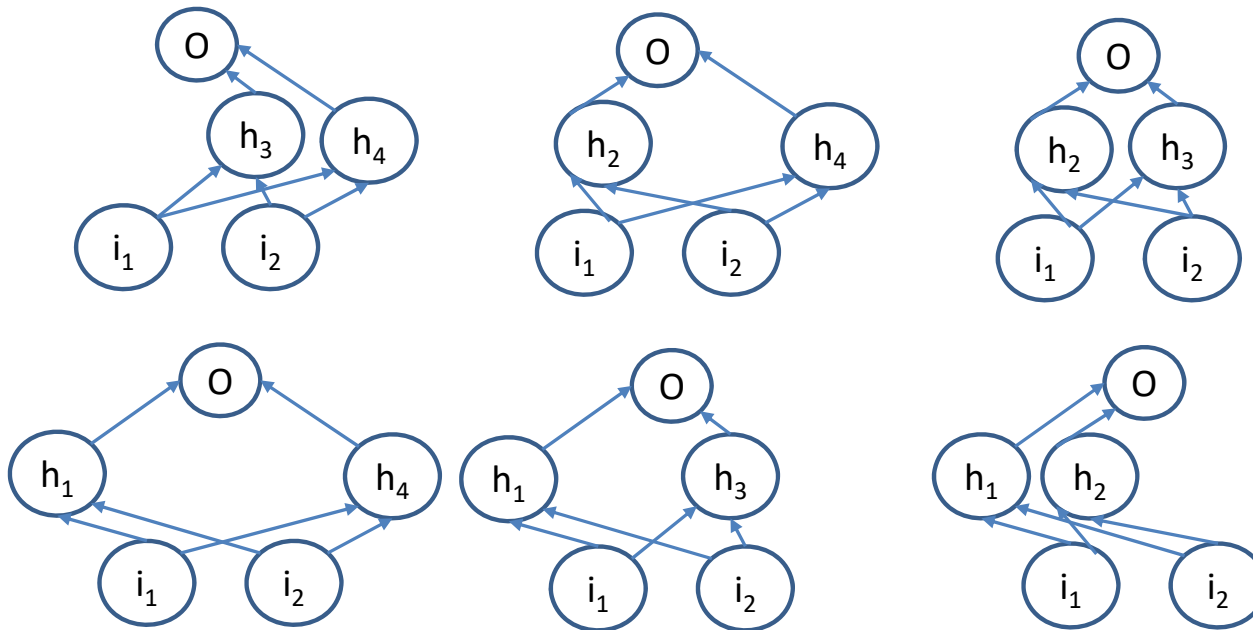
Dropout regularization

- Let's look at a simple example of a network with one hidden layer:



Dropout regularization

- For $p = 0.5$ here are six of the possible samples:



Dropout regularization

- How can the **partial derivatives for backpropagation be computed** with dropout regularization?
- The probability of a weight being in a given model is:

$$r_i^{(l)} \sim \text{Bernoulli}(p)$$

- The forward propagation equations then become:

$$\tilde{h}_i^{(l)} = r_i^{(l)} * y^{(l)}$$

$$z_i^{(l+1)} = w_i^{(l+1)} \cdot \tilde{h}_i^{(l)} + b_i^{(l+1)}$$

$$h_i^{(l+1)} = \sigma(z_i^{(l+1)})$$

Dropout regularization

- How can the **partial derivatives for backpropagation be computed** with dropout regularization?
- We need partial derivatives of

$$h_i^{(l+1)} = \sigma(z_i^{(l+1)})$$

- For the dropout layer σ is linear, and the derivatives with respect to the weights are:
 - $r_i^{(l)} = 1$, in which case the partial derivative = 1
 - $r_i^{(l)} = 0$, in which case the partial derivative = 0

Batch Normalization

- In deep neural networks there is a high chance that units in a hidden layer have a **large range of output values**
 - Causes shifts in the covariance of the output values
 - Leads to difficulty computing the gradient
 - Slows convergence
- A solution is to normalize the output of the hidden layers in the network as a batch
- This simple idea can be really effective
- For more details see Sergey and Szegedy, 2015: <https://arxiv.org/pdf/1502.03167.pdf>