# 学习笔记-3: Task 3

# 7 - 过拟合、欠拟合及其解决方案

## 训练误差和泛化误差

- 训练误差(training error): 模型在训练数据集上表现出的误差
- 泛化误差(generalization error): 模型在任意一个测试数据样本上表现出的误差的期望, 并常常通过测试数据集上的误差来近似

## 模型选择

### 验证数据集

测试集只能在所有超参数和模型参数选定后使用一次，不可以使用测试数据选择模型。由于无法从训练误差估计泛化误差，因此也不应只依赖训练数据选择模型。因此, 我们可以预留一部分在训练数据集和测试数据集以外的数据来进行模型选择。 这部分数据被称为验证数据集，简称验证集(validation set)。

### K折交叉验证

由于验证数据集不参与模型训练，当训练数据不够用时，预留大量的验证数据显得太奢侈。一种改善的方法是$K$折交叉验证($K$-fold cross-validation)。在$K$折交叉验证中，我们把原始训练数据集分割成$K$个不重合的子数据集，然后我们做$K$次模型训练和验证。每一次，我们使用一个子数据集验证模型，并使用其他$K-1$个子数据集来训练模型。在这$K$次训练和验证中，每次用来验证模型的子数据集都不同。最后，我们对这$K$次训练误差和验证误差分别求平均。

## 过拟合和欠拟合

- 欠拟合(underfitting): 无法得到较低的训练误差
- 过拟合(overfitting): 模型的训练误差远小于它在测试数据集上的误差

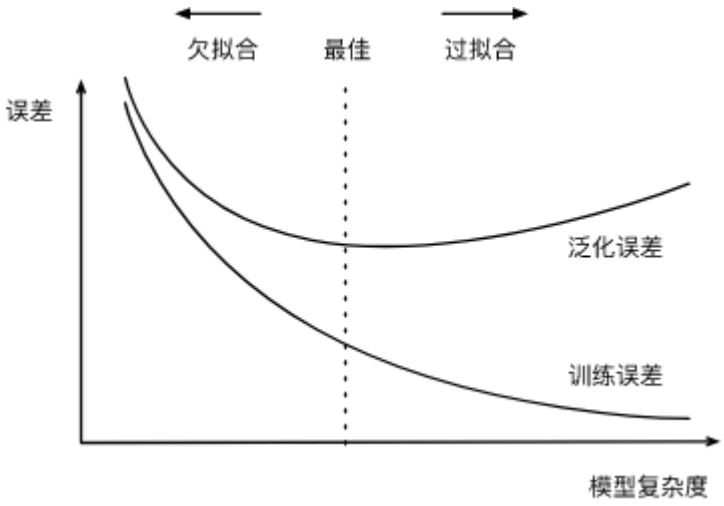
### 模型复杂度

以多项式函数拟合为例，给定一个由标量数据特征$x$和对应的标量标签$y$组成的训练数据集，多项式函数拟合的目标是找一个$K$阶多项式函数

$$\hat{y} = b + \sum_{k=1}^{K} x^k w_k$$

来近似 $y$。在上式中，$w_k$是模型的权重参数，$b$是偏差参数。

给定训练数据集，模型复杂度和误差之间的关系：




**训练数据集大小**

影响欠拟合和过拟合的另一个重要因素是训练数据集的大小。一般来说，如果训练数据集中样本数过少，特别是比模型参数数量更少时，过拟合更容易发生。此外，泛化误差不会随训练数据集里样本数量增加而增大。因此，在计算资源允许的范围之内，我们通常希望训练数据集大一些，特别是在模型复杂度较高时，例如层数较多的深度学习模型。

## 权重衰减

权重衰减等价于 $L_2$ 范数正则化(regularization)。正则化通过为模型损失函数添加惩罚项使学出的模型参数值较小，是应对过拟合的常用手段。 $L_2$范数正则化在模型原损失函数基础上添加$L_2$范数惩罚项，从而得到训练所需要最小化的函数。$L_2$范数惩罚项指的是模型权重参数每个元素的平方和与一个正的常数的乘积。以线性回归中的线性回归损失函数为例

$$\ell(w_1, w_2, b) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\left(x_1^{(i)} w_1 + x_2^{(i)} w_2 + b - y^{(i)}\right)^2$$

其中$w_1, w_2$是权重参数，$b$是偏差参数，样本$i$的输入为$x_1^{(i)}, x_2^{(i)}$，标签为$y^{(i)}$，样本数为$n$。将权重参数用向量 $\boldsymbol{w} = [w_1, w_2]$ 表示，带有$L_2$范数惩罚项的新损失函数为

$$\ell(w_1, w_2, b) + \frac{\lambda}{2n}|\boldsymbol{w}|^2,$$

其中超参数$\lambda > 0$。当权重参数均为0时，惩罚项最小。当$\lambda$较大时，惩罚项在损失函数中的比重较大，这通常会使学到的权重参数的元素较接近0。
有了$L_2$范数惩罚项后，在小批量随机梯度下降中，权重$w_1$和$w_2$的迭代方式为

$$w_1 \leftarrow \left(1 - \frac{\eta\lambda}{|\mathcal{B}|}\right) w_1 - \frac{\eta}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} x_1^{(i)}\left(x_1^{(i)} w_1 + x_2^{(i)} w_2 + b - y^{(i)}\right),$$

$$w_2 \leftarrow \left(1 - \frac{\eta\lambda}{|\mathcal{B}|}\right) w_2 - \frac{\eta}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} x_2^{(i)}\left(x_1^{(i)} w_1 + x_2^{(i)} w_2 + b - y^{(i)}\right).$$

可见，$L_2$范数正则化令权重$w_1$和$w_2$先自乘小于1的数，再减去不含惩罚项的梯度。因此，$L_2$范数正则化又叫权重衰减。权重衰减通过惩罚绝对值较大的模型参数为需要学习的模型增加了限制，这可能对过拟合有效。

## 丢弃法

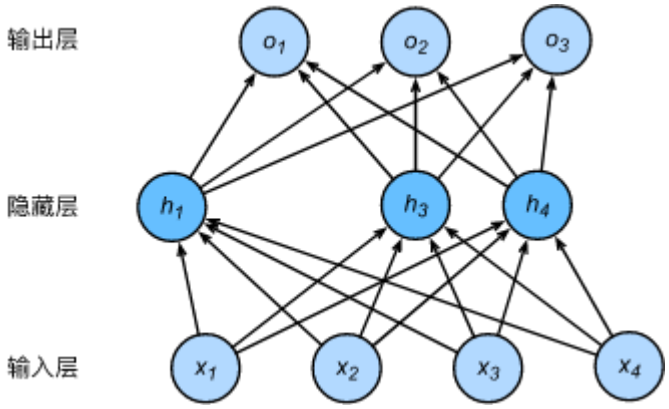
多层感知机中神经网络图含有一个单隐藏层，当对该隐藏层使用丢弃法时，该层的隐藏单元将有一定概率被丢弃掉。设丢弃概率为$p$，那么有$p$的概率$h_i$会被清零，有$1-p$的概率$h_i$会除以$1-p$做拉伸。丢弃概率是丢弃法的超参数。具体来说，设随机变量$\xi_i$为0和1的概率分别为$p$和$1-p$。使用丢弃法时我们计算新的隐藏单元$h_i'$

$$h_i' = \frac{\xi_i}{1-p} h_i$$

由于$E(\xi_i) = 1-p$，因此

$$E(h_i') = \frac{E(\xi_i)}{1-p} h_i = h_i$$

即丢弃法不改变其输入的期望值。对多层感知机的神经网络中的隐藏层使用丢弃法，一种可能的结果如图所示，其中$h_2$和$h_5$被清零。这时输出值的计算不再依赖$h_2$和$h_5$，在反向传播时，与这两个隐藏单元相关的权重的梯度均为0。由于在训练中隐藏层神经元的丢弃是随机的，即$h_1, \ldots, h_5$都有可能被清零，输出层的计算无法过度依赖$h_1, \ldots, h_5$中的任一个，从而在训练模型时起到正则化的作用，并可以用来应对过拟合。



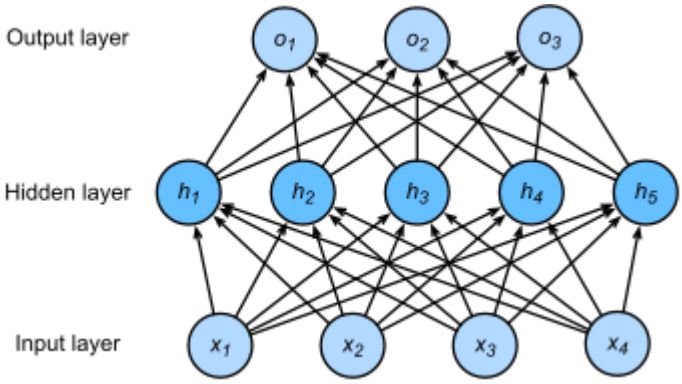

# 8 - 梯度消失、梯度爆炸以及Kaggle房价预测

## 梯度消失和梯度爆炸

深度模型有关数值稳定性的典型问题是消失(vanishing)和爆炸(explosion)。
**当神经网络的层数较多时，模型的数值稳定性容易变差。**

## 随机初始化模型参数

在多层感知机中，如果将每个隐藏单元的参数都初始化为相等的值，那么在正向传播时每个隐藏单元将根据相同的输入计算出相同的值，并传递至输出层。在反向传播中，每个隐藏单元的参数梯度值相等。因此，这些参数在使用基于梯度的优化算法迭代后值依然相等，之后的迭代也是如此。在这种情况下，无论隐藏单元有多少，隐藏层本质上**只有1个**隐藏单元在发挥作用。因此，正如在前面的实验中所做的那样，我们通常将神经网络的模型参数，特别是权重参数，进行随机初始化。




**PyTorch的默认随机初始化**

在线性回归的简洁实现中，`torch.nn.init.normal_()`使模型`net`的权重参数采用正态分布的随机初始化方式。

**Xavier随机初始化**

假设某全连接层的输入个数为$a$，输出个数为$b$，Xavier随机初始化将使该层中权重参数的每个元素都随机采样于均匀分布

$$U\left(-\sqrt{\frac{6}{a+b}},\sqrt{\frac{6}{a+b}}\right)$$

考虑环境因素

协变量偏移

这里我们假设，虽然输入的分布可能随时间而改变，但是标记函数，即条件分布$P(y|x)$不会改变。

标签偏移

认为导致偏移的是标签$P(y)$上的边缘分布的变化，但类条件分布$P(x|y)$是不变的。

概念偏移

另一个相关的问题出现在概念转换中，即标签本身的定义发生变化的情况。

Kaggle房价预测

数据分析项目pipeline：

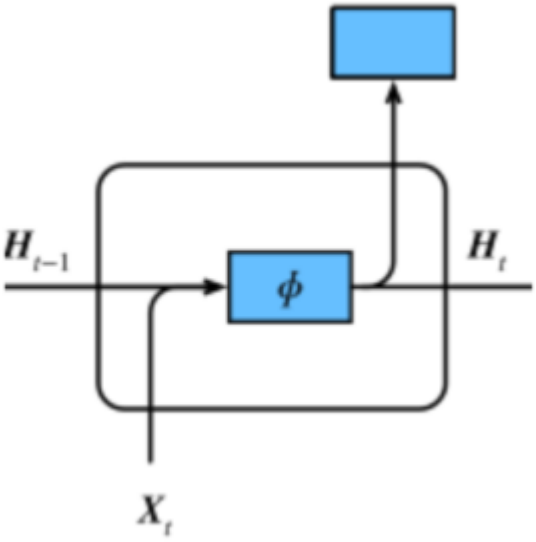
1. 获取数据集
2. 数据预处理：对连续数值的特征做标准化，将离散数值转成指示特征
3. 模型设计
4. 模型验证和模型调整
5. 模型预测

# 9 - 循环神经网络进阶

## GRU (Gate Recurrent Unit)

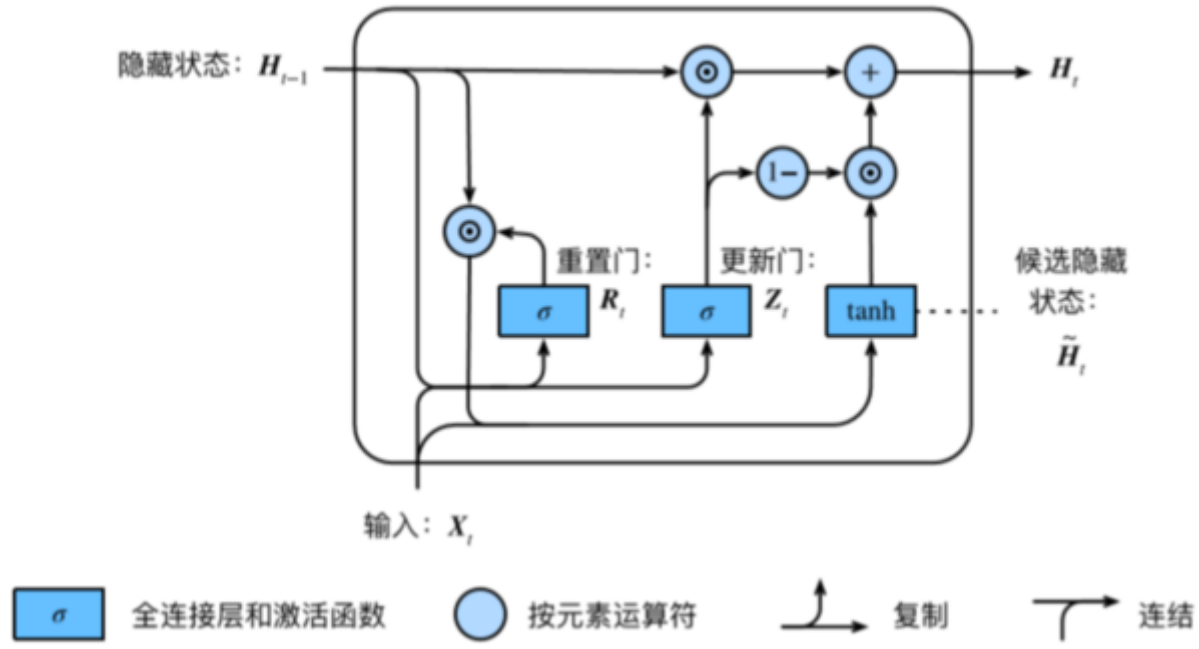
RNN存在的问题：梯度较容易出现衰减或爆炸
门控循环神经网络：捕捉时间序列中时间步距离较大的依赖关系

**RNN**:




$$H_t = \phi(X_t W_{xh} + H_{t-1} W_{hh} + b_h)$$

**GRU**:




$$R_t = \sigma(X_t W_{xr} + H_{t-1} W_{hr} + b_r)$$

$$Z_t = \sigma(X_t W_{xz} + H_{t-1} W_{hz} + b_z)$$

$$\widetilde{H}_t = tanh(X_t W_{xh} + (R_t \odot H_{t-1}) W_{hh} + b_h$$

$$H_t = Z_t \odot H_{t-1} + (1 - Z_t) \odot \widetilde{H}_t$$

- 重置门有助于捕捉时间序列里短期的依赖关系；
- 更新门有助于捕捉时间序列里长期的依赖关系。

## LSTM (long short-term memory)

- 遗忘门: 控制上一时间步的记忆细胞
- 输入门: 控制当前时间步的输入
- 输出门: 控制从记忆细胞到隐藏状态
- 记忆细胞: 一种特殊的隐藏状态的信息的流动

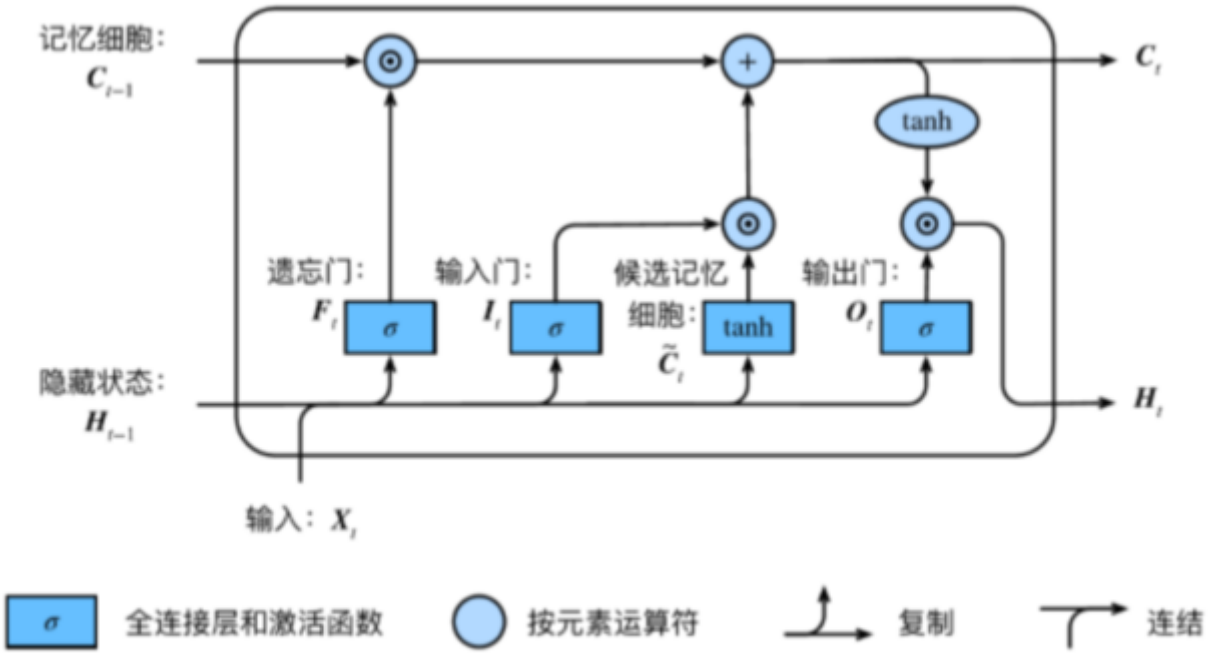



图 6.10: 长短期记忆中隐藏状态的计算。这里的⊙是按元素乘法

$$I_t = \sigma(X_t W_{xi} + H_{t-1} W_{hi} + b_i)$$

$$F_t = \sigma(X_t W_{xf} + H_{t-1} W_{hf} + b_f)$$

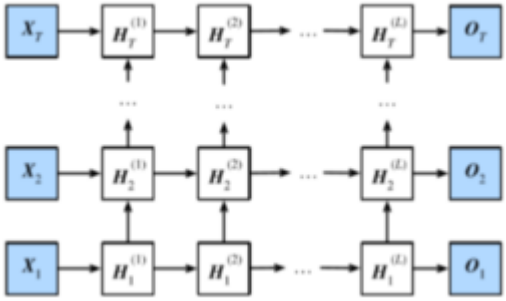
$$O_t = \sigma(X_t W_{xo} + H_{t-1} W_{ho} + b_o)$$

$$\widetilde{C}_t = tanh(X_t W_{xc} + H_{t-1} W_{hc} + b_c)$$

$$C_t = F_t \odot C_{t-1} + I_t \odot \widetilde{C}_t$$

$$H_t = O_t \odot tanh(C_t)$$

深度循环神经网络



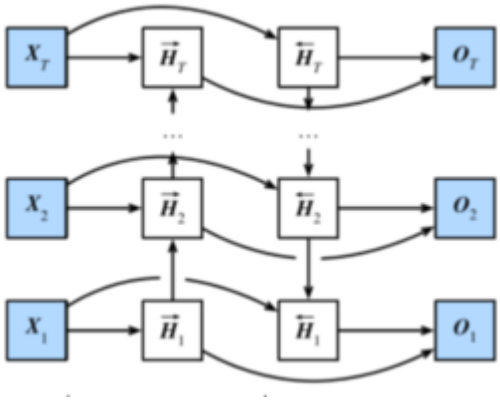
$$\boldsymbol{H}_t^{(1)} = \phi(\boldsymbol{X}_t \boldsymbol{W}_{xh}^{(1)} + \boldsymbol{H}_{t-1}^{(1)} \boldsymbol{W}_{hh}^{(1)} + \boldsymbol{b}_h^{(1)})$$

$$\boldsymbol{H}_t^{(\ell)} = \phi(\boldsymbol{H}_t^{(\ell-1)} \boldsymbol{W}_{xh}^{(\ell)} + \boldsymbol{H}_{t-1}^{(\ell)} \boldsymbol{W}_{hh}^{(\ell)} + \boldsymbol{b}_h^{(\ell)}$$

$$\boldsymbol{O}_t = \boldsymbol{H}_t^{(L)} \boldsymbol{W}_{hq} + \boldsymbol{b}_q$$

深度循环神经网络**不是**越深越好

双向循环神经网络



$$\overrightarrow{\boldsymbol{H}}_t = \phi(\boldsymbol{X}_t \boldsymbol{W}_{xh}^{(f)} + \overrightarrow{\boldsymbol{H}}_{t-1} \boldsymbol{W}_{hh}^{(f)} + \boldsymbol{b}_h^{(f)})$$

$$\overleftarrow{\boldsymbol{H}}_t = \phi(\boldsymbol{X}_t \boldsymbol{W}_{xh}^{(b)} + \overleftarrow{\boldsymbol{H}}_{t+1} \boldsymbol{W}_{hh}^{(b)} + \boldsymbol{b}_h^{(b)})$$

$$\boldsymbol{H}_t = (\overrightarrow{\boldsymbol{H}}_t, \overleftarrow{\boldsymbol{H}}_t)$$

$$\boldsymbol{O}_t = \boldsymbol{H}_t \boldsymbol{W}_{hq} + \boldsymbol{b}_q$$

仍存在的疑问

1. 为什么LSTM/GRU可以解决梯度衰减的问题？
2. 如何理解**重置门**有助于捕捉时间序列里**短期**的依赖关系；**更新门**有助于捕捉时间序列里**长期**的依赖关系？