

CPU9444_Fianl2_yann (1)

Autor: Yann

latest: Aug 2024

Reference:

University of Melbourne - COMP30024 Lecture slides

USYD - COMP5329 Lecture slides

UNSW - COMP9414 Lecture slides

UNSW - COMP9814 Lecture slides

Regularisation

Initialisation

Xavier Initialisation

He Initialization

Regularisation

Early Stopping

Hard VS Soft Constraint

Weight Decay

Bayesian Regulation

Data Argumentation

Dropout

DropConnect

Batch Normalization

Layer Normalization

Group Norm

Probability

Probability Space (概率空间)

Sample Space (样本空间)

Sigma-Algebra (σ -代数)

Probability Measure (概率测度)

Random Variable (随机变量)

Discrete Random Variable (离散随机变量)

Continuous Random Variable (连续随机变量)

Probability Distribution (概率分布)

Discrete Probability Distribution (离散概率分布)

Continuous Probability Distribution (连续概率分布)

Types of Probability (概率种类)

Prior Probability (先验概率)

Joint Probability (联合概率)

Conditional Probability (条件概率)

Properties in Probability Theory (概率论中的性质)

Independence (独立性)

Mutual Exclusivity (互斥性)

Law of Total Probability (全概率公式)

Bayes' Theorem (贝叶斯定理)

Exercise

Ex 1

Ex 2

Ex 3

Ex 4

Ex 5

Entropy

纯度 (Purity)

条件熵 (Conditional Entropy)

定义：

Ex 6

Ex 7

Log softmax

Ex 8

Ex 9

Ex 10

Ex 11

Ex 12

Regularisation

Initialisation

- **Zero Initialisation**：如果所有的权重都是零，不论输入是什么，所有神经元的输出都将是相同的，失去了网络的多样性。导致反向传播过程中，所有权重的梯度也相同，所有权重仍然保持相同。每层所有神经元都保持相同的权重，网络的深度优势丧失，每层实际上都在做相同的事情，失去神经元之间的差异。
- **Weight Too Large/Small**：太大的初始化导致梯度爆炸，前向传播过程中输出会变得非常大，反向传播时梯度也会不断增大，导致梯度爆炸，模型无法收敛。太小的初始化导致梯度消失，前向传播过程中输出会变得非常小，反向传播时梯度也会不断减小，导致梯度消失，学习过程极其缓慢。
- **Expected Initialisation**：激活值的均值应为零，保持激活值的均值为零有助于网络的高效学习，使权重更新在正负方向上保持对称，避免激活函数的饱和区域，防止梯度消失问题。
激活值的方差在每层中应保持相同：保持每层激活值的方差一致，确保信息在前向传播过程中不丢失，防止梯度信号爆炸或消失。

Xavier Initialisation

Xavier初始化（也称为Glorot初始化）使得每层的输入和输出具有相同的方差，从而保证前向传播和反向传播过程中的信号稳定。Xavier初始化假设使用的激活函数是tanh，并且我们希望初始

化权重使得每层的激活值的方差保持一致。对于第 l 层，前向传播的公式为：

$$z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$$

$$a^{[l]} = \tanh(z^{[l]})$$

其中， $W^{[l]}$ 是第 l 层的权重矩阵， $a^{[l-1]}$ 是前一层的激活值， $b^{[l]}$ 是偏置。我们希望初始化权重 W 使得激活值 $a^{[l]}$ 的方差与前一层 $a^{[l-1]}$ 的方差相等，即：

$$\text{Var}(a^{[l-1]}) = \text{Var}(a^{[l]})$$

在训练的早期阶段，我们假设 z 的值较小，因此 $\tanh(z) \approx z$ ，即 $a^{[l]} \approx z^{[l]}$ 。所以，方差为：

$$\text{Var}(a^{[l]}) = \text{Var}(z^{[l]})$$

$$z^{[l]} = W^{[l]}a^{[l-1]}$$

$W_{ij}^{[l]}$ 和 $a_j^{[l-1]}$ 都是均值为0的随机变量，我们有：

$$\text{Var}(z_i^{[l]}) = \text{Var}\left(\sum_{j=1}^{n^{[l-1]}} W_{ij}^{[l]} a_j^{[l-1]}\right)$$

假设 W 和 a 相互独立，独立变量方差的和：

$$\text{Var}\left(\sum_{j=1}^{n^{[l-1]}} W_{ij}^{[l]} a_j^{[l-1]}\right) = \sum_{j=1}^{n^{[l-1]}} \text{Var}(W_{ij}^{[l]} a_j^{[l-1]})$$

独立变量乘积的方差：

$$\text{Var}(W_{ij}^{[l]} a_j^{[l-1]}) = \text{Var}(W_{ij}^{[l]}) \text{Var}(a_j^{[l-1]})$$

总方差：

$$\text{Var}(z_i^{[l]}) = \sum_{j=1}^{n^{[l-1]}} \text{Var}(W_{ij}^{[l]}) \text{Var}(a_j^{[l-1]}) \text{Var}(z_i^{[l]}) = n^{[l-1]} \text{Var}(W_{ij}^{[l]}) \text{Var}(a^{[l-1]})$$

为了使 $\text{Var}(a^{[l]}) = \text{Var}(a^{[l-1]})$ ，我们需要：

$$n^{[l-1]} \text{Var}(W_{ij}^{[l]}) = 1 \text{Var}(W_{ij}^{[l]}) = \frac{1}{n^{[l-1]}}$$

由此，Xavier初始化的权重可以从以下分布中采样：

- 正态分布： $W_{ij}^{[l]} \sim \mathcal{N}(0, \frac{1}{n^{[l-1]}})$
- 均匀分布： $W_{ij}^{[l]} \sim U\left(-\sqrt{\frac{6}{n^{[l-1]}+n^{[l]}}}, \sqrt{\frac{6}{n^{[l-1]}+n^{[l]}}}\right)$

He Initialization

He初始化（由Kaiming He等人提出），假设我们使用的激活函数是ReLU，并且我们希望初始化权重使得每层的激活值的方差保持一致。对于第 l 层，前向传播的公式为： $z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$

$$a^{[l]} = \text{ReLU}(z^{[l]})$$

其中， $W^{[l]}$ 是第 l 层的权重矩阵， $a^{[l-1]}$ 是前一层的激活值， $b^{[l]}$ 是偏置。我们希望初始化权重 W 使得激活值 $a^{[l]}$ 的方差与前一层 $a^{[l-1]}$ 的方差相等，即：

$$\text{Var}(a^{[l-1]}) = \text{Var}(a^{[l]})$$

ReLU激活函数定义为：

$$\text{ReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases}$$

考虑到ReLU的特性，假设输入 z 的值为正负均匀分布，那么激活值 a 的方差将是 z 方差的一半：

$$\text{Var}(a^{[l]}) = \frac{1}{2}\text{Var}(z^{[l]})$$

计算 $z^{[l]}$ 的方差：

$$z^{[l]} = W^{[l]}a^{[l-1]}$$

$W_{ij}^{[l]}$ 和 $a_j^{[l-1]}$ 都是均值为0的随机变量，我们有：

$$\text{Var}(z_i^{[l]}) = \text{Var}\left(\sum_{j=1}^{n^{[l-1]}} W_{ij}^{[l]} a_j^{[l-1]}\right)$$

假设 W 和 a 相互独立，独立变量方差的和：

$$\text{Var}\left(\sum_{j=1}^{n^{[l-1]}} W_{ij}^{[l]} a_j^{[l-1]}\right) = \sum_{j=1}^{n^{[l-1]}} \text{Var}(W_{ij}^{[l]} a_j^{[l-1]})$$

独立变量乘积的方差：

$$\text{Var}(W_{ij}^{[l]} a_j^{[l-1]}) = \text{Var}(W_{ij}^{[l]}) \text{Var}(a_j^{[l-1]})$$

总方差：

$$\text{Var}(z_i^{[l]}) = \sum_{j=1}^{n^{[l-1]}} \text{Var}(W_{ij}^{[l]}) \text{Var}(a_j^{[l-1]}) \text{Var}(z_i^{[l]}) = n^{[l-1]} \text{Var}(W_{ij}^{[l]}) \text{Var}(a^{[l-1]})$$

为了使 $\text{Var}(a^{[l-1]}) = \text{Var}(a^{[l]})$ ，我们需要：

$$\text{Var}(a^{[l]}) = \frac{1}{2} \text{Var}(z^{[l]}) \text{Var}(a^{[l]}) = \frac{1}{2} n^{[l-1]} \text{Var}(W_{ij}^{[l]}) \text{Var}(a^{[l-1]})$$

所以：

$$\text{Var}(a^{[l-1]}) = \frac{1}{2} n^{[l-1]} \text{Var}(W_{ij}^{[l]}) \text{Var}(a^{[l-1]}) \frac{1}{2} n^{[l-1]} \text{Var}(W_{ij}^{[l]}) = 1 \text{Var}(W_{ij}^{[l]}) = \frac{2}{n^{[l-1]}}$$

由此，Xavier初始化的权重可以从以下分布中采样：

- 正态分布： $W_{ij}^{[l]} \sim \mathcal{N}\left(0, \frac{1}{n^{[l-1]}}\right)$
- 均匀分布： $W_{ij}^{[l]} \sim U\left(-\sqrt{\frac{6}{n^{[l-1]}}}, \sqrt{\frac{6}{n^{[l-1]}}}\right)$

Regularisation

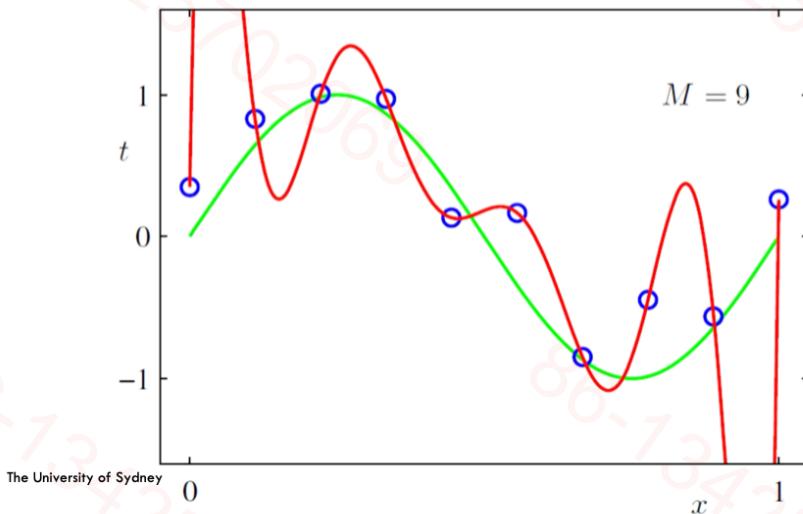
正则化（Regularization）是在机器学习和统计学中用于防止模型过拟合的一种技术。过拟合（Overfitting）是指模型在训练数据上表现得很好，但在新的、未见过的数据上表现较差的现象。过拟合通常发生在模型过于复杂，具有过多的参数，捕捉到了训练数据中的噪声而不是底层数据分布的规律。在过拟合的情况下，模型在训练数据上有很低的误差，但在测试数据上误差很高。这是因为模型已经学习到训练数据中的细节和噪声，而这些细节和噪声在测试数据中并不存在。

In general: any method to prevent overfitting or help the optimization.

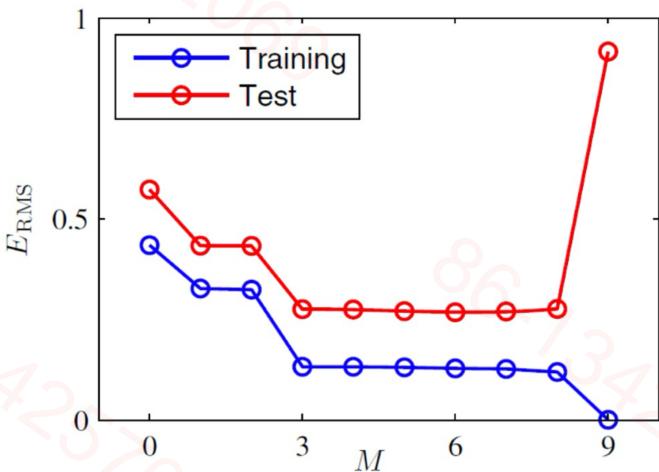
Regression using polynomials,

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots + a_Mx^M + \epsilon$$

$$t = \sin(2\pi x) + \epsilon$$



USYD COMP5329 Lecture slides - Week4



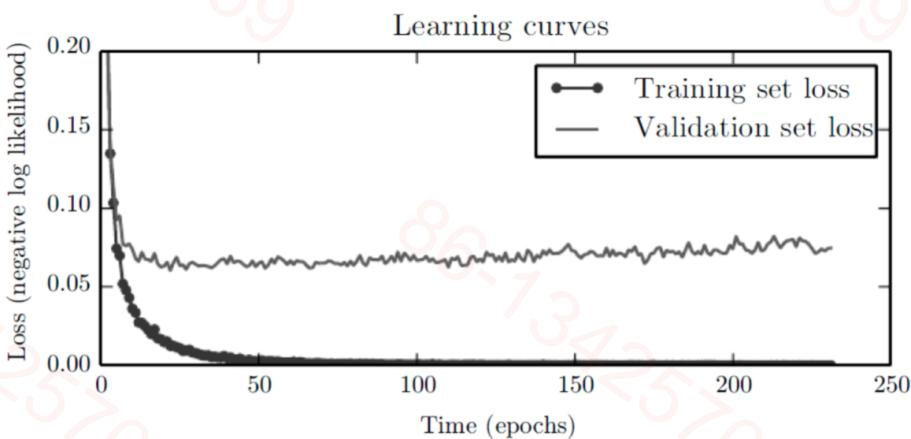
USYD COMP5329 Lecture slides - Week4

正则化通过在模型的损失函数中添加一个额外的项（正则项）来工作，这个正则项惩罚模型的复杂度。简而言之，正则化通过引入一些形式的规则或约束来减少模型的复杂度，从而帮助模型泛化到新数据上。正则化的目的是引入规则来约束模型的学习过程，使其不仅仅适应训练数据，而是学习到数据的一般特征，从而提高模型的泛化能力。正则化通过在损失函数中添加正则项来限制模型的复杂度，从而防止过拟合。正则项对模型的参数施加了惩罚，使得模型倾向于选择较小的参数值，从而减小模型的复杂度。这样，模型就不会过度拟合训练数据中的噪声，而是学习到数据的一般特征，提高在新数据上的泛化能力。

Early Stopping

Early Stopping 通过在训练过程中监控模型在验证集上的性能，并在性能不再改善时提前停止训练，以达到优化模型泛化能力的目的。在训练机器学习模型时，模型的损失函数通常会在训练集上逐渐减少。然而，如果训练时间过长，模型可能会开始对训练数据过拟合，导致在验证集或测试集上的性能下降。Early Stopping 的基本思想是，通过监控模型在验证集上的性能，在性能开始恶化时停止训练，从而避免过拟合。

- Idea: don't train the network to too small training error.



- When training, also output validation error
- Every time validation error improved, store a copy of the weights
- When validation error not improved for some time, stop
- Return the copy of the weights stored

Hard VS Soft Constraint

硬约束是必须严格满足的条件，在整个优化过程中这些条件不可违反。在机器学习的训练目标中，硬约束的优化目标可以表示为：

$$\min_{\theta} \hat{L}(\theta) = \frac{1}{n} \sum_{i=1}^n l(\theta, x_i, y_i) \quad \text{s.t.} \quad R(\theta) \leq r$$

软约束是希望尽量满足的条件，但在某些情况下可以违反。这些约束通过在目标函数中添加惩罚项来实现，惩罚违背约束的程度。在机器学习的训练目标中，软约束的优化目标可以表示为：

$$\min_{\theta} \hat{L}(\theta) = \frac{1}{n} \sum_{i=1}^n l(\theta, x_i, y_i) + \lambda R(\theta)$$

在优化问题中，约束条件通常可以分为硬约束（Hard Constraint）和软约束（Soft Constraint）。硬约束要求在所有情况下都必须严格满足，而软约束则允许一定程度的违反，但会在目标函数中添加惩罚项来减轻这种违反带来的影响。

Weight Decay

权重衰减（Weight Decay）的动机是让参数尽可能小，从而使模型更简单，更不容易过拟合。通过在每次更新参数时减小参数的值，我们实际上是在引导模型找到更平滑、更简单的解。权重衰减的实现非常简单。在每次更新参数时，直接将参数乘以一个小于1的数（如文中的0.98），并且在实践中“just works”，即：

$$\theta \leftarrow \theta \times 0.98$$

Bayesian Regulation

在Bayesian统计中，正则化可以看作是对模型参数的先验信息。这种先验信息可以通过先验分布来表示，不同的先验分布对应不同的正则化方法。L2正则化对应正态分布的先验，L1正则化对应拉普拉斯分布的先验。通过最大化后验概率（MAP），我们可以将正则化引入到优化目标中，从而控制模型的复杂度，提高泛化能力。

在Bayesian框架下，我们的目标是最大化后验概率 $p(\theta|\{x_i, y_i\})$ ，即给定观测数据 $\{x_i, y_i\}$ 后参数 θ 的概率。

$$\max_{\theta} \log p(\theta|\{x_i, y_i\}) = \max_{\theta} \log p(\theta) + \log p(\{x_i, y_i\}|\theta)$$

其中， $p(\theta)$ 是参数 θ 的先验概率，反映了我们对参数的先验知识。 $p(\{x_i, y_i\}|\theta)$ 是似然函数，表示在给定参数 θ 下观测数据的概率。通过对上式进行最大化，我们可以看到它包含两个部分，先验概率 $\log p(\theta)$ 对应于正则化项。似然函数 $\log p(\{x_i, y_i\}|\theta)$ 对应于最大似然估计（MLE）损失。

假设噪声来自正态分布，则参数的先验分布也假设为正态分布 $\theta \sim \mathcal{N}(0, \sigma^2)$ 。那么先验概率为：

$$p(\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\theta^2}{2\sigma^2}\right)$$

取对数后：

$$\log p(\theta) = -\frac{\theta^2}{2\sigma^2} + \text{const}$$

结合MAP目标函数：

$$\max_{\theta} \log p(\theta|\{x_i, y_i\}) = \max_{\theta} \left(\log p(\{x_i, y_i\}|\theta) - \frac{\theta^2}{2\sigma^2} \right)$$

这相当于最小化以下目标函数：

$$\min_{\theta} -\log p(\{x_i, y_i\} | \theta) + \frac{\lambda}{2} \|\theta\|^2$$

其中， $\lambda = \frac{1}{\sigma^2}$ 是正则化参数。这就是L2正则化的形式。

假设噪声来自拉普拉斯分布，则参数的先验分布也假设为拉普拉斯分布 $\theta \sim \text{Laplace}(0, b)$ 。那么先验概率为：

$$p(\theta) = \frac{1}{2b} \exp\left(-\frac{|\theta|}{b}\right)$$

取对数后：

$$\log p(\theta) = -\frac{|\theta|}{b} + \text{const}$$

结合MAP目标函数：

$$\max_{\theta} \log p(\theta | \{x_i, y_i\}) = \max_{\theta} \left(\log p(\{x_i, y_i\} | \theta) - \frac{|\theta|}{b} \right)$$

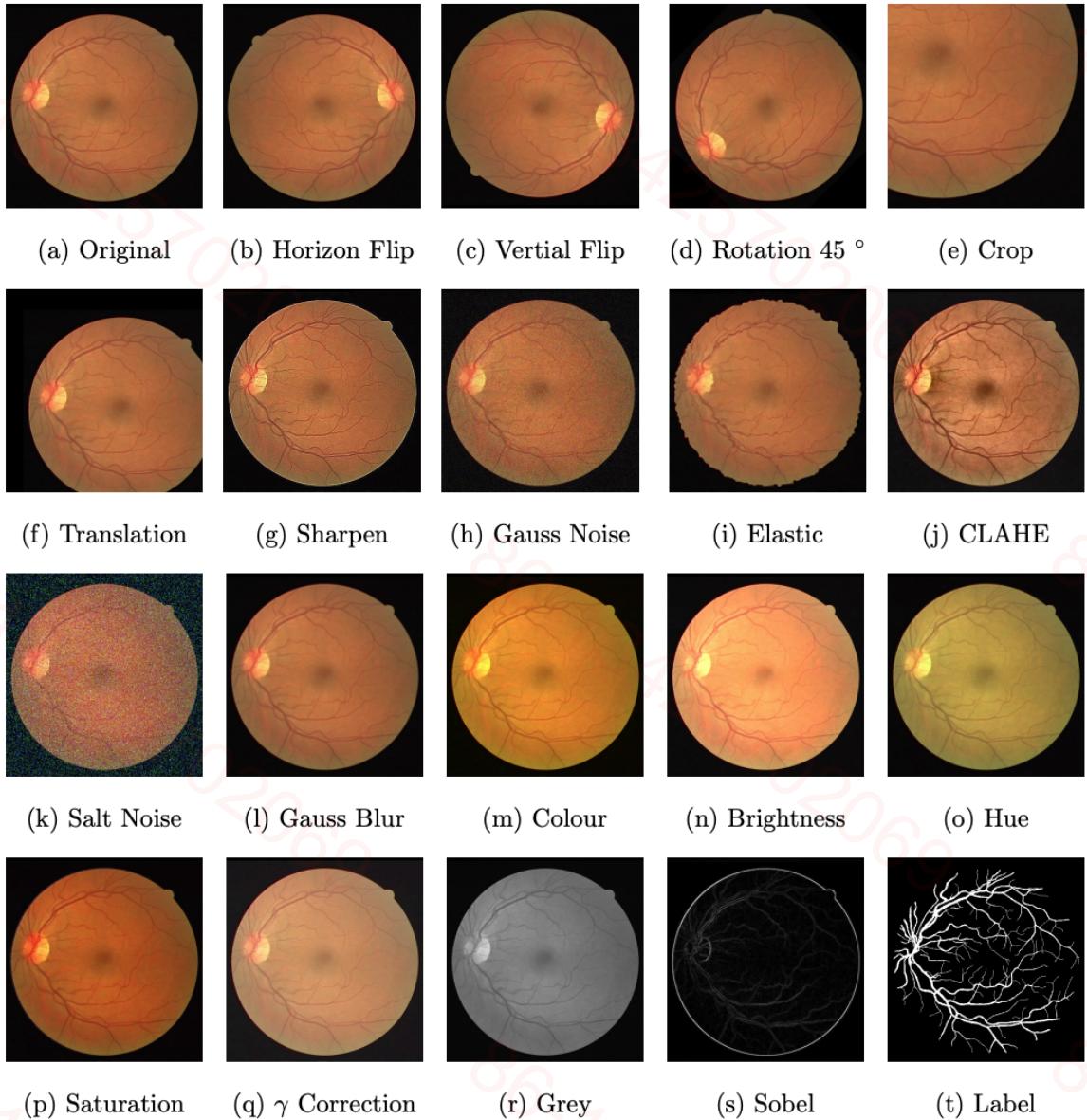
这相当于最小化以下目标函数：

$$\min_{\theta} -\log p(\{x_i, y_i\} | \theta) + \lambda \|\theta\|_1$$

其中， $\lambda = \frac{1}{b}$ 是正则化参数。这就是L1正则化的形式。

Data Argumentation

数据增强是一种通过增加训练数据的多样性来提高模型泛化能力的技术，而无需实际收集新的数据。一种常见的数据增强方法是向输入数据添加噪声。涉及通过噪声扰动输入数据，从而使模型的决策边界更加清晰和鲁棒。当噪声服从高斯分布时，可以证明这相当于在损失函数中添加L2正则项。



Y.Guo <https://arxiv.org/pdf/2311.01023>

假设模型的假设为 $f(x) = w^T x$ ，其中 w 是模型的参数向量， x 是输入数据。设噪声 $\epsilon \sim \mathcal{N}(0, \lambda I)$ ，其中 $\mathcal{N}(0, \lambda I)$ 表示均值为0，协方差为 λI 的高斯分布。加入噪声后的新损失函数为：

$$L(f) = \mathbb{E}_{x,y,\epsilon} [(f(x + \epsilon) - y)^2]$$

将 $f(x + \epsilon) = w^T(x + \epsilon)$ 代入损失函数：

$$L(f) = \mathbb{E}_{x,y,\epsilon} [(w^T x + w^T \epsilon - y)^2]$$

展开得到：

$$L(f) = \mathbb{E}_{x,y} [(w^T x - y)^2] + 2\mathbb{E}_{x,y,\epsilon} [w^T \epsilon (w^T x - y)] + \mathbb{E}_\epsilon [(w^T \epsilon)^2]$$

其中的每一项：

- 第一项是原始损失函数： $\mathbb{E}_{x,y} [(w^T x - y)^2]$
- 第二项包含噪声的期望值： $\mathbb{E}_{x,y,\epsilon} [w^T \epsilon (w^T x - y)] = w^T \mathbb{E}_\epsilon [\epsilon] \mathbb{E}_{x,y} [(w^T x - y)] = 0$
- 第三项为： $\mathbb{E}_\epsilon [(w^T \epsilon)^2] = \mathbb{E}_\epsilon [w^T \epsilon \epsilon^T w] = w^T \mathbb{E}_\epsilon [\epsilon \epsilon^T] w = \mathbb{E}_\epsilon [(w^T \epsilon)^2] = w^T (\lambda I) w = \lambda w^T w$

综合各项，含噪声的损失函数为：

$$L(f) = \mathbb{E}_{x,y} [(w^T x - y)^2] + \lambda w^T w$$

第一项是原始损失函数，第二项是L2正则项。所以通过向输入数据添加高斯噪声，我们实际上在损失函数中添加了L2正则项。这种正则化效果惩罚了较大的权重，从而降低过拟合的风险。这种直观的理解是，决策边界变得更加平滑，更加鲁棒。这一结论与贝叶斯观点一致，其中L2正则化可以看作是对权重施加了高斯先验。

Dropout

Dropout 是一种简单而有效的正则化技术，通过随机遮掉部分神经元，减少神经元之间的依赖性，增强模型的泛化能力。它可以视为一种隐式的模型融合技术，通过训练多个子网络并在测试时结合这些子网络的预测结果来提升模型性能。在实际应用中，适当选择Dropout的保留概率可以显著改善模型的泛化能力。

在每次训练迭代中，对于每一层的每一个神经元，以概率 p 将其暂时从网络中移除（即其输出设置为0）。这样做的目的是为了防止神经元过度依赖其他特定的神经元。对于某个神经元 h ，有 p 的概率将其移除，即 h 的输出变为0。对于保留的神经元（概率为 $1 - p$ ），其输出保持不变。数学表达如下：

$$h'_i = \begin{cases} 0 & \text{with probability } p \\ h_i / (1 - p) & \text{with probability } 1 - p \end{cases}$$

其中 h_i 是第 i 个神经元的输出， h'_i 是应用Dropout后的输出。

在训练过程中，每个神经元以概率

p 被保留，因此每个神经元在训练中的期望输出为 $p \cdot h$ 。为了在测试时保持输出的期望值不变，需要将每个神经元的输出乘以 p ，这样可以保证训练和测试阶段输出的期望值一致。数学表达如下：

$$y^{(l+1)} = f(pW y^{(l)})$$

其中 W 是权重矩阵， $y^{(l)}$ 是第 l 层的输出， f 是激活函数。

- At training (each iteration):

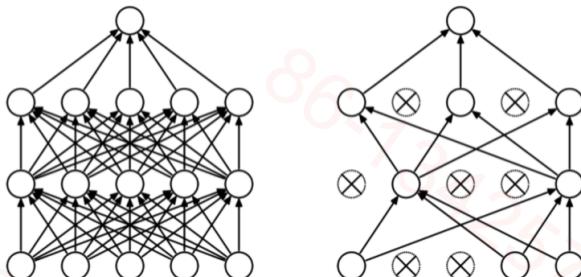
Each unit is retained with a probability p .

Expectation ?

- At test:

The network is used as a whole.

The weights are scaled-down by a factor of p (e.g. 0.5).



The University of Sydney

Page 29

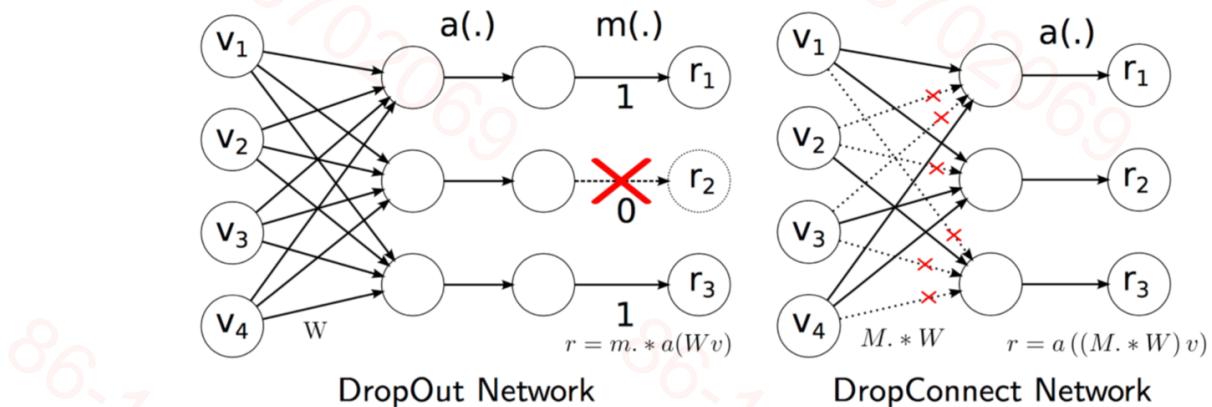
USYD COMP5329 Lecture slides - Week4

Dropout 可以被视为一种模型融合技术。在训练过程中，每次迭代都使用不同的子网络，这相当于训练了 2^n 个不同的神经网络（其中 n 是神经元的数量）。在测试时，通过使用完整的网络并对权重进行缩放，实际上是结合了这些子网络的预测结果，从而增强了模型的鲁棒性和泛化能力。在实际应用中，Dropout 的保留概率 p 通常设置为 0.5，对于输入层，保留概率可以设置得更高一些，例如 0.8。这是因为输入层的特征较少，过多的Dropout可能会丢失太多信息。

DropConnect

DropConnect 是一种与 Dropout 类似的正则化技术，随机扔掉连接而不是神经元，组合更丰富。通过随机将权重设置为零，减少权重之间的依赖性，增强模型的泛化能力。它可以视为一种隐式的模型融合技术，通过训练多个子网络并在测试时结合这些子网络的预测结果来提升模型性能。DropConnect 可以被视为一种模型融合技术。在训练过程中，每次迭代都使用不同的子网络，这相当于训练了 $2^{n \times m}$ 个不同的神经网络（其中 n 是神经元的数量， m 是权重的数量）。在测试时，通过使用完整的网络并对权重进行缩放，实际上是结合了这些子网络的预测结果，从而增强了模型的鲁棒性和泛化能力。

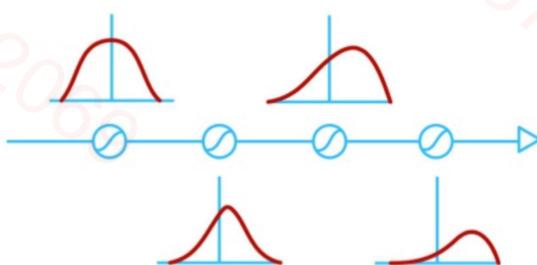
- DropConnect (Yann LeCun et al., 2013) generalizes dropout.
- Randomly drop connections in network with probability $1 - p$.
- As in Dropout, $p = 0.5$ usually gives the best results.



Batch Normalization

Batch Normalization (批量归一化) 计算每个批次的均值和方差，并进行归一化和缩放处理，减少内部协变量偏移 (Internal Covariate Shift) 的影响。为了在测试过程中保持一致性，通常会计算训练过程中的移动平均值和方差，并在测试时使用这些值。

- BN reduces *Covariate Shift*. That is the change in distribution of activation of a component. By using BN, each neuron's activation (sigmoid) becomes (more or less) a Gaussian distribution, i.e. its usually not active, sometimes a bit active, rarely very active.



内部协变量偏移指的是在训练过程中，由于前层参数的更新，导致中间层的输入分布发生变化，从而使训练过程变得复杂和不稳定。BN能够稳定各层输入的分布，减少训练过程中分布的变化，从而减少内部协变量偏移。使得网络可以使用更大的学习率，从而加速训练过程。并且减少对参数初始化的敏感性，增强网络的鲁棒性。

对于每个批次的数据 x ，计算均值 μ 和方差 σ^2 ：

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i$$

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2$$

其中， m 是批次的大小， x_i 是第 i 个样本的输入。

使用计算得到的均值和方差对输入进行归一化：

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

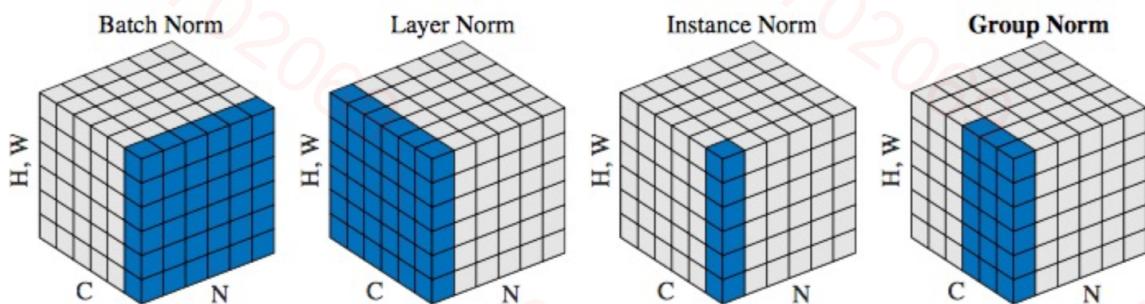
其中， ϵ 是一个小常数，用于避免除以零的情况。

引入可学习的参数 γ 和 β ，对归一化后的数据进行缩放和平移：

$$y_i = \gamma \hat{x}_i + \beta$$

其中， γ 和 β 是与输入相同维度的向量，用于恢复网络的表达能力。

Layer Normalization



USYD COMP5329 Lecture slides - Week4

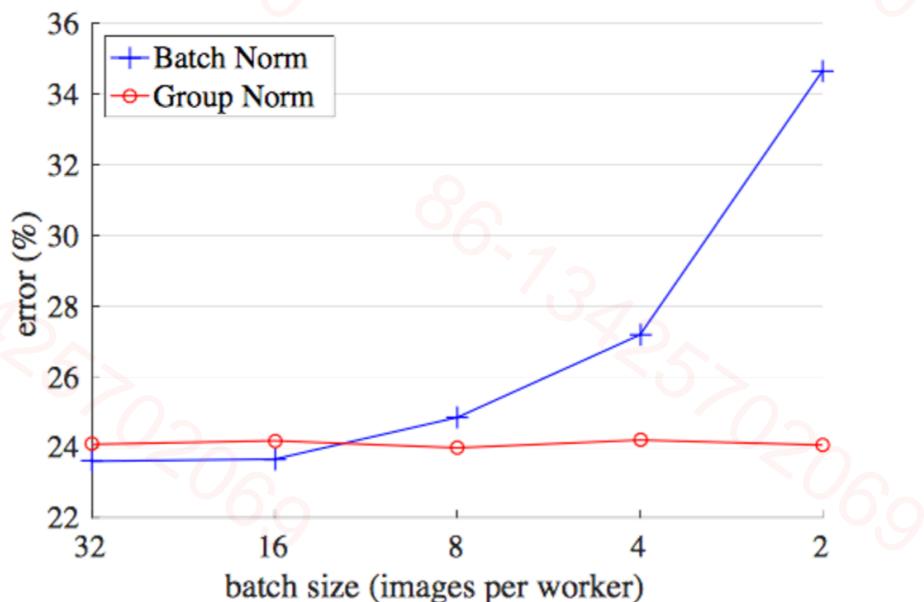
Transformer 这样的时间序列模型通常使用 Layer Normalization 而不是 Batch Normalization，主要有以下几个原因：

- **独立于批量大小**：Batch Normalization 在训练时依赖于批量内的数据分布统计量，这在批量大小较小或不一致时效果不佳。而 Layer Normalization 对每个样本独立进行归一化，不受批量大小的影响，更加稳定。
- **适用于递归和自注意力机制**：时间序列模型（如 RNN 和 Transformer）中的每个时间步的输入可以变化很大。Layer Normalization 对每个时间步的特征进行归一化，可以更好地处理这些变化，而 Batch Normalization 则不太适合这种情况。
- **消除内部协变量偏移**：Layer Normalization 通过对每个样本进行归一化，可以有效消除每个样本内部的协变量偏移，提高训练过程的稳定性和收敛速度。

Group Norm

Group Normalization 通过将通道划分为多个组，并在每组内部进行归一化，成功地消除了对批量大小的依赖，适用于小批量训练的场景。与 Batch Normalization 相比，Group Normalization 提供了一种在小批量情况下依然有效的归一化方法，增强了模型的稳定性和收敛速度。

BN's error increases rapidly when the batch size becomes smaller, caused by inaccurate batch statistics estimation.



ImageNet classification error vs. batch sizes. This is a ResNet-50 model trained in the ImageNet training set using 8 workers (GPUs), evaluated in the validation set.

假设输入为一个四维张量 x ，其形状为 (N, C, H, W) ，分别表示批量大小、通道数、高度和宽度。Group Normalization 将通道 C 划分为 G 组，每组包含 $\frac{C}{G}$ 个通道。对每个组 g 计

算均值 μ_g 和方差 σ_g^2 :

$$\mu_g = \frac{1}{\frac{C}{G} \cdot H \cdot W} \sum_{c \in g} \sum_{h=1}^H \sum_{w=1}^W x_{nchw}$$

$$\sigma_g^2 = \frac{1}{\frac{C}{G} \cdot H \cdot W} \sum_{c \in g} \sum_{h=1}^H \sum_{w=1}^W (x_{nchw} - \mu_g)^2$$

对每个组内的通道进行归一化：

$$\hat{x}_{nchw} = \frac{x_{nchw} - \mu_g}{\sqrt{\sigma_g^2 + \epsilon}}$$

其中

ϵ 是一个小常数，用于避免除以零的情况。

引入可学习的参数

γ 和 β ，对归一化后的数据进行缩放和平移：

$$y_{nchw} = \gamma_c \hat{x}_{nchw} + \beta_c$$

其中 γ 和 β 的维度与通道数 C 相同。

Probability

Probability Space (概率空间)

定义：概率空间 (Ω, \mathcal{F}, P) 被定义为一个三元组：

- Ω 是样本空间。
- \mathcal{F} 是定义在 Ω 上的 σ -代数。
- P 是定义在 \mathcal{F} 上的概率测度。

Sample Space (样本空间)

定义：样本空间 Ω 是所有可能结果的一个非空集合，样本空间表示所有可能的基本事件 (outcomes) 的集合。

例如，在掷硬币的实验中，样本空间可以定义为 $\Omega = \{1, 2, 3, 4, 5, 6\}$ ，其中其中的每一个元素叫做 ω ，例如 $\omega = 1$ ，代表实验的结果。

Sigma-Algebra (σ -代数)

定义： σ -代数 \mathcal{F} 是样本空间的一个子集族，满足特定的性质。这些性质确保我们可以对事件进行联合、交和补操作，并且这些操作的结果仍然是事件。：

1. $\Omega \in \mathcal{F}$ (样本空间本身是 σ -代数中的一个元素)。
2. 如果 $A \in \mathcal{F}$ ，则 $A^c \in \mathcal{F}$ (σ -代数对补集运算封闭)。

例如，在掷骰子的实验中，可能的 σ -代数包括：

$$\mathcal{F} = 2^{|\Omega|} = \{\emptyset, \{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{1, 2\}, \{1, 3\}, \dots, \{1, 2, 3, 4, 5, 6\}\}$$

Probability Measure (概率测度)

定义：概率测度 P 是定义在 σ -代数 \mathcal{F} 上的一个集合函数，用于度量事件发生的可能性，满足以下条件：

1. 非负性：对于任何 $A \in \mathcal{F}$ ，有 $P(A) \geq 0$ 。
2. 正规性： $P(\Omega) = 1$ 。
3. 可数可加性：如果 A_1, A_2, \dots 是两两互不相交的事件，即 $A_i \cap A_j = \emptyset$ (对于 $i \neq j$)，则有：

$$P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(A_i)$$

例如，在掷骰子的实验中，假设骰子是均匀的，则概率测度 P 为：

$$P(A) = \frac{|A|}{6}$$

其中，

$|A|$ 表示事件 A 中元素的数量。

Random Variable (随机变量)

随机变量是概率论中的一个核心概念。尽管名字中有“变量”二字，但随机变量实际上更像一个函数。它是一个从样本空间 Ω 到实数集（或布尔值、整数集等）的映射。

$$X : \Omega \rightarrow \{\text{True}, \text{False}\} \quad \text{or} \quad X : \Omega \rightarrow \mathbb{R}$$

这个映射将每一个可能的事件（或样本点）映射到一个实数（或其他值）。根据其取值性质，随机变量可以分为离散随机变量和连续随机变量：

- **离散随机变量**：取值为有限个或可列无限个值，每个值有相应的概率。
- **连续随机变量**：取值为实数集中的任意值，其概率由概率密度函数表示。

Discrete Random Variable (离散随机变量)

定义：一个随机变量 X 是离散的，如果它的取值集合是有限集或可列无限集，每个取值都有一个相应的概率。

比如，假设样本空间 $\Omega = \{1, 2, 3, 4, 5, 6\}$ ，我们定义一个随机变量 Odd 如下：

- 如果投掷结果是偶数 (2, 4, 6)，则 $\text{Odd}(\text{event}) = \text{False}$

这个随机变量 Odd 将样本空间中的每个元素映射到布尔值 {True, False} 中。如果我们知道 $\text{Odd} = \text{True}$ ，那么我们能够推断出：

- 投掷结果一定是奇数。
- 这个结果是 1、3 或 5 中的一个。
- 不能确定具体是哪一个奇数，除非有其他信息。

这表明通过随机变量的值表示原始样本空间的一些信息，但信息可能是不完全的，所以存在随机性。

Continuous Random Variable (连续随机变量)

定义：连续随机变量是一个从样本空间 Ω 到实数集 \mathbb{R} 的函数 X ，满足对于任何 Borel 集合 $B \subset \mathbb{R}$ ，集合 $\{\omega \in \Omega \mid X(\omega) \in B\}$ 属于 \mathcal{F} ，即：

$$X : \Omega \rightarrow \mathbb{R}$$

$$\{\omega \in \Omega \mid X(\omega) \in B\} \in \mathcal{F}, \forall B \in \mathcal{B}(\mathbb{R})$$

其中， $\mathcal{B}(\mathbb{R})$ 表示实数集 \mathbb{R} 上的 Borel σ -代数。在概率论中，随机变量 (Random Variable) 是一个定义在概率空间上的函数，它将样本空间的元素映射到实数集。随机变量是研究随机现象的核心工具，能够将复杂的随机事件简化为数值分析：

- **映射性质**：随机变量 X 将每个样本空间中的元素（基本事件）映射到实数集中的一个数值。
- **可测性**：对于任意 Borel 集合 B （连续随机变量），集合 $\{\omega \in \Omega \mid X(\omega) \in B\}$ 必须是 \mathcal{F} 的一个事件。这一性质保证了我们可以对随机变量进行概率分析。
- **实数值**：尽管随机变量可以定义为取值在任意集合中的函数，但在实际应用中，我们通常关注实数值随机变量。

Probability Distribution (概率分布)

概率分布是概率论中的一个核心概念，用于描述随机变量取不同值的概率。在统计学中，概率分布可以是离散的或连续的，具体取决于随机变量的类型。概率分布提供了随机变量可能取值的范围以及这些值发生的可能性。

Discrete Probability Distribution (离散概率分布)

定义：离散概率分布描述离散随机变量的概率分布，即随机变量只取有限个或可列无限个离散值。设 X 是一个离散随机变量，其取值集合为 $\{x_1, x_2, \dots\}$ 。离散随机变量 X 的概率分布由概率

质量函数 (Probability Mass Function, PMF) 定义：

$$P(X = x_i) = p_i, \quad i = 1, 2, \dots$$

$$\sum_i p_i = 1$$

每个 p_i 表示随机变量 X 取值 x_i 的概率。

例如，一些经典的离散概率分布：

- **Bernoulli Distribution (伯努利分布)**：设随机变量 X 表示一次伯努利试验的结果，即 X 取值为 0 或 1。其概率质量函数为：

$$P(X = 1) = p, \quad P(X = 0) = 1 - p$$

- **Binomial Distribution (二项分布)**：设随机变量 (X) 表示进行 (n) 次独立伯努利试验中成功的次数。其概率质量函数为：

$$P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}, \quad k = 0, 1, \dots, n$$

Continuous Probability Distribution (连续概率分布)

定义：连续概率分布描述连续随机变量的概率分布，即随机变量取值为连续范围内的任意值。设 X 是一个连续随机变量，其概率分布由概率密度函数 (Probability Density Function, PDF) 定义：

$$f_X(x)$$

其中， $f_X(x)$ 满足以下条件：

1. 非负性：对于所有 x ， $f_X(x) \geq 0$
2. 归一性： $\int_{-\infty}^{\infty} f_X(x) dx = 1$
3. 对于任意区间 $[a, b]$ ， X 落在该区间内的概率为：

$$P(a \leq X \leq b) = \int_a^b f_X(x) dx$$

例如，一些经典的连续概率分布：

- **Uniform Distribution (均匀分布)**：设随机变量 X 在区间 $[a, b]$ 上均匀分布，其概率密度函数为：

$$f_X(x) = \begin{cases} \frac{1}{b-a}, & \text{if } a \leq x \leq b \\ 0, & \text{otherwise} \end{cases}$$

- **Normal Distribution (正态分布)** : 设随机变量 X 服从均值为 μ ，标准差为 σ 的正态分布，其概率密度函数为：

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Types of Probability (概率种类)

概率是衡量某事件发生可能性的一个数值，通常取值在 $[0, 1]$ 之间。概率的三个主要概念是先验概率 (Prior Probability)，联合概率 (Joint Probability) 和条件概率 (Conditional Probability)：

- **Prior Probability (先验概率)** : 在没有额外信息或条件下，对某个事件发生可能性的初步估计。
- **Joint Probability (联合概率)** : 两个或多个事件同时发生的概率。
- **Conditional Probability (条件概率)** : 在已知某一事件发生的情况下，另一个事件发生的概率。

Prior Probability (先验概率)

定义：先验概率 $P(A)$ 是事件 A 在没有考虑任何额外信息或条件下发生的概率，指在考虑任何新证据或数据之前，对某个事件发生可能性的初步估计。它反映了基于现有知识对事件发生的主观信念或客观频率。

例如，假设我们要计算一个袋子里随机抽取一个红球的概率，而袋子里共有10个球，其中有3个红球，那么先验概率 $P(\text{Red})$ 就是：

$$P(\text{Red}) = \frac{3}{10} = 0.3$$

Joint Probability (联合概率)

定义：联合概率 $P(A \cap B)$ 是事件 A 和事件 B 同时发生的概率，指两个或多个事件同时发生的概率。它反映了事件之间的相互关系。

例如，假设我们有两个袋子，袋子1有5个红球和5个蓝球，袋子2有4个红球和6个蓝球。如果我们随机选择一个袋子，然后从中随机抽取一个球，联合概率 $P(\text{Red} \cap \text{Bag1})$ 就是选择袋子1且抽到红球的概率。

Conditional Probability (条件概率)

定义：条件概率 $P(A | B)$ 是在事件 B 已经发生的前提下，事件 A 发生的概率，指在已知某一事件发生的情况下，另一个事件发生的概率。它反映了事件之间的依赖关系。定义为：

$$P(A | B) = \frac{P(A \cap B)}{P(B)}, \quad P(B) \neq 0$$

例如，假设我们知道袋子里抽到的球是红色的，想要计算它来自袋子1的概率，那么条件概率 $P(\text{Bag1} | \text{Red})$ 就是：

$$P(\text{Bag1} | \text{Red}) = \frac{P(\text{Red} \cap \text{Bag1})}{P(\text{Red})}$$

Properties in Probability Theory (概率论中的性质)

概率论中有一些重要的性质和概念，这些性质帮助我们理解和处理随机事件之间的关系。以下是一些关键的性质，包括独立性（Independence）、互斥性（Mutual Exclusivity）、全概率公式（Law of Total Probability）、贝叶斯定理（Bayes' Theorem）等。

- **Independence (独立性)**：两个事件相互独立时，一个事件的发生不影响另一个事件的发生。
- **Mutual Exclusivity (互斥性)**：两个事件互斥时，它们不能同时发生。
- **Law of Total Probability (全概率公式)**：用于将一个事件的概率表示为在若干互斥子事件下的条件概率的加权和。
- **Bayes' Theorem (贝叶斯定理)**：用于通过已知条件概率更新事件概率。

Independence (独立性)

定义：两个事件的独立性是指一个事件的发生不影响另一个事件的发生。也就是说，事件 A 和事件 B 是独立的，当且仅当 A 发生的概率与 B 发生的概率没有关系。事件 A 和事件 B 是独立的，当且仅当：

$$P(A \cap B) = P(A) \cdot P(B)$$

如果有多个事件 A_1, A_2, \dots, A_n ，它们是两两独立的，如果对于所有 $i \neq j$ 都有：

$$P(A_i \cap A_j) = P(A_i) \cdot P(A_j)$$

假设我们有一个公平的六面骰子和一枚公平的硬币。投掷骰子得到 6 和硬币得到正面是独立事件，因为：

$$P(\text{骰子得到 } 6 \cap \text{硬币正面}) = P(\text{骰子得到 } 6) \cdot P(\text{硬币正面}) = \frac{1}{6} \cdot \frac{1}{2} = \frac{1}{12}$$

Mutual Exclusivity (互斥性)

定义：两个事件的互斥性是指它们不能同时发生。也就是说，事件 $\{\text{A}\}$ 和事件 $\{\text{B}\}$ 是互斥的，当且仅当：

$$P(A \cap B) = 0$$

比如，掷一个骰子，得到 2 和得到 5 是互斥事件，因为一个骰子不能同时显示两个不同的面：

$$P(\text{得到 } 2 \cap \text{得到 } 5) = 0$$

Law of Total Probability (全概率公式)

定义：全概率公式用于将一个事件的概率表示为该事件在若干互斥子事件下的条件概率的加权和。设 B_1, B_2, \dots, B_n 是样本空间的一个划分，即这些事件两两互斥，且 $\bigcup_{i=1}^n B_i = \Omega$ 。对于任意事件 A ，有：

$$P(A) = \sum_{i=1}^n P(A | B_i) \cdot P(B_i)$$

假设一个袋子中有红球和蓝球，我们从中随机抽取一个球。袋子中有两种情况：袋子1有3个红球和2个蓝球，袋子2有1个红球和4个蓝球。假设我们随机选择一个袋子，然后从中抽取一个球。我们想计算抽到红球的概率。

- 选择袋子1的概率： $P(B_1) = \frac{1}{2}$
- 选择袋子2的概率： $P(B_2) = \frac{1}{2}$
- 从袋子1中抽到红球的条件概率： $P(A | B_1) = \frac{3}{5}$
- 从袋子2中抽到红球的条件概率： $P(A | B_2) = \frac{1}{5}$

根据全概率公式：

$$P(A) = P(A | B_1) \cdot P(B_1) + P(A | B_2) \cdot P(B_2) = \frac{3}{5} \cdot \frac{1}{2} + \frac{1}{5} \cdot \frac{1}{2} = \frac{2}{5}$$

Bayes' Theorem (贝叶斯定理)

定义：贝叶斯定理提供了一种通过已知条件概率来更新事件概率的方法。设 A 和 B 是两个事件，且 $P(B) \neq 0$ 。贝叶斯定理给出条件概率 $P(A | B)$ 的计算方法：

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)}$$

贝叶斯定理也可以推广到多个事件：

$$P(A_i | B) = \frac{P(B | A_i) \cdot P(A_i)}{\sum_{j=1}^n P(B | A_j) \cdot P(A_j)}$$

Exercise

Ex 1

Which of these statements about Dropout is FALSE:

- a. Dropout simulates an ensemble of network architectures
- b. Dropout helps to prevent overfitting
- c. Dropout encourages redundancy
- d. Dropout encourages the weight values to be small

Ex 2

When training on linearly separable data using the Perceptron Learning Rule, what will happen if both the learning rate and the initial weights are scaled up by a large factor?

- a. The data will be learned successfully, but in a larger number of epochs
- b. The data will be learned successfully, in a smaller number of epochs
- c. The data will be learned successfully, in about the same number of epochs
- d. Learning may become unstable and fail to converge

Ex 3

When using Batch Normalization, in the Testing phase, the Mean and Variance of the activations at each node are typically:

- a. pre-computed from the training set
- b. estimated using running averages
- c. either of the above
- d. none of the above

Ex 4

Which of these is NOT a method for dealing with the problem of vanishing or exploding gradients?

Select one:

- a. Batch Normalization
- b. Rectified Linear Unit
- c. Weight Initialization
- d. Conjugate Gradients

Ex 5

Entropy

熵是一个衡量集合中的不确定性或混乱程度的度量。在决策树中，给定一组样本和它们的类别，熵就用来衡量这组样本相对于其类别的同质性（纯度）。

(Shannon) entropy

- Shannon entropy of a random variable X:

$$H(X) = \sum_{x \in A_x} -p(x) \log_2 p(x)$$

熵的计算公式是：

$$\text{Entropy}(S) = - \sum_{i=1}^n p_i \log_2(p_i)$$

其中

p_i 是第 i 个类别在样本集 S 中出现的概率。

所以，对于我们的这个数据集来说，计算出来的entropy为：

- For our example: weather data - 9 yes and 5 no examples:

$$H(S) = -P_{yes} \log_2 P_{yes} - P_{no} \log_2 P_{no} = I(P_{yes}, P_{no}) = I\left(\frac{9}{14}, \frac{5}{14}\right) = -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14} = 0.940 \text{ bits}$$

- The entropy is measured in bits
- When calculating entropy, we will assume that $\log_2 0 = 0$

纯度 (Purity)

- 高熵 = 低纯度：如果一个数据集的熵值很高，意味着数据集中包含多种类别的样本，纯度较低。
- 低熵 = 高纯度：相反，如果一个数据集的熵值很低，意味着数据集大多数样本属于同一类别，纯度较高。

条件熵 (Conditional Entropy)

定义：

条件熵 $H(X|Y)$ 量化了在已知随机变量 Y 的值的条件下，随机变量 X 的不确定性。其数学定义为：

$$H(X|Y) = \sum_{y \in Y} p(y) H(X|Y=y)$$

这里, $p(y)$ 是 Y 取特定值 y 的概率, 而 $H(X|Y=y)$ 是 $Y = y$ 的条件下 X 的熵。

另一种更直接的表达式是：

$$H(X|Y) = - \sum_{x \in X, y \in Y} p(x,y) \log_2 \left(\frac{p(x|y)}{p(x)} \right)$$

Conditional entropy

- What if we already know something that may pertain to X ? Does this change our surprise/uncertainty?
- Conditional entropy: (average) surprise remaining about sample x of X if we already know the sample y of Y

$$h(x|y) = h(x,y) - h(y)$$

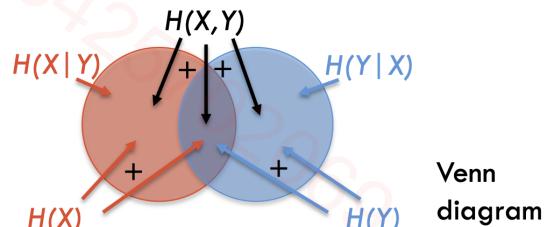
$$h(x|y) = -\log_2 p(x|y)$$

$$H(X|y) = \sum_{x \in A_x} p(x|y) \log_2 \frac{1}{p(x|y)}$$

$$H(X|Y) = \sum_{y \in A_y} p(y) H(X|y)$$

$$H(X|Y) = \sum_{x \in A_x} \sum_{y \in A_y} p(x,y) \log_2 \frac{1}{p(x|y)}$$

$$H(X|Y) = H(X,Y) - H(Y)$$



Properties:

- $0 \leq H(X|Y) \leq H(X)$
- $H(X|Y) = H(X)$ iff X and Y are independent
- $H(X|Y) = 0$ means there is no surprise left in X once we know Y .

Entropy and KL-Divergence (3.13)

The *entropy* of a discrete probability distribution $p = \langle p_1, \dots, p_n \rangle$ is

$$H(p) = \sum_{i=1}^n p_i (-\log_2 p_i)$$

Given two probability distributions $p = \langle p_1, \dots, p_n \rangle$ and $q = \langle q_1, \dots, q_n \rangle$ on the same set Ω , the *Kullback-Leibler Divergence* between p and q is

$$D_{KL}(p \parallel q) = \sum_{i=1}^n p_i (\log_2 p_i - \log_2 q_i)$$

KL-Divergence is like a “distance” from one probability distribution to another.
But, it is not symmetric.

$$D_{KL}(p \parallel q) \neq D_{KL}(q \parallel p)$$

16



Consider a probability distribution p on the same space $\Omega = \{A, B, C, D, E\}$ given by:

$$p = \left\langle \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16} \right\rangle$$

Compute the Entropy of p (give your answer in terms of log base 2 (bits) correct to at least two decimal places):

$$H(p) =$$

Ex 6

Given the probability distributions p and q :

$$p = \left\langle \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16} \right\rangle$$

$$q = \left\langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{2}, \frac{1}{16} \right\rangle$$

Compute the KL-Divergence between p and q (give your answer in terms of log base 2 (bits) correct to at least two decimal places):

$$D_{KL}(p \parallel q) =$$

Ex 7

Log softmax

Softmax (6.2.2)

- classification task with N classes
- neural network with N outputs z_1, \dots, z_N
- assume the network's estimate for the probability of the correct class being j is proportional to $\exp(z_j)$
- because the probabilities must add up to 1, we need to *normalize* by dividing by their sum:

$$\begin{aligned}\text{Prob}(i) &= \frac{\exp(z_i)}{\sum_{j=1}^N \exp(z_j)} \\ \log \text{Prob}(i) &= z_i - \log \sum_{j=1}^N \exp(z_j)\end{aligned}$$

11



Log Softmax and Backprop

If the correct class is k , we can treat $-\log \text{Prob}(k)$ as our cost function, and the gradient is

$$\frac{d}{dz_i} \log \text{Prob}(k) = \delta_{ik} - \frac{\exp(z_i)}{\sum_{j=1}^N \exp(z_j)} = \delta_{ik} - \text{Prob}(i),$$

where δ_{ik} is the *Kronecker delta*.

This gradient pushes up the correct class $i = k$ in proportion to the difference between its assigned probability and 1, and it pushes down the incorrect classes $i \neq k$ in proportion to the probabilities assigned to them by the network.

12



Consider a neural network trained using **softmax** for a classification task with three classes 1, 2, 3. Suppose a particular input is presented, producing outputs:

$$z_1 = 2.5, z_2 = 1.5, z_3 = 0$$

Assuming the correct class for this input is Class 2, that $\text{Prob}(2)$ is the softmax probability of the network choosing Class 2, and that \log_e is the natural logarithm (\ln), compute the following (correct to at least two decimal places):

- $\frac{d(\log_e \text{Prob}(2))}{dz_1} =$

- $\frac{d(\log_e \text{Prob}(2))}{dz_2} =$
- $\frac{d(\log_e \text{Prob}(2))}{dz_3} =$

Ex 8

Base rates of women having breast cancer and having no breast cancer are 0.02% and 99.98% respectively. The true positive rate or sensitivity $P(\text{positive mammography} \mid \text{breast cancer}) = 85\%$ and the true negative or specificity $P(\text{negative mammography} \mid \sim\text{breast cancer}) = 95\%$. Compute $P(C \mid M)$.

Ex 9

Write the formula for Bayes' Rule, in terms of a cause A and an effect B.

Ex 10

In the context of supervised learning, explain the difference between maximum likelihood estimation and Bayesian inference.

Ex 11

Briefly explain the concept of Data Augmentation, and how it has been used in a neural network application of your choosing.

Ex 12

Consider a fully connected feedforward neural network with 6 inputs, 2 hidden units and 4 outputs, using tanh activation at the hidden units and sigmoid at the outputs. Suppose this network is trained on the following data, and that the training is successful.

Item	Inputs	Outputs
1	100000	0001
2	010000	0011
3	001000	0100
4	000100	1010
5	000010	1011
6	000001	1110