第 3 章

模 型 基 础

学习目标：

- 熟练掌握常用的传统机器学习模型。
- 熟练掌握常用的深度学习模型。
- 熟练掌握常用的模型评价指标。

3.1 传统机器学习模型

3.1.1 Logistic Regression

Logistic Regression 又称逻辑回归，简记为 LR，是一种广义的线性回归分析模型和机器学习领域极为常用的模型。逻辑回归模型的本质是假设数据服从 Logistic 分布，然后使用极大似然估计计算模型参数。

Logistic 分布是一种连续型的概率分布，其分布函数和密度函数分别为

$$F(x)=P(X\leqslant x)=\frac{1}{1+\mathrm{e}^{-\frac{(x-\mu)}{\gamma}}} \tag{3-1}$$

$$f(x)=F'(X\leqslant x)=\frac{\mathrm{e}^{-\frac{(x-\mu)}{\gamma}}}{\gamma(1+\mathrm{e}^{-\frac{(x-\mu)}{\gamma}})^2} \tag{3-2}$$

其中，μ 表示位置参数，$\gamma>0$ 为尺度参数。其分布函数和密度函数分别如图 3-1 和图 3-2 所示。

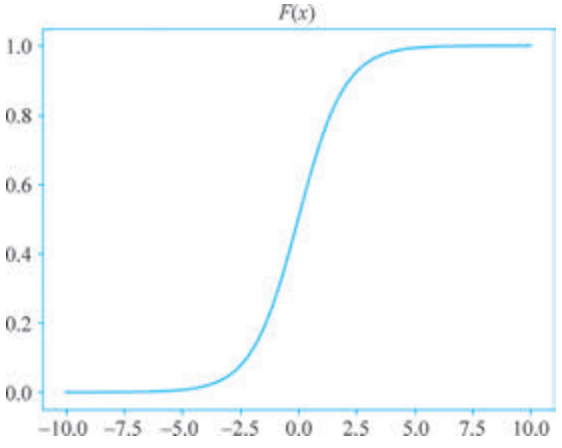

图 3-1 Logistic 分布函数

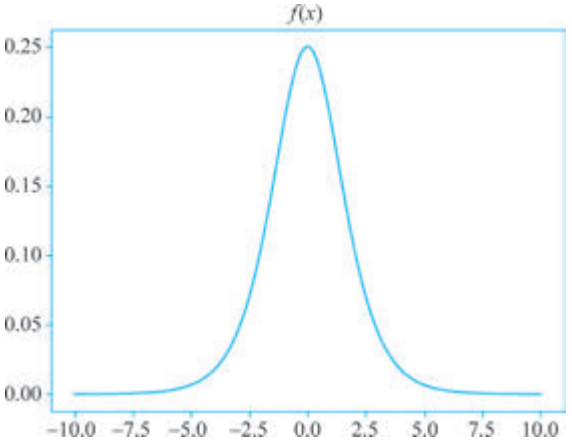

图 3-2 Logistic 密度函数

Logistic 分布是由其位置和尺度参数定义的连续分布。Logistic 分布的形状与正态分布的形状相似，但是 Logistic 分布的尾部更长，所以使用 Logistic 分布来建模比使用正态分布具有更长尾部和更高波峰的数据分布。在深度学习中常用到的 sigmoid 函数就是 Logistic 的分布函数在 $\mu=0,\gamma=1$ 的特殊形式。

逻辑回归主要用于分类问题，接下来以二分类为例，图 3-3 给出了存在一条直线对数据集中所有数据完成线性可分的二分类问题示例。

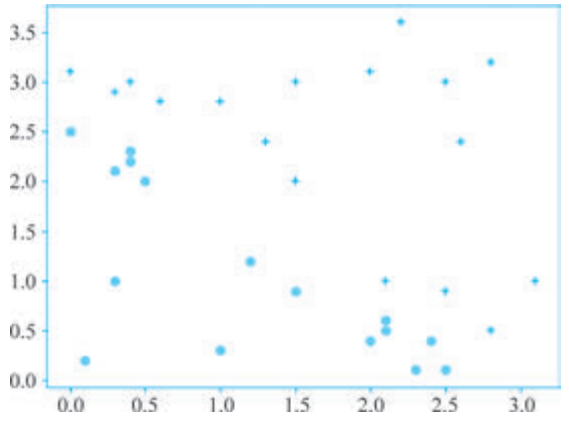

图 3-3 线性可分的二分类问题示例

由图 3-3 可知，二分类问题的决策边界可表示为 $w_1x_1+w_2x_2+b=0$，即假设某样本点'+'使得 $h_w(\boldsymbol{x})=w_1x_1+w_2x_2+b>0$，则该样本点类别判定为 1。考虑到 $h_w(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$ 取值是连续的，无法拟合离散变量'+'，因而 Logistic Regression 需找到分类概率 $P(Y=1)$ 与输入向量 $\boldsymbol{x}$ 的直接关系，从而通过比较概率值判断样本的类别。

给定数据集 $D=(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N),x_i\subseteq\mathbb{R}^n,y_i\in\{0,1\},i=1,2,\cdots,N$，考虑到 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$ 的取值是连续的，无法拟合离散变量，故将其用来拟合连续的条件概率 $p(Y=1|\boldsymbol{x})$。但对于 $\boldsymbol{w}\neq0,\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$ 的取值为 R，不符合概率取值为 0～1，因而考虑采用广义线性模型，最理想的是单位阶跃函数：

$$p(y=1 \mid \boldsymbol{x})=\begin{cases}0, & z<0\\ 0.5, & z=0, z=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b\\ 1, & z>0\end{cases} \tag{3-3}$$

然而,单位阶跃函数不可微,常使用对数几率函数替代单位阶跃函数:

$$y=\frac{1}{1+\mathrm{e}^{-(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b)}} \tag{3-4}$$

替代后可得

$$\ln\frac{y}{1-y}=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b \tag{3-5}$$

其中,将 y 视为 x 为其正例的概率,将 $1-y$ 视为 x 为其反例的概率。两者的比值称为几率(odds),指该事件发生与不发生的概率比值,即

$$\ln(\mathrm{odds})=\ln\frac{y}{1-y} \tag{3-6}$$

若将 y 视为类后验概率的估计,则可重写公式为

$$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=\ln\frac{P(Y=1 \mid \boldsymbol{x})}{1-P(Y=1 \mid \boldsymbol{x})} \tag{3-7}$$

$$P(Y=1 \mid \boldsymbol{x})=\frac{1}{1+\mathrm{e}^{-(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b)}} \tag{3-8}$$

可知,输出 $Y=1$ 的对数几率是由输入 $\boldsymbol{x}$ 的线性函数表示的模型,即逻辑回归模型。当 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$ 的值接近正无穷时,$P(Y=1|\boldsymbol{x})$的概率值接近 1。

为学习逻辑回归模型的参数,采用交叉熵作为损失函数,并使用梯度下降法对参数进行优化。给定包含 N 个训练样本的数据集 $D=(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)$,$x_i\subseteq\mathbb{R}^n$,$y_i\in\{0,1\}$,$i=1,2,\cdots,N$,用 Logistic 回归模型对每个样本 x_i 进行预测,输出其标签为 1 的后验概率,记为 $\hat{y}_i$,

$$\hat{y}_i=\frac{1}{1+\mathrm{e}^{-(\boldsymbol{w}^{\mathrm{T}}x_i+b)}},\quad 1\leqslant i\leqslant N \tag{3-9}$$

由于 $y_i\in\{0,1\}$,样本(x_i,y_i)的真实条件概率可表示为

$$p_r(y_i=1 \mid x_i)=y_i \tag{3-10}$$

$$p_r(y_i=0 \mid x_i)=1-y_i \tag{3-11}$$

使用交叉熵损失函数,其风险函数为

$$\mathcal{R}(w)=-\frac{1}{N}(p_r(y_i=1 \mid x_i)\log\hat{y}_i+p_r(y_i=0 \mid x_i)\log(1-\hat{y}_i)) \tag{3-12}$$

$$=-\frac{1}{N}\sum_{i=1}^{N}(y_i\log\hat{y}_i+(1-y_i)\log(1-\hat{y}_i)) \tag{3-13}$$

风险函数$\mathcal{R}(w)$关于参数 w 的偏导数为

$$\frac{\partial\mathcal{R}(w)}{\partial w}=-\frac{1}{N}\sum_{i=1}^{N}\left(y_i\frac{\hat{y}_i(1-\hat{y}_i)}{\hat{y}_i}x_i-(1-y_i)\frac{\hat{y}_i(1-\hat{y}_i)}{1-\hat{y}_i}x_i\right) \tag{3-14}$$

$$=-\frac{1}{N}\sum_{i=1}^{N}(y_i(1-\hat{y}_i)x_i-(1-y_i)\hat{y}_ix_i) \tag{3-15}$$

$$=\frac{1}{N}\sum_{i=1}^{N}x_i(y_i-\hat{y}_i) \tag{3-16}$$

采用梯度下降法，Logistic 回归的训练过程为：初始化 $w_0 \leftarrow 0$，然后通过式(3-17)迭代更新参数。

$$w_{t+1} \leftarrow w_t + \alpha \frac{1}{N} \sum_{i=1}^{N} x_i (y_i - \hat{y}_{i_{w_t}}) \tag{3-17}$$

其中，α 是学习率，$\hat{y}_{i_{w_t}}$ 是当参数为 w_t 时，Logistic 回归模型的输出。

从式(3-13)可知，风险函数$\mathcal{R}(w)$是关于参数 w 的连续可导的凸函数。因此，除梯度下降法外，Logistic 回归还可以用高阶的优化方法(如牛顿法)进行优化。

3.1.2 支持向量机

支持向量机(Support Vector Machine，SVM)是一种经典的二分类模型，其基本模型定义是特征空间上间隔最大的线性分类器(当采用线性核时)，即支持向量机的学习策略是间隔最大化，最终可转换为一个凸二次规划问题的求解。支持向量机找到的分割超平面具有更好的稳健性，因此广泛使用在很多任务上，并表现出了很强的优势。

给定一个二分类数据集 $D=(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)$，$x_i \subseteq \mathbb{R}^n$，$y_i \in \{0,1\}$，$i=1,2,\cdots,N$，如果两类样本是线性可分的，即存在一个超平面

$$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = 0 \tag{3-18}$$

将两类样本分开。那么，对于每个样本，都有 $y_i(\boldsymbol{w}^{\mathrm{T}}x_i + b) > 0$。

然而，能将训练样本分开的分割超平面可能有很多，如图 3-4 所示。那么，如何选取一个稳健的分割超平面呢？

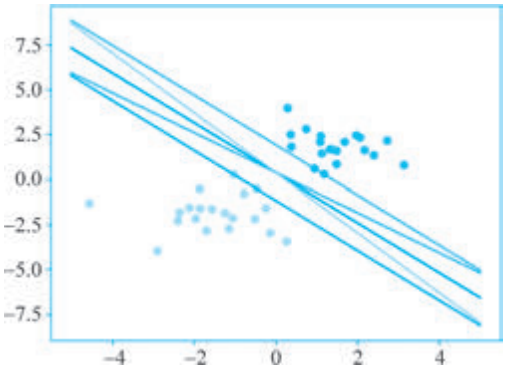

图 3-4 多分割超平面

为此，定义整个数据集 D 中所有样本到分割超平面的最短距离为间隔(Margin)，用 γ 表示。

$$\gamma = \min_i \gamma_i \tag{3-19}$$

间隔 γ 越大，其分割超平面对两个类别的划分越稳定，即不容易受噪声等因素影响。支持向量机的目标是寻找一个超平面$(\boldsymbol{w}^*, b^*)$使得 γ 最大，即

$$\max_{\boldsymbol{w},b} \gamma$$

$$\text{s.t.} \frac{y_i(\boldsymbol{w}^{\mathrm{T}}x_i + b)}{\|\boldsymbol{w}\|} \geqslant \gamma, \quad \forall i \tag{3-20}$$

令 $\| \boldsymbol{w} \| \cdot \gamma = 1$，则式(3-20)等价于 0，即间隔为 $\gamma = \frac{1}{\| \boldsymbol{w} \|}$。

$$\max_{\boldsymbol{w},b} \frac{1}{\| \boldsymbol{w} \|^2}$$

$$\text{s.t. } y_i(\boldsymbol{w}^{\mathrm{T}} x_i + b) \geqslant 1, \quad \forall i \tag{3-21}$$

数据集中所有满足 $y_i(\boldsymbol{w}^{\mathrm{T}} x_i + b) = 1$ 的样本点，都称为支持向量(Support Vector)。

如上所述，对于一个线性可分的数据集，其分割超平面有很多个，但是间隔最大的超平面是唯一的。图 3-5 给定了支持向量机的最大间隔分割超平面的示例，其中在分割线上的样本点为支持向量。

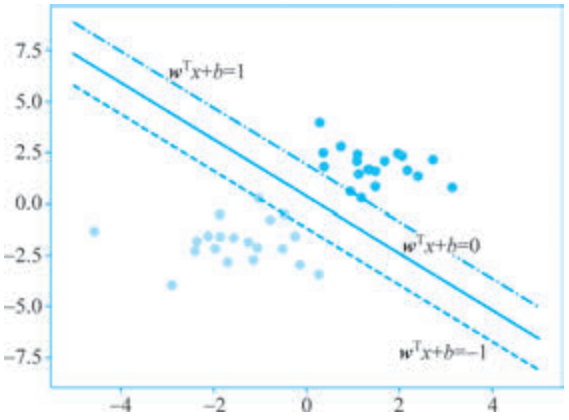

图 3-5 支持向量机示例

为找到最大间隔分割超平面，将式(3-21)的目标函数写为凸优化问题，即

$$\min_{\boldsymbol{w},b} \frac{1}{2} \| \boldsymbol{w} \|^2$$

$$\text{s.t. } 1 - y_i(\boldsymbol{w}^{\mathrm{T}} x_i + b) \leqslant 0, \quad \forall i \tag{3-22}$$

使用拉格朗日乘数法，式(3-22)的拉格朗日函数为

$$\Lambda(\boldsymbol{w}, b, \lambda) = \frac{1}{2} \| \boldsymbol{w} \|^2 + \sum_{i=1}^{N} \lambda_i (1 - y_i(\boldsymbol{w}^{\mathrm{T}} x_i + b)) \tag{3-23}$$

其中，$\lambda_1 \geqslant 0, \lambda_2 \geqslant 0, \cdots, \lambda_N \geqslant 0$ 为拉格朗日乘数。计算 $\Lambda(\boldsymbol{w}, b, \lambda)$ 关于 $\boldsymbol{w}$ 和 b 的导数，并令其等于 0，得到：

$$\boldsymbol{w} = \sum_{i=1}^{N} \lambda_i y_i x_i \tag{3-24}$$

$$0 = \sum_{i=1}^{N} \lambda_i y_i \tag{3-25}$$

将式(3-24)代入式(3-23)，并利用式(3-25)，得到拉格朗日对偶函数：

$$\Gamma(\lambda) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_j \lambda_i y_j y_i \boldsymbol{x}_j^{\mathrm{T}} x_i + \sum_{i=1}^{N} \lambda_i \tag{3-26}$$

支持向量机的主优化问题为凸优化问题，满足强对偶性，即主优化问题可以通过最大化

对偶函数$\max_{\lambda\geqslant 0}\Gamma(\lambda)$求解。对偶函数$\Gamma(\lambda)$是一个凹函数，因此最大化对偶函数是一个凸优化问题，可以通过多种凸优化方法进行求解，得到拉格朗日乘数的最优值λ^*。但由于其约束条件的数量为训练样本数量，一般的优化方法代价比较高，因此在实践中通常采用比较高效的优化方法，如序列最小优化(Sequential Minimal Optimization，SMO)算法等。

根据KKT条件中的互补松弛条件，最优解满足$\lambda_i^*(1-y_i(\boldsymbol{w}^{*\mathrm{T}}x_i+b^*))=0$。如果样本$x_i$不在约束边界上，则$\lambda_i^*=0$，其约束失效；如果样本$x_i$在约束边界上，则$\lambda_i^*\geqslant 0$。那么，这些在约束边界上的样本点称为支持向量(Support Vector)，即离决策平面距离最近的点。

计算出λ^*后，根据式(3-24)计算出最优权重$\boldsymbol{w}^*$，最优偏置b^*可通过任选一个支持向量$(\tilde{x},\tilde{y})$计算得到。

$$b^*=\tilde{y}-\boldsymbol{w}^{*\mathrm{T}}\tilde{x} \tag{3-27}$$

最优参数的支持向量机的决策函数为

$$f(x)=\mathrm{sgn}(\boldsymbol{w}^{*\mathrm{T}}+b^*) \tag{3-28}$$

$$=\mathrm{sgn}\Big(\sum_{i=1}^{N}\lambda_i^* y_i(\boldsymbol{x}_i^{\mathrm{T}}x+b^*)\Big) \tag{3-29}$$

支持向量机的决策函数只依赖$\lambda_i^*>0$的样本点，即支持向量。支持向量机的目标函数可以通过序列最小优化(Sequential Minimal Optimization，SMO)等方法得到全局最优解，因此比其他分类器的学习效率更高。此外，支持向量机的决策函数只依赖支持向量，与训练样本总数无关，分类速度比较快。支持向量机还有一个重要的优点是可以使用核函数(Kernel Function)隐式地将样本从原始特征空间映射到更高维的空间，并解决原始特征空间中的线性不可分问题。同时，在支持向量机的优化问题中，约束条件比较严格。如果训练集中的样本在特征空间中不是线性可分的，就无法找到最优解。为了能够容忍部分不满足约束的样本，可以引入松弛变量ξ。引入松弛变量的间隔称为软间隔(Soft Margin)。

3.1.3 最大熵模型

最大熵模型(Maximum Entropy Model)是一种基于信息论的方法，它通过最大化系统的熵来求解问题。熵在信息论中是信息的度量，事件越不确定，其信息量越大，熵也越大。最大熵法利用已知的自相关函数值外推未知的自相关函数值，从而使谱估计的结果更合理。这种方法不仅在信息论中有所应用，在统计物理中也有所应用，即通过最大化系统的熵来求解问题。

最大熵模型是由最大熵原理推导实现的，所以在讲解最大熵模型前先讲讲最大熵原理。最大熵原理是由杰尼斯(E. T. Jaynes)于1957年提出的，他在这个方向做了开创性的工作。最大熵原理不仅在信息论和统计物理中有所应用，还在热力学中有所应用，如熵增原理规定了能量转换过程中的方向、条件和限度问题。熵增原理是热力学第二定律的微观描述，表明一个封闭系统的熵永远无法自发减小。

最大熵原理认为，学习概率模型时，在所有可能的概率模型(分布)中，熵最大的模型是最好的模型。通常用约束条件确定概率模型的集合。所以，最大熵原理也可表述为在满足约束条件的模型集合中选取熵最大的模型。故计算最大熵根据两个前提去解决问题：①解决问题要满足一定约束；②不做任何假设在约束外的事件发生概率为等概率。

接下来以统计建模的形式描述最大熵模型，给定训练数据集 $D=(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)$，最大熵模型需建立一个概率判别模型，使得对于给定的 $X=x$，可计算得到条件概率分布 $P(Y|X=x)$以预测得到 Y 的值。根据训练数据集可确定联合分布 $P(X,Y)$的经验分布和边缘分布 $P(X)$的经验分布，两者均可通过训练数据集计算得到。联合分布 $P(X,Y)$和边缘分布 $P(X)$的经验分布计算公式如式(3-30)和式(3-31)所示。

$$\widetilde{P}(X=x,Y=y)=\frac{v(X=x,Y=y)}{N} \tag{3-30}$$

$$\widetilde{P}(X=x)=\frac{v(X=x)}{N} \tag{3-31}$$

其中，$v(X=x,Y=y)$表示训练数据集中样本(x,y)出现的频数，$v(X=x)$表示训练数据中输入 x 出现的频数，N 表示样本容量。$\widetilde{P}$ 表示根据已有训练数据集得到变量的经验分布，而不是变量的真实分布。

因此，上述问题变成求部分信息下的最大熵或满足一定约束的最优解，约束条件由特征函数引入。特征函数 $f(x,y)$描述输入 x 和输出 y 之间的某一个事实，其定义如下。

$$f(x,y)=\begin{cases}1, & x、y\text{ 满足某一事实}\\ 0, & x、y\text{ 不满足该事实}\end{cases} \tag{3-32}$$

用 $E_{\widetilde{P}}(f)$表示特征函数 $f(x,y)$关于经验分布 $P(X,Y)$的期望，可得

$$E_{\widetilde{P}}(f)=\sum_{x,y}\widetilde{P}(x,y)f(x,y)=\sum_{x,y}\widetilde{P}(x)\widetilde{P}(y\mid x)f(x,y) \tag{3-33}$$

由于式(3-33)中均为经验分布，故 $E_{\widetilde{P}}(f)$仅与训练数据集有关，并不是真实情况。那么，真实情况应该是什么样的呢？只需将经验分布由真实分布替换上述公式即可：

$$E_P(f)=\sum_{x,y}P(x)P(y\mid x)f(x,y) \tag{3-34}$$

式(3-34)是特征函数 $f(x,y)$在模型上关于概率分布 $P(Y|X)$与 $P(x)$的期望值，最大熵模型需要求解的概率分布是 $P(Y|X)$。但式(3-34)中存在未知分布 $P(x)$。由于并不清楚概率分布 $P(x)$的真实情况，因此采用 $P(x)$的经验分布 $\widetilde{P}(x)$近似表示 $P(x)$，即

$$E_P(f)=\sum_{x,y}\widetilde{P}(x)P(y\mid x)f(x,y) \tag{3-35}$$

为求得最大熵模型，只需使针对训练集的期望 $E_{\widetilde{P}}(f)$和针对模型的期望 $E_P(f)$相等，即

$$E_P(f)=E_{\widetilde{P}}(f) \tag{3-36}$$

$$\sum_{x,y}\widetilde{P}(x)P(y\mid x)f(x,y)=\sum_{x,y}\widetilde{P}(x,y)f(x,y) \tag{3-37}$$

式(3-37)为最大熵模型需要满足的约束。给定 n 个特征函数 $f_i(x,y)$，则有 n 个约束条件，用 C 表示满足约束的模型集合：

$$C=\{P\mid E_P(f_i)=E_{\widetilde{P}}(f_i),i=1,2,\cdots,n\} \tag{3-38}$$

从满足约束的模型集合 C 中找到使得 $P(Y|X)$的熵最大的模型，即最大熵模型。

定义在条件概率分布 $P(Y|X)$上的条件熵为

$$H(P)=\sum_{x}P(x)H(y\mid x) \tag{3-39}$$

$$=\sum_{x}P(x)\sum_{y}-P(y\mid x)\log(y\mid x)$$

$$=-\sum_{x,y}P(x)P(y\mid x)\log(y\mid x)$$

$$=-\sum_{x,y}\widetilde{P}(x)P(y\mid x)\log P(y\mid x) \tag{3-40}$$

综上，给出最大熵模型的形式化定义，给定数据集 $D=(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)$，特征函数 $f_i(x,y),i=1,2,\cdots,n$，根据经验分布得到满足约束集的模型集合 C：

$$\min_{P\in C}-H(P)=\sum_{x,y}\widetilde{P}(x)P(y\mid x)\log P(y\mid x) \tag{3-41}$$

$$\text{s.t. } E_P(f)=E_{\widetilde{P}}(f),\sum_{y}P(y\mid x)=1 \tag{3-42}$$

为求解最大熵模型，采用拉格朗日乘数法，可定义拉格朗日函数为

$$L(P,w)=-H(P)+w_0\Big(1-\sum_{y}P(y\mid x)\Big)+\sum_{i=1}^{n}w_i(E_{\widetilde{P}}(f_i)-E_P(f_i))$$

$$=\sum_{x,y}\widetilde{P}(x)P(y\mid x)\log P(y\mid x)+w_0\Big(1-\sum_{y}P(y\mid x)\Big)+$$

$$\sum_{i=1}^{n}w_i(E_{\widetilde{P}}(f_i)-E_P(f_i)) \tag{3-43}$$

其中，w_0 和 w_i 是拉格朗日乘数法引入的参数。问题形式化为便于拉格朗日对偶处理的极小极大问题：

$$\min_{P\in C}\max_{w}L(P,w) \tag{3-44}$$

由于 $L(P,w)$ 是关于 P 的凸函数，根据拉格朗日对偶可得 $L(P,w)$ 的极小极大问题与极大极小问题是等价的：

$$\min_{P\in C}\max_{w}L(P,w)=\max_{w}\min_{P\in C}L(P,w) \tag{3-45}$$

从而可对 $\min_{P\in C}L(P,w)$ 进行计算，求 $L(P,w)$ 对 $P(y|x)$ 的偏导数：

$$\frac{\partial L(P,w)}{\partial P(y\mid x)}=\frac{\partial}{\partial P(y\mid x)}\Big(\sum_{x,y}\widetilde{P}(x)P(y\mid x)\log P(y\mid x)+$$

$$w_0\Big(1-\sum_{y}P(y\mid x)\Big)+\sum_{i=1}^{n}w_i(E_{\widetilde{P}}(f_i)-E_P(f_i))\Big)$$

$$=\sum_{x,y}\widetilde{P}(x)(1+\log P(y\mid x))-w_0\sum_{y}1+\sum_{i=1}^{n}w_i\Big(0-\sum_{x,y}\widetilde{P}(x)f_i(x,y)\Big)$$

$$=\sum_{x,y}\widetilde{P}(x)(1+\log P(y\mid x))-\sum_{x}\widetilde{P}(x)\sum_{y}w_0+\sum_{i=1}^{n}w_i\Big(0-\sum_{x,y}\widetilde{P}(x)f_i(x,y)\Big)$$

$$=\sum_{x,y}\widetilde{P}(x)(1+\log P(y\mid x))-\sum_{x}\widetilde{P}(x)w_0-\sum_{x,y}\widetilde{P}(x)\sum_{i=1}^{n}w_if_i(x,y)$$

$$=\sum_{x,y}\widetilde{P}(x)\Big(1+\log P(y\mid x)-w_0-\sum_{i=1}^{n}w_if_i(x,y)\Big) \tag{3-46}$$

令偏导数为0，又因为$\sum_{x,y}\widetilde{P}(x)>0$，可得

$$1+\log P(y\mid x)-w_0-\sum_{i=1}^{n}w_i f_i(x,y)=0 \tag{3-47}$$

即

$$P(y\mid x)=\mathrm{e}^{\sum_{i=1}^{n}w_i f_i(x,y)+w_0-1}=\mathrm{e}^{w_0-1}\mathrm{e}^{\sum_{i=1}^{n}w_i f_i(x,y)} \tag{3-48}$$

由于$\sum_{y}P(y\mid x)=1$，代入式(3-48)可得

$$\sum_{y}P(y\mid x)=\sum_{y}\mathrm{e}^{w_0-1}\mathrm{e}^{\sum_{i=1}^{n}w_i f_i(x,y)}=1 \tag{3-49}$$

则$\mathrm{e}^{w_0-1}=1\Big/\sum_{y}\mathrm{e}^{\sum_{i=1}^{n}w_i f_i(x,y)}$，并设其分母为$Z_w(x)$，称为规范化因子。代入式(3-48)可得

$$P_w(y\mid x)=\mathrm{e}^{w_0-1}\mathrm{e}^{\sum_{i=1}^{n}w_i f_i(x,y)}=\frac{1}{Z_w(x)}\mathrm{e}^{\sum_{i=1}^{n}w_i f_i(x,y)} \tag{3-50}$$

该值为对偶问题的内部最小化问题的解，也是需要的模型估计。现在，将式(3-50)代回拉格朗日函数，求解对偶问题外部的极大化问题。由于$\sum_{y}P(y\mid x)=1$，可得拉格朗日函数的推导为

$$\begin{aligned}
L(P,w)=&\sum_{x,y}\widetilde{P}(x)P(y\mid x)\log P(y\mid x)+\\
&\sum_{i=1}^{n}w_i\Big(\sum_{x,y}\widetilde{P}(x,y)f_i(x,y)-\sum_{x,y}\widetilde{P}(x)P(y\mid x)f_i(x,y)\Big)\\
=&\sum_{x,y}\widetilde{P}(x)P(y\mid x)\Big(\log P(y\mid x)-\sum_{i=1}^{n}w_i f_i(x,y)\Big)+\\
&\sum_{i=1}^{n}w_i\sum_{x,y}\widetilde{P}(x,y)f_i(x,y)\\
=&\sum_{x,y}\widetilde{P}(x)P(y\mid x)\Big(\sum_{i=1}^{n}w_i f_i(x,y)-\log Z_w(x)-\sum_{i=1}^{n}w_i f_i(x,y)\Big)+\\
&\sum_{i=1}^{n}w_i\sum_{x,y}\widetilde{P}(x,y)f_i(x,y)\\
=&\sum_{i=1}^{n}w_i\sum_{x,y}\widetilde{P}(x,y)f_i(x,y)-\sum_{x}\widetilde{P}(x)\log Z_w(x)\sum_{y}P(y\mid x)\\
=&\sum_{i=1}^{n}w_i\sum_{x,y}\widetilde{P}(x,y)f_i(x,y)-\sum_{x}\widetilde{P}(x)\log Z_w(x)=\varphi(w)
\end{aligned} \tag{3-51}$$

最后，对每一个w求偏导，并使偏导数等于0，得到非显式的解。再将w的非显示解代入式(3-50)得到条件概率分布$P_w(y|x)$，即最大熵模型。

3.1.4 随机森林

随机森林(Random Forest)属于集成学习,其核心思想是集成多个弱分类器以达到一个强分类器的效果。随机森林算法是 Bagging 算法的进化版,Bagging 算法思想如图 3-6 所示。

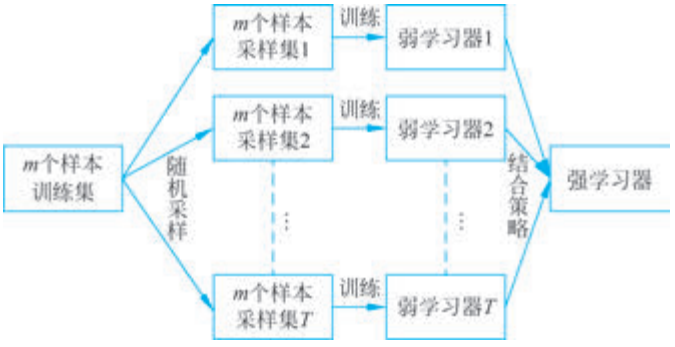

图 3-6 Bagging 算法思想

在 Bagging 算法中,每次从训练集中随机采集与训练集样本数一样的样本,然后利用随机采样得到的样本集训练决策树和神经网络,得到多个弱分类器。在模型预测阶段,对于分类问题,Bagging 算法通过简单投票法得到最多票数的类别或者类别之一为最终的模型输出;而对于回归问题,通常简单平均法对所有弱学习器得到的回归结果进行算术平均得到最终的模型输出。

随机森林算法对 Bagging 算法进行了两方面的改进:一是不同于 Bagging 算法选择决策树和神经网络作为弱分类器,随机森林算法选择 CART 决策树作为弱学习器;二是随机森林算法对 CART 决策树进行了改进。不同于普通的决策树在节点上所有的 n 个样本特征中选择一个最优的特征作为决策树的左右子树划分,随机森林在建立决策树时随机选择节点上的一部分特征并选择其中最优的特征作为决策树的左右子树划分。通过上述操作,进一步增强了模型的泛化能力。

随机森林算法大致分为以下 4 个步骤。

(1) 抽取 N 个样本:一个样本容量为 N 的样本,有放回地抽取 N 次,每次抽取 1 个,最终形成 N 个样本。这选择好了的 N 个样本用来训练一棵决策树,作为决策树根节点处的样本。

(2) 选择 m 个属性:当每个样本有 M 个属性时,在决策树的每个节点需要分裂时,随机从这 M 个属性中选取出 m 个属性,满足条件 $m \ll M$。然后,从这 m 个属性中采用某种策略(如信息增益)来选择 1 个属性作为该节点的分裂属性。

(3) 构造决策树:决策树形成过程中每个节点都要按照步骤(2)分裂(很容易理解,如果下一次该节点选出来的哪一个属性是刚刚其父节点分裂时用过的属性,则该节点已经达到了叶子节点,无须继续分裂了)。一直到不能够再分裂为止。注意:整棵决策树形成过程中没有进行剪枝。

(4) 形成森林:按照步骤(1)~(3)建立大量的决策树,这样就构成了随机森林。

在实际运用中,随机森林树的类目是超参,一般 k 越大,随机森林性能越好,计算成本

越高。另外,抽样样本的个数 N 能够控制均值方差平衡。抽样个数 N 越大,随机性越小,随机森林算法越容易过拟合。当 N 很小时,虽然模型不会过拟合,但模型性能会降低。

3.1.5 条件随机场

条件随机场(Conditional Random Fields,CRF)是给定一组输入序列条件下另一组输出序列的条件概率分布模型,是一种判别式概率模型,常用于标注或分析序列资料,在自然语言处理中广泛应用。

条件随机场是马尔可夫随机场(Markov Random Field,MRF)的一种扩展,它假设输出随机变量构成马尔可夫随机场,即输出变量的当前状态只与当前输入变量和相邻输出变量的状态有关。具体来说,条件随机场通过无向图表示输入序列和输出序列之间的条件依赖关系,每个节点对应一个输出标签,每条边表示两个相邻标签之间的相关性。形式化表示如下:

设 X 与 Y 是随机变量,$P(Y|X)$是给定 X 时 Y 的条件概率分布,若随机变量 Y 构成的是一个无向图的马尔可夫随机场,则称条件概率分布 $P(Y|X)$是条件随机场。

需要注意的是,CRF 的定义中并未要求随机变量 X 与 Y 具有相同的结构,但在实际应用中一般假设 X 与 Y 具有相同的结构,即

$$X=(X_1,X_2,\cdots,X_n),Y=(Y_1,Y_2,\cdots,Y_n) \tag{3-52}$$

这种 X 与 Y 具有相同结构的条件随机场就构成了线性链条件随机场(Linear Chain Conditional Random Fields,Linear-CRF)。线性链条件随机场的示意图如图 3-7 所示。

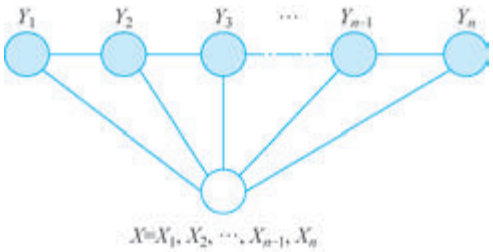

图 3-7 线性链条件随机场的示意图

接下来看 Linear-CRF 的定义:设 $X=(X_1,X_2,\cdots,X_n)$,$Y=(Y_1,Y_2,\cdots,Y_n)$均为线性链表示的随机变量序列,在给定随机变量序列 X 的情况下,随机变量 Y 的条件概率分布 $P(Y|X)$构成条件随机场,即满足马尔可夫性:

$$P(Y_i\mid X,Y_1,Y_2,\cdots,Y_n)=P(Y_i\mid,Y_{i-1},Y_{i+1}) \tag{3-53}$$

则称 $P(Y|X)$为线性链条件随机场。那么,如何将其转换为可学习的机器学习模型呢?这就需要通过特征函数及其权重系数来定义了。在 Linear-CRF 中,特征函数分为两类,第一类是定义在 Y 节点上的节点特征函数,这类特征函数只和当前节点有关,记为

$$s_l(y_i,x,i),l=1,2,\cdots,L \tag{3-54}$$

其中,L 是定义在该节点的节点特征函数的总个数,i 是当前节点在序列中的位置。第二类是定义在 Y 节点上的局部特征函数,这类特征函数只和当前节点与上一个节点有关,记为

$$t_k(y_{i-1},y_i,x,i),k=1,2,\cdots,K \tag{3-55}$$

其中,K 是定义在该节点的局部特征函数的总个数,i 是当前节点在序列中的位置。无论是节点特征函数,还是局部特征函数,它们的取值只能是 0 或者 1,即满足特征条件或者不满足特征条件。同时,我们可以为每个特征函数赋予一个权值,用以表达我们对这个特征函数的信任度。假设 t_k 的权重系数是 λ_k,s_l 的权重系数是 μ_l,则可得 Linear-CRF 的参数化形式表示:

$$P(y \mid x)=\frac{1}{Z(x)}\exp\Big(\sum_{i,k}\lambda_k t_k(y_{i-1},y_i,x,i)+\sum_{i,l}\mu_l s_l(y_i,x,i)\Big) \tag{3-56}$$

其中，$Z(x)$为规范化因子：

$$Z(x)=\sum_{y}\exp\Big(\sum_{i,k}\lambda_k t_k(y_{i-1},y_i,x,i)+\sum_{i,l}\mu_l s_l(y_i,x,i)\Big) \tag{3-57}$$

在 Linear-CRF 中，每个特征函数定义了一个 Linear-CRF 的规则，其系数定义了这个规则的可信度。所有的规则和其可信度一起构成 Linear-CRF 的最终的条件概率分布。

【例 3.1】 利用 Linear-CRF 进行词性标注。

假设输入是包含 3 个词的句子，即 $X=(X_1,X_2,X_3)$，输出的词性标记为 $Y=(Y_1,Y_2,Y_3)$，其中 $Y\in\{1(名词),2(动词)\}$。假设标记为 1 的特征函数有

$$\begin{aligned}
&t_1=t_1(y_{i-1}=1,y_i=2,x,i),\quad i=2,3,\lambda_1=1\\
&t_2=t_2(y_1=1,y_2=1,x,2)\lambda_2=0.5\\
&t_3=t_3(y_2=2,y_3=1,x,3)\lambda_3=1\\
&t_4=t_4(y_1=2,y_2=1,x,2)\lambda_4=1\\
&t_5=t_5(y_2=2,y_3=2,x,3)\lambda_5=0.2\\
&s_1=s_1(y_1=1,x,1)\mu_1=1\\
&s_2=s_2(y_i=2,x,i),\quad i=1,2,\mu_2=0.5\\
&s_3=s_3(y_i=1,x,i),\quad i=1,2,\mu_3=0.8\\
&s_4=s_4(y_3=2,x,3)\mu_4=0.5
\end{aligned} \tag{3-58}$$

求解标记(1,2,2)的非规范化概率。

由 Linear-CRF 的参数化公式可得

$$P(y \mid x)\propto\exp\left[\sum_{k=1}^{5}\lambda_k\sum_{i=2}^{3}t_k(y_{i-1},y_i,x,i)+\sum_{l=1}^{4}\mu_l\sum_{i=1}^{3}s_l(y_i,x,i)\right] \tag{3-59}$$

标记(1,2,2)可得

$$P(y_1=1,y_2=2,y_3=2 \mid x)\propto\exp(3.2) \tag{3-60}$$

特征函数集的大小和多样性影响了条件随机场的模型复杂度和表达能力。一般来说，特征函数越多越丰富，条件随机场的性能越好，但也会增加计算的代价和过拟合的风险。因此，特征函数集的定义需要平衡特征的有效性和效率，以达到最佳的效果。

模型训练完成后，每一个特征函数有一个权重，这个权重表示该特征函数对标注序列的评分和概率的贡献程度。特征函数的权重可以通过最大似然估计(Maximum Likelihood Estimation，MLE)或最大后验估计(Maximum A Posteriori Estimation，MAP)求解，这两种方法都是基于训练数据优化对数似然函数或对数后验概率的方法，可以用梯度下降(Gradient Descent)或拟牛顿法(Quasi-Newton Method)等优化算法实现。

3.1.6 集成学习

集成学习是通过构建并结合多个学习器完成学习任务，其过程是：先产生一组“个体学习器”，再用某种策略将它们结合起来。个体学习器一般就是常见的机器学习算法，如决策树、神经网络等。集成一般有两种：同质和异质。同质是指个体学习器全是同一类型，这种

同质集成中的个体学习器又称“基学习器”。异质是指个体学习器包含不同类型的学习算法，比如同时包含决策树和神经网络。一般常用的集成学习方法是同质的，即个体学习器属于同一类型。集成学习的过程如图 3-8 所示。

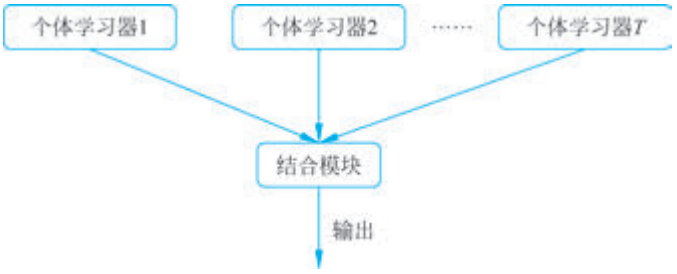

图 3-8 集成学习的过程

根据结合方式的不同，一般可将集成学习方法分为 Bagging 和 Boosting 方法。Bagging (Boostrap Aggregating)是由 Breiman 于 1996 年提出的，其基本思想如下：①每次采用有放回的抽样从训练集中取 n 个训练样本组成新的训练集；②利用新的训练集，训练得到 M 个子模型$\{h_1,h_2,\cdots,h_M\}$；③对于分类问题，采用投票的方法，得票最多子模型的分类类别为最终的类别；对于回归问题，采用简单的平均方法得到预测值。而 Boosting 是一种提升算法，可以将弱的学习算法提升(boost)为强的学习算法，其基本思路如下：①利用初始训练样本集训练得到一个基学习器；②提高被基学习器误分的样本的权重，使得被错误分类的样本在下一轮训练中可以得到更大的关注，利用调整后的样本训练得到下一个基学习器；③重复上述步骤，直至得到 M 个学习器；④对于分类问题，采用有权重的投票方式；对于回归问题，采用加权平均得到预测值。

3.2 深度学习模型

3.2.1 感知机

感知机(Perceptron)是二分类的线性分类模型，属于监督学习算法。输入为实例的特征向量，输出为实例的类别(取+1 和-1)。如图 3-9 所示，感知机旨在求出将输入空间中的实例划分为两类的分离超平面。为求得超平面，感知机导入了基于误分类的损失函数，利用梯度下降法对损失函数进行最优化求解。如果训练数据集是线性可分的，则感知机一定能求得分离超平面。如果训练数据集是非线性可分的数据，则无法获得超平面。

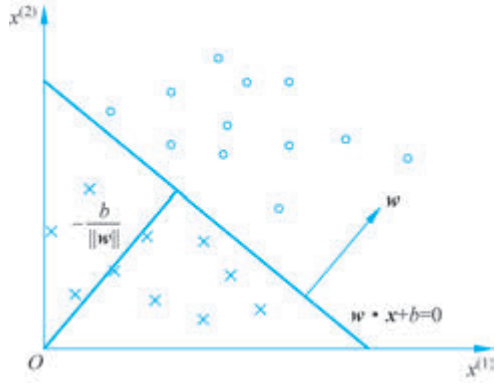

图 3-9 感知机模型

假设训练数据集为 $D=\{(x_i,y_i)\}_{i=1}^m$，其中，$x_i\in X\subseteq\mathbb{R}^n$，$y_i\in Y=\{+1,-1\}$，则感知机模型为

$$f(\boldsymbol{x})=\text{sign}(\boldsymbol{w}\cdot\boldsymbol{x}+b) \tag{3-61}$$

其中，$\boldsymbol{w}$ 和 b 称为感知机模型参数，$\boldsymbol{w}\in\mathbb{R}^n$ 叫作权值或者权值向量，$b\in\mathbb{R}$ 叫作偏置。$\boldsymbol{w}\cdot\boldsymbol{x}$ 表示 $\boldsymbol{w}$ 和 $\boldsymbol{x}$ 的内积。sign 函数是符号函数：

$$\text{sign}(\boldsymbol{x})=\begin{cases}+1, & \boldsymbol{x}\geqslant 0\\ -1, & \boldsymbol{x}<0\end{cases} \tag{3-62}$$

感知机模型的其中一个超平面是

$$\boldsymbol{w}\cdot\boldsymbol{x}+\boldsymbol{b}=0 \tag{3-63}$$

其中，$\boldsymbol{w}$ 是超平面的法向量，$\boldsymbol{b}$ 是超平面的截距。这个超平面将样本点分为正负两类，即对所有 $y_i=+1$ 的样本，都有 $\boldsymbol{w}\cdot\boldsymbol{x}+\boldsymbol{b}>0$；对所有 $y_i=-1$ 的样本，都有 $\boldsymbol{w}\cdot\boldsymbol{x}+\boldsymbol{b}<0$。

假设训练数据集是线性可分的，为找出一个能够将训练数据集正实例点和负实例点完全正确分开的超平面，即确定感知机模型参数 $\boldsymbol{w}$、b，需确定一个学习策略，即定义（经验）损失函数，并将损失函数极小化。损失函数的一个自然选择是误分类点的总数，但是该函数不是连续可导函数，不易优化。因此，感知机采用的损失函数是误分类点到超平面的总距离。该函数的形式化如下。

假设直线方程为 $Ax+By+C=0$，点 P 的坐标为 (x_0,y_0)，则点到直线的距离为

$$d=\frac{Ax_0+By_0+C}{\sqrt{A^2+B^2}} \tag{3-64}$$

由此可知，假设超平面是 $h=\boldsymbol{w}\cdot\boldsymbol{x}+\boldsymbol{b}$，样本点 x' 到超平面的距离如下：

$$d=\frac{\boldsymbol{w}\cdot\boldsymbol{x}'+\boldsymbol{b}}{\|\boldsymbol{w}\|} \tag{3-65}$$

其中，$\|\boldsymbol{w}\|$ 是 $\boldsymbol{w}$ 的 L_2 范式。对于误分类数据 (x_i,y_i)，有 $-y_i(\boldsymbol{w},x_i+\boldsymbol{b})=|\boldsymbol{w}\cdot x_i+\boldsymbol{b}|>0$，因此，误分类点 x_i 到超平面 S 的距离是

$$-\frac{1}{\|\boldsymbol{w}\|}y_i(\boldsymbol{w}\cdot x_i+\boldsymbol{b}) \tag{3-66}$$

假设超平面 S 的误分类点集合为 M，那么所有误分类点到超平面 S 的总距离为

$$-\frac{1}{\|\boldsymbol{w}\|}\sum_{x_i\in M}y_i(\boldsymbol{w}\cdot x_i+\boldsymbol{b}) \tag{3-67}$$

因此，在不考虑 $-\frac{1}{\|\boldsymbol{w}\|}$ 的情形下，可得感知机学习的经验风险损失函数为

$$L(\boldsymbol{w},b)=-\sum_{x_i\in M}y_i(\boldsymbol{w}\cdot x_i+\boldsymbol{b}) \tag{3-68}$$

至此，感知机学习问题就转换为求解损失函数的最优化问题。由于感知机学习算法是误分类样本驱动的，因此可采用基于随机梯度下降法（SGD）进行优化求解。

3.2.2 前馈神经网络

前馈神经网络（Feedforward Neural Network，FNN）是最早发明的简单人工神经网络。前馈神经网络也经常称为多层感知机（Multi-Layer Perceptron，MLP）。但多层感知机的叫法并不是十分合理，因为前馈神经网络其实是由多层的 Logistic 回归模型（连续的非线性函

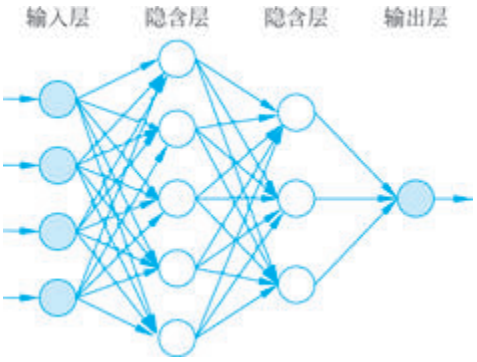

图 3-10 多层前馈神经网络

数)组成,而不是由多层的感知机(不连续的非线性函数)组成。

在前馈神经网络中,各神经元分别属于不同的层。每一层的神经元可以接收前一层神经元的信号,并产生信号输出到下一层。第 0 层称为输入层,最后一层称为输出层,其他中间层称为隐含层。整个网络中无反馈,信号从输入层向输出层单向传播,可用一个有向无环图表示。多层前馈神经网络如图 3-10 所示。

多层前馈神经网络涉及的记号描述如表 3-1 所示。

表 3-1 多层前馈神经网络涉及的记号描述

记 号	含 义
L	神经网络的层数
M_l	第 l 层神经元的个数
$f_l(\cdot)$	第 l 层神经元的激活函数
$\boldsymbol{W}^{(l)} \in \mathbb{R}^{M_l \times M_{l-1}}$	第 $l-1$ 层到第 l 层的权重矩阵
$b^{(l)} \in \mathbb{R}^{M_l}$	第 $l-1$ 层到第 l 层的偏置
$Z^{(l)} \in \mathbb{R}^{M_l}$	第 l 层神经元的净输入(净活性值)
$a^{(l)} \in \mathbb{R}^{M_l}$	第 l 层神经元的输出(活性值)

令 $a^{(0)}=x$,前馈神经网络通过不断迭代下面的公式进行信息传播:

$$Z^{(l)}=\boldsymbol{W}^{(l)}a^{(l-1)}+b^{(l)} \tag{3-69}$$

$$a^{(l)}=f_l(Z^{(l)}) \tag{3-70}$$

首先根据第 $l-1$ 层神经元的活性值(Activation)$a^{(l)}$ 计算出第 l 层神经元的净活性值(Net Activation)$Z^{(l)}$,然后经过一个激活函数得到第 l 层神经元的活性值。因此,我们也可以把每个神经层看作一个仿射变换(Affine Transformation)和一个非线性变换。

式(3-69)和式(3-70)也可以合并为

$$Z^{(l)}=\boldsymbol{W}^{(l)}f_{l-1}(Z^{(l-1)})+b^{(l)} \tag{3-71}$$

或者

$$a^{(l)}=f_l(\boldsymbol{W}^{(l)}a^{(l-1)}+b^{(l)}) \tag{3-72}$$

这样,前馈神经网络可以通过逐层的信息传递,得到网络最后的输出 $a^{(L)}$。整个网络可以看作一个复合函数 $\phi(\boldsymbol{x};\boldsymbol{W},b)$,将向量 $\boldsymbol{x}$ 作为第 1 层的输入 $a^{(0)}$,将第 L 层的输出 $a^{(L)}$ 作为整个函数的输出。

$$\boldsymbol{x}=a^{(0)}\rightarrow Z^{(1)}\rightarrow a^{(1)}\rightarrow Z^{(2)}\rightarrow\cdots\rightarrow a^{(L-1)}\rightarrow Z^{(L)}\rightarrow a^{(L)}=\phi(\boldsymbol{x};\boldsymbol{W},b), \tag{3-73}$$

其中,$\boldsymbol{W}$、b 表示网络中所有层的连接权重和偏置。

前馈神经网络具有很强的拟合能力,常见的连续非线性函数都可以用前馈神经网络近似。通用近似定理证明了神经网络的计算能力可以近似一个给定的连续函数。然而,该定理并没有给出如何找到这样一个网络,以及是否为最优的。因此,在机器学习应用时,真实的映射函数并不知道,一般通过经验风险最小化和正则化进行参数学习。具体地,对于基于

前馈神经网络的分类器，采用交叉熵损失函数作为模型的损失函数，然后采用梯度下降法对参数进行优化。梯度下降法需要计算损失函数对参数的偏导数，可通过链式法则逐一对每个参数求偏导，但计算效率比较低。在实际神经网络的训练中经常使用反向传播算法高效地计算梯度。

3.2.3 递归神经网络

循环神经网络(Recurrent Neural Network，RNN)是一类用于处理序列数据的神经网络。就像卷积网络是专门用于处理网格化数据 X(如一个图像)的神经网络，循环神经网络是专门用于处理序列 $x^{(1)},x^{(2)},\cdots,x^{(t)}$ 的神经网络。正如卷积网络可以很容易地扩展到处理具有很大宽度和高度的图像，以及处理大小可变的图像；循环网络也可扩展到更长的序列，比传统非序列特化网络(如全连接网络或卷积网络)长得多。大多数循环网络也能处理可变长的序列。

RNN 的结构细节如图 3-11 所示，可形式化为

$$s_t = f(Ux_t + Ws_{t-1}) \tag{3-74}$$

$$y = g(Vs_t) \tag{3-75}$$

其中，x_t 表示 t 时刻的输入，s_t 表示 t 时刻的隐藏状态，$f(\cdot)$和 $g(\cdot)$表示激活函数。常见的激活函数有 tanh 和 ReLU 函数，U、W、V 表示循环神经网络模型的参数。

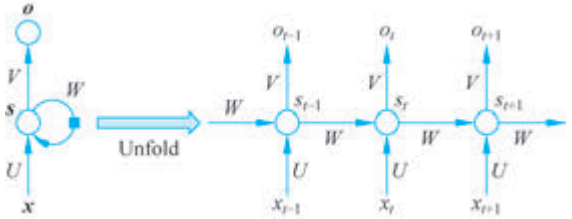

图 3-11 RNN 的结构细节

循环神经网络不同于其他神经网络的显著特征是，其在每个网络层中共享权重参数。尽管如此，循环神经网络的权重参数仍可通过反向传播和梯度下降过程进行调整，以学习模型权重参数。循环神经网络利用随时间推移的反向传播(Back-Propagation Through Time，BPTT)算法确定梯度，这与传统的反向传播略有不同，因为它特定于序列数据。BPTT 的原理与传统的反向传播相同，模型通过计算输出层与输入层之间的误差来训练自身。这些计算帮助我们适当地调整和拟合模型的参数。BPTT 与传统方法的不同之处在于，BPTT 会在每个时间步长对误差求和，而前馈网络则不需要对误差求和，因为它们不会在每层共享参数。

通过 BPTT 算法更新模型的参数，循环神经网络往往会产生两个问题，即梯度爆炸和梯度消失。梯度爆炸和消失问题由梯度的大小定义导致，即损失函数是沿着错误曲线的斜率导致的。如果梯度过小，它会更新权重参数，让梯度继续变小，直到变得可以忽略，即为 0。当出现这种情况时，算法参数就不再学习。如果梯度过大，就会发生梯度爆炸，导致模型不稳定。在这种情况下，模型权重会变得太大，并最终被表示为 NaN。

为解决朴素循环神经网络存在的梯度爆炸和梯度消失问题，研究者提出一些循环神经网络的变体，如长短期记忆网络(Long Short-Term Memory，LSTM)、门控循环单元(Gate

Recurrent Unit,GRU)等。长短期记忆网络由 SeppHochreiter 和 JuergenSchmidhuber 于 1997 年提出,用于避免长期依赖问题和梯度消失问题。为此,LSTM 特意在神经网络的隐含层中设计了一个记忆控制器"元胞"(cell)。LSMT 包含 4 个核心模块:一个输入门 i、一个输出门 o、一个遗忘门 f 和一个记忆控制器 c。这些组件控制着长短期记忆网络中的输出所需信息的流动。LSTM 单元的计算公式如下所示。

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)\\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f)\\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{xi}x_t + W_{hi}h_{t-1} + b_i)\\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o)\\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
\tag{3-76}
$$

遗忘门决定 LSTM 模型要从单元中抛弃哪些信息,输入门的 sigmoid 层决定将要更新哪些值,而 tanh 层创建一个信息的候选向量$\widetilde{\boldsymbol{C}}_t$。接着把旧的状态乘以 f_t,用于遗忘之前模型决定遗忘的信息,然后再加上 $i_t * \widetilde{\boldsymbol{C}}_t$,得到新的候选值。然后使用 sigmoid 层决定要输出单元状态的哪个部分,然后用 tanh 处理单元状态,即将状态映射到$-1\sim1$。最后将其与 sigmoid 门输出值相乘,得到最终的结果。LSTM 单元的结构如图 3-12 所示。

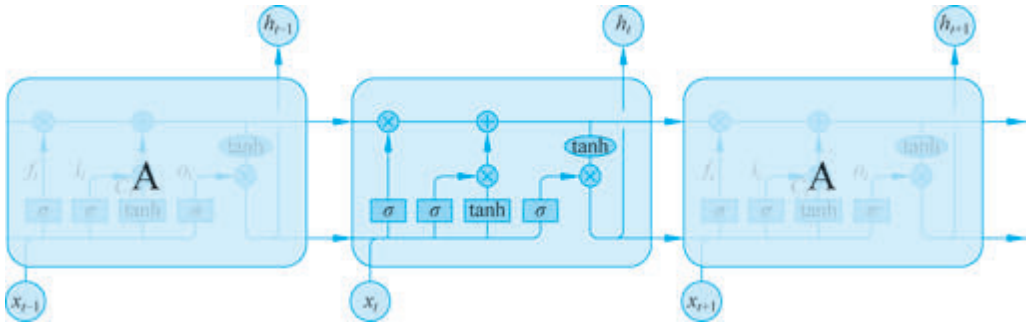

图 3-12 LSTM 单元的结构

门控循环单元类似于 LSTM 单元,旨在解决 RNN 模型的短期记忆问题,但它不使用"元胞状态"调节信息,而是使用隐藏状态。它将 LSTM 单元中输入门和遗忘门结合成一个单独的"更新门",同时合并了细胞状态和隐含状态。因此,在 GRU 中只有两个门结构:一个重置门和一个更新门,如图 3-13 所示,计算公式也简化为

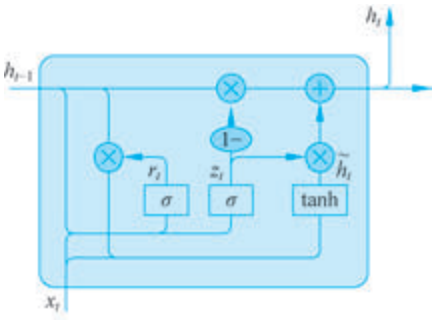

图 3-13 GRU 结构

$$
\begin{aligned}
z_t &= \sigma(W_z \cdot [h_{t-1}, x_t])\\
r_t &= \sigma(W_r \cdot [h_{t-1}, x_t])\\
\widetilde{h}_t &= \tanh(W \cdot [r_t \cdot h_{t-1}, x_t])\\
h_t &= (1 - z_t) \cdot h_{t-1} + z_t \cdot \widetilde{h}_t
\end{aligned}
\tag{3-77}
$$

通过以上操作,相对于 LSTM 模型,门控循环单元模型精简、参数少,但拟合能力相对较弱,其更适合小规模且不复杂的数据集。

3.2.4 卷积神经网络

卷积神经网络(Convolutional Neural Network,CNN 或 ConvNet)是一种具有局部连接、权重共享等特性的深层前馈神经网络。

卷积神经网络最早主要用来处理图像信息。在用全连接前馈网络处理图像时,存在以下两个问题。

(1) **参数太多**:如果输入图像大小为 100×100×3(即图像高度为 100,宽度为 100 以及 RGB 3 个颜色通道),在全连接前馈网络中,第一个隐含层的每个神经元到输入层都有 100×100×3=30000 个互相独立的连接,每个连接都对应一个权重参数。随着隐含层神经元数量的增多,参数的规模也会急剧增加。这会导致整个神经网络的训练效率非常低,也很容易过拟合。

(2) **局部不变性特征**:自然图像中的物体都具有局部不变性特征,如尺度缩放、平移、旋转等操作不影响其语义信息。而全连接前馈网络很难提取这些局部不变性特征,一般需要进行数据增强来提高性能。

卷积神经网络是受生物学上感受野机制的启发而提出的。感受野(Receptive Field)机制主要指听觉、视觉等神经系统中一些神经元的特性,即神经元只接收其所支配的刺激区域内的信号。在视觉神经系统中,视觉皮层中的神经细胞的输出依赖视网膜上的光感受器。视网膜上的光感受器受刺激兴奋时,将神经冲动信号传到视觉皮层,但不是所有视觉皮层中的神经元都会接收这些信号。一个神经元的感受野是指视网膜上的特定区域,只有这个区域内的刺激才能够激活该神经元。

目前的卷积神经网络一般是由卷积层、池化层和全连接层交叉堆叠而成的前馈神经网络,结构示意图如图 3-14 所示。卷积层的作用是提取一个局部区域的特征,不同的卷积核相当于不同的特征提取器。池化层(Pooling Layer)也叫子采样层(Subsampling Layer),其作用是进行特征选择,降低特征数量,从而减少参数数量。因此,卷积神经网络有 3 个结构上的特性:局部连接、权重共享以及汇聚。这些特性使得卷积神经网络具有一定程度上的平移、缩放和旋转不变性。和前馈神经网络相比,卷积神经网络的参数更少。

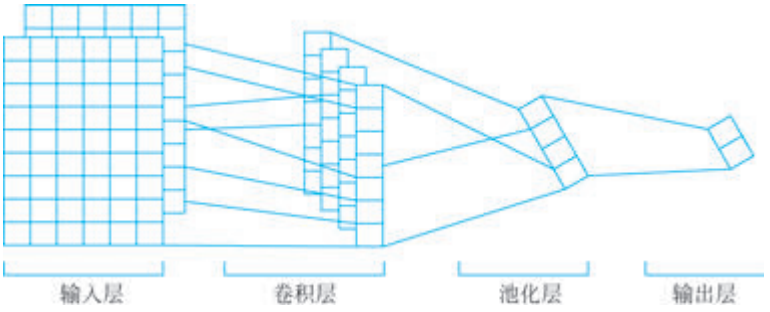

图 3-14 CNN 结构示意图

卷积层采用卷积代替神经网络中的全连接,第 l 层的净输入 $z^{(l)}$ 为第 $l-1$ 层活性值 $a^{(l-1)}$ 和卷积核 $\boldsymbol{w}^{(l)} \in \mathbb{R}^k$ 的卷积,即

$$z^{(l)} = \boldsymbol{w}^{(l)} \otimes a^{(l-1)} + b^{(l)} \tag{3-78}$$

其中,卷积核 $\boldsymbol{w}^{(l)} \in \mathbb{R}^k$ 为可学习的权重向量,$b^{(l)} \in \mathbb{R}^k$ 为可学习的偏置。常用的卷积操作

有一维卷积和二维卷积。一维卷积经常用在信号处理中，用于计算信号的延迟累积。假设一个信号发生器每个时刻 t 产生一个信号 x_t，其信息的衰减率为 w_k，即在 $k-1$ 个时间步长后，信息为原来的 w_k 倍。假设 $w_1=1, w_2=\frac{1}{2}, w_3=\frac{1}{4}$，那么在时刻 t 收到的信号 y_t 为当前时刻产生的信息和以前时刻延迟信息的叠加，即

$$y_t = 1\times x_t + \frac{1}{2}\times x_{t-1} + \frac{1}{4}\times x_{t-2} \tag{3-79}$$

$$= w_1 \times x_t + w_2 \times x_{t-1} + w_3 \times x_{t-2} \tag{3-80}$$

$$= \sum_{k=1}^{3} w_k x_{t-k+1} \tag{3-81}$$

这里把 $w_1, w_2, \cdots$ 称为滤波器(Filter)或卷积核(Convolution Kernel)。假设滤波器长度为 K，它和一个信号序列 $x_1, x_2, \cdots$ 的卷积为

$$y_t = \sum_{k=1}^{K} w_k x_{t-k+1} \tag{3-82}$$

卷积也经常用在图像处理中。因为图像为一个二维结构，所以需要将一维卷积进行扩展。给定一个图像 $x\in\mathbb{R}^{M\times N}$ 和一个滤波器 $W\in\mathbb{R}^{U\times V}$，一般 $U\ll M, V\ll N$，其卷积为

$$y_{i,j} = \sum_{u=1}^{U}\sum_{v=1}^{V} w_{uv} x_{i-u+1, j-v+1} \tag{3-83}$$

【例 3.2】 给定一维信号序列 $X=\{1,1,-1,1,1,1,-1,1,1\}$、低频滤波器 $W_1=\left\{\frac{1}{3},\frac{1}{3},\frac{1}{3}\right\}$ 和高频滤波器 $W_2=\{1,-2,1\}$，计算卷积后的向量。

低频滤波器卷积后输出的向量为 $\left\{1\times\frac{1}{3}+1\times\frac{1}{3}+(-1)\times\frac{1}{3}, 1\times\frac{1}{3}+(-1)\times\frac{1}{3}+1\times\frac{1}{3}, (-1)\times\frac{1}{3}+1\times\frac{1}{3}+1\times\frac{1}{3}, 1\times\frac{1}{3}+1\times\frac{1}{3}+1\times\frac{1}{3}, 1\times\frac{1}{3}+1\times\frac{1}{3}+(-1)\times\frac{1}{3}, 1\times\frac{1}{3}+(-1)\times\frac{1}{3}+1\times\frac{1}{3}, (-1)\times\frac{1}{3}+1\times\frac{1}{3}+1\times\frac{1}{3}\right\}=\left\{\frac{1}{3},\frac{1}{3},\frac{1}{3},1,\frac{1}{3},\frac{1}{3},\frac{1}{3}\right\}$，

高频滤波器卷积后输出的向量为 $\{1\times1+1\times(-2)+(-1)\times1, 1\times1+(-1)\times(-2)+1\times1, (-1)\times1+1\times(-2)+1\times1, 1\times1+1\times(-2)+1\times1, 1\times1+1\times(-2)+(-1)\times1, 1\times1+(-1)\times(-2)+1\times1, (-1)\times1+1\times(-2)+1\times1\}=\{-2,4,-2,0,-2,4,-2\}$.

【例 3.3】 给定图像向量 $\boldsymbol{X}=\begin{bmatrix}1 & 1 & -12 & 1\\ -1 & 0 & 10 & 1\\ 2 & 1 & 11 & -1\\ 0 & -1 & 12 & 1\\ 1 & 2 & 11 & 1\end{bmatrix}$，卷积核 $\boldsymbol{W}=\begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & -1\end{bmatrix}$，计算卷积后的向量。

卷积后输出的向量为：$y_{00}=1\times1+11\times(-1)=-10$，$y_{01}=1\times1+(-1)\times(-1)=2$，$y_{10}=(-1)\times1+12\times(-1)=-13$，$y_{11}=1\times(-1)=-1$，$y_{20}=2\times1+11\times(-1)=-9$，$y_{21}=1\times1+1\times(-1)=0$，卷积后的向量 $\boldsymbol{Y}=\begin{bmatrix}-10 & 2\\ -13 & -1\\ -9 & 0\end{bmatrix}$。

卷积层虽然可以显著减少网络中连接的数量，但特征映射组中的神经元个数并没有显著减少。如果后面接一个分类器，分类器的输入维数依然很高，很容易出现过拟合。为了解决这个问题，可以在卷积层之后加上一个池化层，从而降低特征维数，避免过拟合。

假设池化层的输入特征映射组为 $X \in \mathbb{R}^{M \times N \times D}$，对于其中每一个特征映射 $x^d \in \mathbb{R}^{M \times N}$，$1 \leqslant d \leqslant D$，将其划分为很多区域 $R^d_{m,n}$，$1 \leqslant m \leqslant M'$，$1 \leqslant n \leqslant N'$，这些区域可以重叠，也可以不重叠。池化(Pooling)是指对每个区域进行下采样(Down Sampling)得到一个值，作为这个区域的概括。

常用的汇聚函数有两种。

(1) **最大池化(Maximum Pooling 或 Max Pooling)**：对于一个区域 $R^d_{m,n}$，选择这个区域内所有神经元的最大活性值作为这个区域的表示，即

$$y^d_{m,n} = \max_{i \in R^d_{m,n}} x_i \tag{3-84}$$

其中，x_i 为区域 R^d_k 内每个神经元的活性值。

(2) **平均汇聚(Mean Pooling)**：一般取区域内所有神经元活性值的平均值，即

$$y^d_{m,n} = \frac{1}{|R^d_{m,n}|} \sum_{i \in R^d_{m,n}} x_i \tag{3-85}$$

对每一个输入特征映射 x^d 的 $M' \times N'$ 个区域进行子采样，得到池化层的输出特征映射 $y^d = \{y^d_{m,n}\}$，$1 \leqslant m \leqslant M'$，$1 \leqslant n \leqslant n'$。

在卷积网络中，参数为卷积核中的权重以及偏置。和全连接前馈网络类似，卷积网络也可以通过误差反向传播算法进行参数学习。在全连接前馈神经网络中，梯度主要通过每一层的误差项 δ 进行反向传播，并进一步计算每层参数的梯度。在卷积神经网络中，主要有两种不同功能的神经层：卷积层和池化层。而参数为卷积核以及偏置，因此只需要计算卷积层中参数的梯度。

3.2.5 生成式对抗网络

生成式对抗网络(Generative Adversarial Network，GAN)由 Goodfellow 等于 2014 年提出，它可以替代 VAE 学习图像的潜在空间。它能够迫使生成图像与真实图像在统计上几乎无法区分，从而生成相当逼真的合成图像。

对 GAN 的一种直观理解是，想象一名伪造者试图伪造一幅毕加索的画作。一开始，伪造者非常不擅长这项任务。他将自己的一些赝品与毕加索真迹混在一起，并将其展示给一位艺术商人。艺术商人对每幅画进行真实性评估，并向伪造者给出反馈，告诉他是什么让毕加索作品看起来像一幅毕加索作品。伪造者回到自己的工作室，并准备一些新的赝品。随着时间的推移，伪造者变得越来越擅长模仿毕加索的风格，艺术商人也变得越来越擅长找出赝品。最后，他们手上拥有了一些优秀的毕加索赝品。

这就是 GAN 的工作原理：一个伪造者网络和一个专家网络，二者训练的目的都是打败彼此。因此，GAN 由以下两部分组成。

◇ **生成器网络**(Generator Network)：它以一个随机向量(潜在空间中的一个随机点)作为输入，并将其解码为一幅合成图像。

◇ **判别器网络**(Discriminator Network)或**对手**(Adversary)：以一幅图像(真实的或合

成的均可)作为输入,并预测该图像是来自训练集,还是由生成器网络创建。

训练生成器网络的目的是使其能够欺骗判别器网络。因此,随着训练的进行,它能够逐渐生成越来越逼真的图像,即看起来与真实图像无法区分的人造图像,以至于判别器网络无法区分二者(见图 3-15)。与此同时,判别器也在不断适应生成器逐渐提高的能力,为生成图像的真实性设置了很高的标准。一旦训练结束,生成器就能将其输入空间中的任何点转换为一幅可信图像(见图 3-16)。与 VAE 不同,这个潜在空间无法保证具有有意义的结构,而且它还是不连续的。

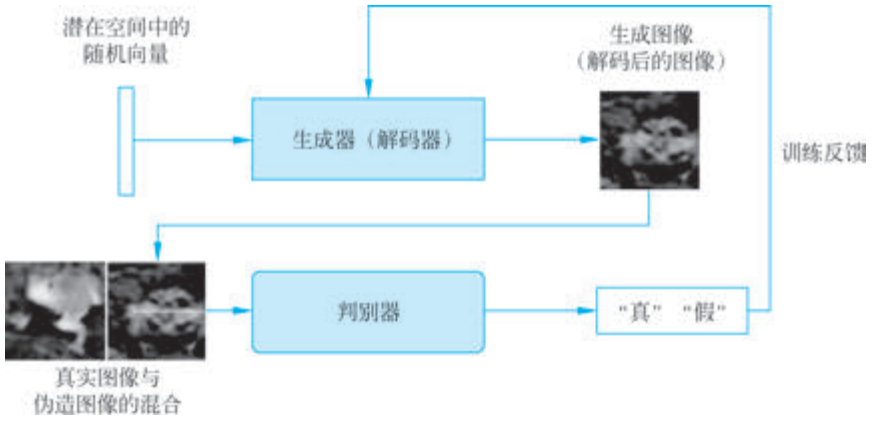

图 3-15 GAN 网络结构

图 3-16 潜在空间的"居民"(MikeTyka 利用在人脸数据集上训练的多级 GAN 所生成的图像)

值得注意的是,GAN 这个系统与本书中其他任何训练方法都不同,它的优化最小值是不固定的。通常,梯度下降是沿着静态的损失地形滚下山坡。但对于 GAN 而言,每下山一步,都会对整个地形造成一点改变。它是一个动态的系统,其最优化过程寻找的不是一个最小值,而是两股力量之间的平衡。因此,GAN 的训练极其困难,想要让 GAN 正常运行,需要对模型架构和训练参数进行大量调整。

3.2.6 注意力机制

在计算能力有限的情况下,注意力机制(Attention Mechanism)作为一种资源分配方案,将有限的计算资源用来处理更重要的信息,是解决信息超载问题的主要手段。当用神经网络处理大量的输入信息时,也可以借鉴人脑的注意力机制,只选择一些关键的信息输入进行处理,以提高神经网络的效率。

在目前的神经网络模型中，我们可以将最大池化(Max Pooling)、门控(Gating)机制近似地看作自下而上的基于显著性的注意力机制。除此之外，自上而下的聚焦式注意力也是一种有效的信息选择方式。以阅读理解任务为例，给定一篇很长的文章，然后就此文章的内容进行提问。提出的问题只和段落中的一两个句子相关，其余部分都是无关的。为了减小神经网络的计算负担，只需要把相关的片段挑选出来让后续的神经网络处理，而不需要把所有文章内容都输入神经网络。

用 $\boldsymbol{x}=[x_1,x_2,\cdots,x_N]\in\mathbb{R}^{D\times N}$ 表示 N 组输入信息，其中 D 维向量 $\boldsymbol{x}_n\in\mathbb{R}^D$，$n\in[1,N]$表示一组输入信息。为了节省计算资源，不需要将所有信息都输入神经网络，只需要从 $\boldsymbol{x}$ 中选择一些和任务相关的信息。注意力机制的计算可以分为两步：一是在所有输入信息上计算注意力分布；二是根据注意力分布计算输入信息的加权平均。

定义 1 注意力分布：为了从 N 个输入向量$[x_1,x_2,\cdots,x_N]$中选择出和某个特定任务相关的信息，我们需要引入一个和任务相关的表示，称为查询向量(Query Vector)，并通过一个打分函数计算每个输入向量和查询向量之间的相关性。

给定一个和任务相关的查询向量 $\boldsymbol{q}$，我们用注意力变量 $z\in[1,N]$表示被选择信息的索引位置，即 $z=n$ 表示选择了第 n 个输入向量。为了方便计算，我们采用一种“软性”的信息选择机制。首先计算在给定 $\boldsymbol{q}$ 和 $\boldsymbol{x}$ 下，选择第 n 个输入向量的概率 α_n：

$$\begin{aligned}\alpha_n&=p(z=n\mid\boldsymbol{x},\boldsymbol{q})\\&=\mathrm{softmax}(s(x_n,\boldsymbol{q}))\\&=\frac{\exp(s(x_n,\boldsymbol{q}))}{\sum_{j=1}^{N}s(x_j,\boldsymbol{q})}\end{aligned}\tag{3-86}$$

其中，α_n 称为注意力分布(Attention Distribution)，$s(x_n,\boldsymbol{q})$为注意力打分函数，可以使用以下几种方式计算。

$$\text{加性模型：}s(\boldsymbol{x},\boldsymbol{q})=\boldsymbol{V}^{\mathrm{T}}\tanh(w\boldsymbol{x}+u\boldsymbol{q})\tag{3-87}$$

$$\text{点积模型：}s(\boldsymbol{x},\boldsymbol{q})=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{q}\tag{3-88}$$

$$\text{缩放点积模型：}s(\boldsymbol{x},\boldsymbol{q})=\frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{q}}{\sqrt{D}}\tag{3-89}$$

$$\text{双线性模型：}s(\boldsymbol{x},\boldsymbol{q})=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{q}\tag{3-90}$$

其中，$\boldsymbol{V}$、w、u、$\boldsymbol{W}$ 为可学习的参数，D 为输入向量的维度。理论上，加性模型和点积模型的复杂度差不多，但是点积模型在实现上可以更好地利用矩阵乘积，从而计算效率更高。

当输入向量的维度 D 比较高时，点积模型的值通常有比较大的方差，从而导致 softmax 函数的梯度会比较小。因此，缩放点积模型可以较好地解决这个问题。双线性模型是一种泛化的点积模型。假设式(3-90)中 $\boldsymbol{W}=\boldsymbol{U}^{\mathrm{T}}\boldsymbol{V}$，双线性模型可以写为 $s(\boldsymbol{x},\boldsymbol{q})=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{U}^{\mathrm{T}}\boldsymbol{V}\boldsymbol{q}=(\boldsymbol{U}\boldsymbol{x})^{\mathrm{T}}(\boldsymbol{V}\boldsymbol{q})$，即分别对 $\boldsymbol{x}$ 和 $\boldsymbol{q}$ 进行线性变换后计算点积。相比点积模型，双线性模型在计算相似度时引入了非对称性。

定义 2 加权平均：注意力分布 α_n 可以解释为在给定任务相关的查询 q 时，第 n 个输入向量受关注的程度。我们采用一种“软性”的信息选择机制对输入信息进行汇总，即

$$\mathrm{att}(X,q)=\sum_{n=1}^{N}\alpha_n x_n\tag{3-91}$$

式(3-91)称为软性注意力机制(Soft Attention Mechanism)。

3.2.7 Transformer 模型

循环神经网络和长短期记忆网络已经广泛应用于时序任务中,如文本预测、机器翻译、文章生成等。然而,它们面临的一个大问题就是如何记录长期依赖。为解决这个问题,一个名为 Transformer 的新框架应运而生。从那以后,Transformer 被应用到多个自然语言处理方向,到目前为止还未有新的架构能够将其替代。可以说它的出现是自然语言处理领域的突破,并为新的革命架构(如 BERT、GPT-3、T5 等)打下了理论基础。Transformer 完全依赖注意力机制,并摒弃了循环。它使用的是一种特殊的注意力机制,称为自注意力(Self-Attention)。

Transformer 模型由 6 层编码器和 6 层解码器构成。其中,编码器由 Self-Attention 和 Feed Forward Neural Network 构成;解码器由 Self-Attention、Encoder-DecoderAttention 和 Feed Forward Neural Network 构成。Transformer 模型架构图如图 3-16 所示。具体地,我们通过一个文本翻译实例了解 Transformer 是如何工作的。首先,向编码器输入一句话(原句),让其学习这句话的特征,再将特征作为输入传输给解码器。最后,此特征会通过解码器生成输出句(目标句)。假设我们需要将一个句子从英文翻译为法文,如图 3-17 所示。首先,我们将英文句子(原句)输入编码器进行编码。然后,编码器提取英文句子的特征并提供给解码器进行解码。最后,解码器通过特征完成法文句子(目标句)的翻译。

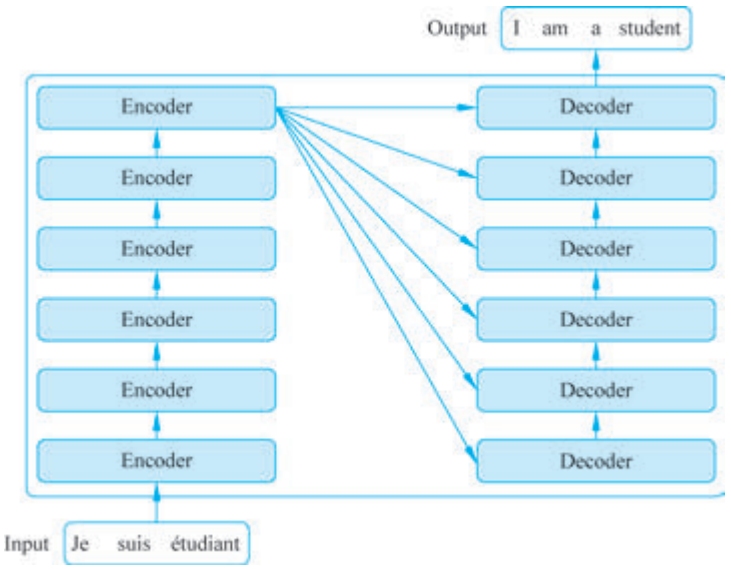

图 3-17 Transformer 模型架构

Transformer 中每个编码器的构造都是相同的,并且包含两部分:多头注意力层和前馈神经网络层。接下来我们通过一个例子快速理解自注意力机制。

A dog ate the food because it was hungry(一只狗吃了食物,因为它很饿)

例句中的代词 it(它)可以指代 dog(狗)或者 food(食物)。当读这段文字的时候,我们自然而然地认为 it 指代的是 dog,而不是 food。但是,当计算机模型在面对这两种选择时该如何决定呢?自注意力机制的出现有助于解决这个问题。

还是以上句为例,模型首先需要计算出单词 A 的特征值,其次计算 dog 的特征值,然后

计算 ate 的特征值，以此类推。当计算每个词的特征值时，模型需要遍历每个词与句子中其他词的关系。模型可以通过词与词之间的关系更好地理解当前词的意思。比如，当计算 it 的特征值时，模型会将 it 与句子中的其他词一一关联，以便更好地理解它的意思。

如图 3-18 所示，it 的特征值由它本身与句子中其他词的关系计算所得。通过关系连线，模型可以明确知道原句中 it 所指代的是 dog，而不是 food，这是因为 it 与 dog 的关系更紧密，关系连线相较于其他词也更粗。

图 3-18 自注意力示例

此外，为提升注意力机制的稳定性，Transformer 模型采用多头注意力机制对注意力层进行实现。

3.2.8 预训练模型

预训练模型并不是自然语言处理领域的“首创”技术。在计算机视觉（Computer Vision，CV）领域，以 ImageNet 为代表的大规模图像数据为图像识别、图像分割等任务提供了良好的数据基础。因此，在计算机视觉领域，通常会使用 ImageNet 进行一次预训练，让模型从海量图像中充分学习如何从图像中提取特征。然后，根据具体的目标任务，使用相应的领域数据精调，使模型进一步“靠近”目标任务的应用场景，起到领域适配和任务适配的作用。这好比人们在小学、初中和高中阶段会学习数学、语文、物理、化学和地理等基础知识，夯实基本功并构建基本的知识体系（预训练阶段）。而当人们步入大学后，将根据选择的专业（目标任务）学习某一领域更深层次的知识（精调阶段）。从以上介绍中可以看出，“预训练＋精调”模式在自然语言处理领域的兴起并非偶然现象。

由于自然语言处理的核心在于如何更好地建模语言，所以在自然语言处理领域中，预训练模型通常指代的是预训练语言模型。广义上的预训练语言模型可以泛指提前经过大规模数据训练的语言模型，包括早期的以 Word2Vec、GloVe 为代表的静态词向量模型，以及基于上下文建模的 CoVe、ELMo 等动态词向量模型。2018 年，以 GPT 和 BERT 为代表的基于深层 Transformer 的表示模型出现后，预训练语言模型这个词才真正被大家广泛熟知。因此，目前在自然语言处理领域中提到的预训练语言模型大多指此类模型。预训练语言模型的出现使得自然语言处理进入新的时代，也被认为是近些年来自然语言处理领域中的里程碑事件。

接下来介绍 GPT 和 BERT 预训练语言模型。GPT 是 OpenAI 公司于 2018 年提出的一种生成式预训练（Generative Pre-Training，GPT）模型，该模型用来提升自然语言理解任务的效果，正式将自然语言处理带入“预训练”时代——利用更大规模的文本数据及更深层的神经网络模型学习更丰富的文本语义表示。同时，GPT 的出现打破了自然语言处理各个任务之间的壁垒，使得搭建一个面向特定任务的自然语言处理模型不再需要了解非常多的任务背景，只需要根据任务的输入/输出形式应用这些预训练语言模型，就能达到一个不错的效果。因此，GPT 提出了“生成式预训练＋判别式任务精调”的自然语言处理新范式，使得自然语言处理模型的搭建变得不再复杂。

◇ **生成式预训练**：在大规模文本数据上训练一个高容量的语言模型，从而学习更加丰

富的上下文信息；

◇ **判别式任务精调**：将预训练好的模型适配到下游任务中，并使用有标注数据学习判别式任务。

BERT(Bidirectional Encoder Representation from Transformers)是由 Devlin 等于 2018 年提出的基于深层 Transformer 的预训练语言模型。BERT 不仅充分利用大规模无标注文本挖掘其中丰富的语义信息，同时还进一步加深了自然语言处理模型的深度。

3.3 模型评价

为了衡量一个机器学习模型的好坏，需要给定一个测试集，用模型对测试集中的每一个样本进行预测，并根据预测结果计算评价分数。

3.3.1 正确率

定义 1 正确率(Accuracy)：是最直观的评价指标之一，用于衡量模型在整个数据集上的表现，它是预测正确的样本数量与总样本数量之比，计算公式为

$$\text{Accuracy}=\frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \tag{3-92}$$

模型中可能存在的预测结果类型见表 3-2。

表 3-2 模型中可能存在的预测结果类型

简　称		含　义
TP	真正例	模型将正类别样本预测为正类别的数量
TN	真负例	模型将负类别样本预测为负类别的数量
FP	假正例	模型将负类别样本预测为正类别的数量
FN	假负例	模型将正类别样本预测为负类别的数量

正确率适用于样本类别分布均衡的场景，但在类别不平衡(即某一类样本数量明显多于其他类别)时，正确率可能不是一个很好的评估指标，可能会产生误导性结果，因为它可能会高估模型的性能。因此，在处理类别不平衡的数据时，正确率不应作为唯一的评估指标。

正确率具有以下两个局限：①类别不平衡问题，在高度不平衡的数据集中，即某一类别的样本数量显著多于其他类别时，正确率可能会失去其意义。例如，如果 99%的样本是负类，模型始终预测为负类仍然可以达到 99%的正确率，但这并不意味着模型是好的。②无法反映分类的细节，正确率不能提供有关误分类样本数量的详细信息，也不能揭示模型在各类样本上的表现差异。

为此，可通过以下两方面对正确率指标进行改进：①基于类别加权的正确率，通过为每个类别分配不同的权重，可以减轻类别不平衡对正确率的影响。②多指标综合使用，评估模型时，应结合其他指标(如精确率、召回率、F1 值等)全面衡量模型性能。

定义 2 误差率：表示模型预测错误的样本数量与总样本数量之比。其计算公式为

$$\text{Error Rate}=\frac{\text{FP}+\text{FN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \tag{3-93}$$

【例 3.4】 计算正确率。

在一场疾病预测的分类任务中，有 1000 个测试样本，其中 950 个为健康（负类），50 个为患病（正类）。模型预测的结果如下。

TP=40（预测为患病且实际为患病）
TN=940（预测为健康且实际为健康）
FP=10（预测为患病但实际为健康）
FN=10（预测为健康但实际为患病）

根据正确率计算公式：$\text{Accuracy}=\dfrac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$，可以得出正确率为 $\dfrac{40+940}{40+940+10+10}=\dfrac{980}{1000}=0.98$。

根据误差率计算公式：$\text{Error Rate}=\dfrac{\text{FP}+\text{FN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$，可以得出误差率为 $\dfrac{10+10}{40+940+10+10}=\dfrac{20}{1000}=0.02$。

即模型的正确率为 98%，误差率为 2%。然而，由于数据集不平衡，正确率高并不一定意味着模型表现良好。因此，应结合其他指标进行全面评价。

定义 3 Kappa 系数（Cohen's Kappa）：是一种用于衡量分类模型性能的统计指标，考虑了随机猜测的影响。Kappa 系数的取值范围为[−1,1]，其中 1 表示完全一致，0 表示随机猜测，负值表示系统性的不一致。计算公式为

$$k=\frac{p_o-p_e}{1-p_e} \tag{3-94}$$

其中，p_o 是观察到的一致性（即正确率），p_e 是随机猜测的预期一致性，计算方法基于类别的边际分布。

【例 3.5】 Kappa 系数计算。

假设有一个数据集，其中正类和负类的比例相等，且模型的正确率为 0.8。如果随机猜测的预期一致性为 0.5，则 Kappa 系数计算如下。

$$k=\frac{0.8-0.5}{1-0.5}=\frac{0.3}{0.5}=0.6$$

一般而言，可分为 5 组表示不同级别的一致性：0.0～0.20 表示极低的一致性（slight）、0.21～0.40 表示一般的一致性（fair）、0.41～0.60 表示中等的一致性（moderate）、0.61～0.80 表示高度的一致性（substantial）和 0.81～1 表示几乎完全一致（almost perfect）。

例 3.5 的 Kappa 值表示模型性能比随机猜测好得多，但仍有改进空间。

3.3.2 精确率

定义 4 精确率（Precision）：又称查准率，是机器学习和数据分类任务中常用的评估指标之一，用于衡量模型在预测为正类别的样本中，真正为正类别的样本所占的比例。

精确率用于解决以下问题：在所有预测为正类别的样本中，有多少是真正的正类别。其计算公式为

$$\text{Precision}=\frac{\text{TP}}{\text{TP}+\text{FP}} \tag{3-95}$$

精确率特别适用于在假正例代价较高的场景下使用，例如，在垃圾邮件检测中，高精确率意味着大部分标记为垃圾邮件的邮件确实是垃圾邮件，而错误地将正常邮件分类为垃圾邮件会带来较大影响。高精确率意味着模型在预测为正样本时，错误的次数较少。

因此，研究者一般会采用以下策略提升模型的精确率：①通过提高分类阈值，可以减少假阳性数量，从而提高精确率；②使用精度调整技术，如正则化、特征选择和样本加权等技术，可以优化模型，以提高精确率。

虽然精确率一定程度上能反映模型的性能，但精确率关注的是假正例，而对假负例不敏感。因此，仅依靠精确率评估模型可能无法全面反映模型在实际应用中的表现。为此，一些研究者在进行模型评价时通常会结合精确率和召回率对模型性能进行评价，以提供对模型性能更全面的评估。通常，提高精确率往往会降低召回率，因而在一些应用中需在二者之间找到平衡。

定义 5　加权精确率：加权精确率考虑了每个类别的重要性，通过为每个类别分配不同的权重来计算整体精确率。其计算公式为

$$\text{Weight Precision}=\sum_{i=1}^{n}w_i\times\text{Precision}_i \tag{3-96}$$

其中，w_i 为类别 i 的权重，Precision_i 为类别 i 的精确率。

【例 3.6】 计算精确率。

在一个金融欺诈检测任务中，有 1000 个交易记录，其中 950 个为正常交易(负类)，50 个为欺诈交易(正类)。模型预测的结果如下。

TP=40(预测为欺诈且实际为欺诈)
FP=20(预测为欺诈但实际为正常交易)

根据精确率计算公式：$\text{Precision}=\frac{\text{TP}}{\text{TP}+\text{FP}}$，可以得出精确率为 $\frac{40}{40+20}=\frac{40}{60}=0.67$，即模型的精确率为 67%。

3.3.3　召回率

定义 6　召回率(Recall)：也称为查全率或灵敏度，是衡量分类模型能力的指标之一，用于评估模型在所有实际为正类别的样本中，有多少被正确识别为正类别。它反映了模型对正类别的覆盖能力。召回率越高，越说明模型能够更全面地检测出正类别样本，但可能导致更多的假正例。其计算公式为

$$\text{Recall}=\frac{\text{TP}}{\text{TP}+\text{FN}} \tag{3-97}$$

召回率的核心思想是衡量模型在识别正例方面的表现，它强调了模型对实际为正类别的样本的识别能力。对于关心捕获正类样本的任务，如癌症检测任务，未能识别出患病个体的后果可能非常严重。因此，在类似于癌症检测的任务上通常需要提高模型的召回率。然而，提升模型的召回率通常会降低其精确率，因此模型需在两者之间找到平衡。

如上所述，召回率通常用于重视假阴性代价的场景。这些场景中，如医疗诊断或安全监控，未能识别出正类(如病人或危险行为)的代价可能非常高。因此，召回率是这些场景中的关键指标。

为提升模型的召回率，研究者通常采用以下两种策略：①调整分类阈值，通过降低分类阈值，可以增加模型预测为正类的样本数，从而提高召回率；②数据增强，通过增加数据集中的正类样本，模型可以学到更多正类的特征，从而提高召回率。

然而，召回率的提升通常会导致更多的假阳性，从而降低模型的精确率。因此，仅依靠召回率评估模型可能会导致高误报率。

例如，在一个医疗诊断系统中，假设我们要检测某种罕见但致命的疾病。由于这种疾病的发病率非常低，大多数样本都是健康的(负类)，而少部分是患者(正类)。假设系统的召回率很低，即很多患者未能被诊断出来(FN 较多)，这意味着这些患者可能没有及时得到治疗，增加了病情恶化，甚至死亡的风险。

在这种场景下，低召回率的后果非常严重。因此，在医疗诊断中，特别是在处理危及生命的疾病时，召回率比精确率更重要，必须尽量提高召回率，以确保患者得到及时诊断。

定义 7　加权召回率(Weighted Recall)：加权召回率考虑了每个类别的重要性，通过为每个类别分配不同的权重计算整体召回率。加权召回率能够更好地反映在类别不平衡数据中的模型性能。其计算公式为

$$\text{Weighted Recall}=\sum_{i=1}^{n} w_i \times \text{Recall}_i \tag{3-98}$$

其中，w_i 为类别 i 的权重，Recall_i 为类别 i 的召回率。

加权召回率通常应用在类别不平衡的场景中，其能够更准确地反映模型在不同类别上的表现。例如，在医疗诊断任务中，不同疾病类别的重要性不同，因此可以为更严重的疾病分配更高的权重。

定义 8　平均召回率(Average Recall)：是多个类别召回率的平均值，适用于多分类问题。在多分类问题中，如图像分类任务，可以使用平均召回率评价模型在所有类别上的表现。平均召回率对每个类别的召回率求平均值，得出模型的整体召回率。

【例 3.7】 计算召回率。

在一场癌症诊断任务中，有 100 个测试样本，其中 90 个为健康(负类)，10 个为癌症患者(正类)。模型预测的结果如下。

TP=8(预测为癌症且实际为癌症)
FN=2(预测为健康且实际为癌症)

根据召回率计算公式：$\text{Recall}=\dfrac{\text{TP}}{\text{TP}+\text{FN}}=\dfrac{8}{8+2}=0.8$，可以得出召回率为 80%。这意味着，模型能够正确识别出 80%的癌症患者，但仍然漏掉了 20%的患者(假负例)。

3.3.4　F1 值

定义 9　F1 值(F1-Score)：机器学习和数据分类任务中常用的综合性评估指标，同时兼顾了分类模型的精确率和召回率。F1 值是精确率和召回率的调和平均，用于衡量模型在正类别的识别和预测能力的综合表现。其计算公式为

$$\text{F1}=2\times\frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \tag{3-99}$$

F1 值综合考虑了模型的精确率和召回率，因此可以一定程度上弥补精确率和召回率单独使用时的不足，特别适合于类不平衡问题。在业务决策中。F1 值通常更易于解释，并能更直观地反映模型的实际效果。F1 值越高，表示模型识别和预测正类别的综合表现越好。

与准确率相比，在类别分布均衡的情况下，准确率和 F1 值均能反映出模型的好坏。但在类别分布不均衡的情况下，F1 值更能反映出模型的真实性能，而不仅是整体的预测准确性。相对于精确率和召回率来说，F1 值是精确率和召回率的调和平均，因此该值偏向两者的较低值。如果精确率高但召回率低，F1 值也会较低，反之亦然，这反映出模型在一方面表现不佳。

综上所述，F1 值能够在正负类样本不平衡时，提供更公平的评估指标，同时兼顾了精确率和召回率，适用于完成追求两者兼顾的任务。但在精确率或召回率一方表现极差时，F1 值可能会被拉低，难以单独评估模型优劣。因此，对于部分任务，如果需要单方面优化精确率或召回率，F1 值并非最优选择。

【例 3.8】 计算 F1 值。

在金融欺诈检测中，假设我们使用一个模型识别欺诈交易，得到以下结果：

TP=25(预测为欺诈且实际为欺诈)
FP=5(预测为欺诈但实际为正常交易)
FN=15(预测为正常交易但实际为欺诈)

计算得出：

$$\text{精确率(Precision)} = 25/(25+5) \approx 0.833$$

$$\text{召回率(Recall)} = 25/(25+15) = 0.625$$

根据计算公式：$F1 = 2 \times \dfrac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \times \dfrac{0.833 \times 0.625}{0.833 + 0.625} \approx 0.714$，可以得出尽管模型的精确率较高，但由于召回率偏低，F1 值为 0.714，显示出在这一任务中模型在漏检方面还有较大的改进空间。

3.3.5 混淆矩阵

定义 10 混淆矩阵(Confusion Matrix)：是用于总结分类模型性能的工具，通过一个矩阵显示预测结果与实际情况的对比。矩阵中的元素包括 TP、TN、FP、FN，它提供了模型在每个类别上的详细表现。混淆矩阵不仅可以用于二分类问题，还可以扩展至多分类问题，帮助识别模型在特定类别上的误差来源。

一个二分类问题的混淆矩阵通常是一个 2×2 的矩阵，如表 3-3 所示。

表 3-3 混淆矩阵

	预测为正类(Positive)	预测为负类(Negative)
实际为正类(Positive)	True Positive(TP)	False Negative(FN)
实际为负类(Negative)	False Positive(FP)	True Negative(TN)

通过混淆矩阵，可以直观地看到模型在哪些类别上表现良好，在哪些类别上容易出错。它还为计算其他评价指标(如精确率、召回率和特异性)提供了基础数据，是理解和改进模型的重要工具。

但混淆矩存在以下两个问题：①在处理多类别分类任务时，混淆矩阵可能变得过于复杂，难以直观地分析模型性能。②虽然混淆矩阵能展示详细的分类信息，但它并不提供单一的性能指标，仍需结合其他指标（如准确率、精确率和召回率）全面评估模型。

定义 11　多分类混淆矩阵：对于多分类问题，混淆矩阵的维度不再是 2×2，而是 $N \times N$，其中 N 是类别的数量。每个元素(i,j) 表示实际为类别 i 的样本被分类器预测为类别 j 的数量。多分类混淆矩阵可以帮助我们识别分类器在哪些类别上表现较好，哪些类别容易混淆。

混淆矩阵常应用在以下两种场景：①混淆矩阵能帮助识别模型在哪些类别上表现不佳，进而指导模型进一步优化。②在多类别分类问题中，混淆矩阵可以展示每个类别的误分类情况，帮助理解模型的细节表现。

【例 3.9】 假设有一个三分类问题，类别分别为 A、B 和 C。模型的预测结果如下。

	预测为 A	预测为 B	预测为 C
实际为 A	50	20	8
实际为 B	5	40	5
实际为 C	10	3	37

从矩阵可以看出：

（1）类别 A 被正确分类为 A 的样本有 50 个，但有 20 个样本分类为 B，8 个样本被误分类为 C。

（2）类别 B 的分类表现较好，但各有 5 个样本被误分类为 A 和 C。

（3）类别 C 的表现稍差，有 10 个样本被误分类为 A、3 个样本被误分类为 B。

定义 12　特异度（Specificity）：表示模型正确识别负类的能力，计算公式为 Specificity＝TN /（TN＋FP）。特异度与召回率相辅相成，共同描述模型的全面性能。

定义 13　平衡精度（Balanced Accuracy）：是正确率在类不平衡问题上的一种改进版本，通过计算所有类别精度的平均值反映模型的整体表现。

3.3.6　ROC 曲线

定义 14　ROC 曲线（ROC Curve）：是一种绘制真正例率（TPR）随着假正例率（FPR）变化的图形，ROC 曲线用于评估分类模型的整体表现，展示了不同阈值下的真正例率与假正例率之间的权衡。

其中，TPR（True Positive Rate）表示召回率，其计算公式为 TPR＝TP/（TP＋FN），表示模型正确识别正例的比例。FPR（False Positive Rate）的计算公式为 FPR＝FP/（FP＋TN），表示模型错误识别负例为正例的比例。

ROC 曲线基于真正例率和假正例率，展示了在不同分类阈值下，模型在识别正例和负例方面的性能。ROC 曲线上的点越靠近左上角，表示模型性能越好。在 ROC 曲线上，横轴表示 FPR，纵轴表示 TPR。AUC 是 ROC 曲线下的面积，它等于 ROC 曲线与横轴之间的面积，可用来比较不同模型的性能，AUC 的取值范围在 0～1。一般来说，AUC 越大，表示模型性能越好。因此，理想的 ROC 曲线应该尽量靠近左上角，这表明模型具有较高的 TPR 和较低的 FPR。对于随机猜测的模型，其 ROC 曲线是一条对角线，表示 TPR 与 FPR 相等。此外，曲线的斜率反映了模型在不同阈值下的表现。当曲线的斜率较大时，意味着模型

在该阈值下的性能较好。

由上可知,ROC 曲线和 AUC 曲线是用于评估二分类模型性能的重要工具,它们基于真正例率(True Positive Rate,召回率)和假正例率(False Positive Rate)描述模型在不同阈值下的性能表现。ROC 曲线常用于以下两种情况:一是模型对比。ROC 曲线适用于比较多个模型的分类能力,尤其是在数据集较为平衡的情况下。二是阈值调整。通过观察 ROC 曲线,用户可以选择一个合理的阈值,以在精确率和召回率之间取得最佳平衡。ROC 曲线特别适合在不平衡数据集中使用,因为它能够在各种阈值下提供对模型性能的全面评价。然而,ROC 曲线的缺点在于,当数据集高度不平衡时,曲线的形状可能导致误导性解释。因此,ROC 曲线配合 AUC 曲线使用时效果最佳。

【例 3.10】 计算 TPR 和 FPR。

假设有一个疾病检测的分类模型,目标是预测某人是否患有一种疾病。数据集包含 1000 个样本,其中 100 个样本为正类(患病),900 个样本为负类(健康)。模型对每个样本生成一个预测概率,以下是几组阈值下的 TPR 和 FPR 计算。

(1) 阈值=0.9:只有预测概率大于 0.9 的样本被分类为患病。

$$TPR=0.40,\quad FPR=0.01$$

(2) 阈值=0.7:预测概率大于 0.7 的样本被分类为患病。

$$TPR=0.65,\quad FPR=0.05$$

(3) 阈值=0.5:预测概率大于 0.5 的样本被分类为患病。

$$TPR=0.85,\quad FPR=0.15$$

(4) 阈值=0.3:预测概率大于 0.3 的样本被分类为患病。

$$TPR=0.95,\quad FPR=0.30$$

基于这些阈值可以绘制 ROC 曲线。假设 ROC 曲线的 AUC 为 0.85,说明模型在区分患病和健康样本上表现良好。比较阈值为 0.9 和 0.3 的点,可以看出降低阈值会增加 TPR,但同时也增加了 FPR。这意味着,在实际应用中,需要根据实际情况选择合适的阈值来平衡 TPR 和 FPR。

3.3.7 AUC 曲线

定义 15 AUC(Area Under the Curve)曲线: AUC 表示 ROC 曲线下的面积,取值范围在 0~1。AUC 的值越接近 1,说明模型区分正负样本的能力越强。计算从(0,0)到(1,1)整个 ROC 曲线以下的整个二维面积,用于衡量二分类问题其机器学习算法性能的泛化能力。其另一种解读方式可以是模型将某个随机正类别样本排列在某个随机负类别样本之上的概率。

(1) AUC=1:完美分类,模型能够完美区分正负类样本。

(2) 0.5<AUC<1:模型有一定的区分能力。

(3) AUC=0.5:模型相当于随机猜测。

(4) AUC<0.5:模型表现很差,甚至不如随机猜测。

AUC 是 ROC 曲线下面积的缩写,用于量化模型的整体性能。AUC 值不依赖具体的阈值,因此它可以作为不同模型间性能的一个综合比较指标。AUC 在处理类别不均衡时依然保持稳定,因此在这种情况下常被使用。

AUC 指标具有稳健性高和排序能力强的优势。AUC 值能够综合评估模型在不同决策阈值下的性能，因此比单一的准确率更稳健。此外，AUC 擅长衡量模型对样本的排序能力，即预测的分数如何反映出正负样本的实际分布情况。

综上所述，AUC 指标常用于模型选择和数据不平衡等应用场景。在实际项目中，AUC 值常用于选择和优化分类模型，特别是在需要比较多个模型的情况下。此外，虽然 AUC 值在类别不平衡的数据集中仍具有意义，但需要注意其局限性。当正负类分布极为不平衡时，AUC 值可能会高估模型的实际效果。

【例 3.11】 假设有一个电子邮件分类系统，目标是将邮件分为“垃圾邮件”和“正常邮件”。数据集包含 1000 封邮件，其中 900 封为正常邮件（负类），100 封为垃圾邮件（正类）。分类模型根据每封邮件的特征输出一个介于 0 到 1 的概率值，表示该邮件是垃圾邮件的可能性。

通过改变分类阈值，可以得到不同的 TPR 和 FPR，然后绘制出 ROC 曲线。如果计算得出模型的 AUC 为 0.9，这意味着模型在区分垃圾邮件和正常邮件方面表现非常好。

解释 1：AUC=0.9 表示随机选取一封垃圾邮件和一封正常邮件，该模型有 90% 的概率将垃圾邮件的预测分数排在正常邮件之前。

解释 2：如果有多个模型可以选择，AUC=0.9 的模型显然优于 AUC=0.7 的模型。

3.3.8 BLEU

定义 16 BLEU： 是机器翻译领域中的一种评估指标，用于度量机器翻译输出与多个参考翻译的相似性。BLEU 的核心思想是通过比较 N-gram 的一致性评估翻译质量。它既考虑了词序的一致性，也关注了翻译的完整性。BLEU 分数介于 0 到 1，分数越高，表示翻译质量越好。然而，BLEU 也存在一些局限性，例如无法完全捕捉语义上的准确性。

BLEU 计算步骤如下。

（1）N-gram 精确率：计算机器翻译文本和参考翻译文本中每个 N-gram 的匹配比例。

（2）BP(Brevity Penalty)：对过短的翻译结果进行惩罚，防止模型生成过短的文本，以提高精确率。

（3）加权平均：对不同 N-gram 的精确率取加权平均，常见的 BLEU 实现使用 1-gram 到 4-gram 的加权平均。

计算公式如下。

$$\mathrm{BLEU}=\mathrm{BP}\times\exp\left(\sum_{n=1}^{N}w_n\cdot\log(p_n)\right) \tag{3-100}$$

其中，BP 是长度惩罚因子：

$$\begin{cases}1, & \text{length}\geqslant\text{referencelength}\\ \exp\left(1-\dfrac{\text{referencelength}}{\text{length}}\right), & \text{其他}\end{cases}$$

p_n 是 N-gram 精确率：$p_n=\dfrac{\text{Number of matched N-gram}}{\text{Number of generated N-gram}}$，

w_n 是 N-gram 的权重，通常与所有 N-gram 的权重相等。

BLEU 的优势在于自动化评估和通用性。BLEU 分数可以通过编程直接计算，适合大

规模自动化评估。此外,BLEU 不仅适用于机器翻译,还广泛应用于其他文本生成任务,如摘要生成和文本复述。然而,BLEU 主要基于字面匹配,忽视了文本的语义和上下文,因此在一些情况下可能无法准确反映翻译质量。另外,BLEU 的表现高度依赖参考文本的质量和多样性。如果参考文本不够丰富,BLEU 分数可能会产生偏差。

3.3.9 ROUGE

定义 17 ROUGE:是文本摘要评估中常用的指标,特别是在自然语言处理任务中。

与 BLEU 不同,ROUGE 更多关注的是模型生成内容的召回率,通常衡量参考摘要与生成摘要之间的 N-gram、词语,甚至句子级别的重叠程度。ROUGE 包括多种变体,如 ROUGE-N、ROUGE-L、ROUGE-W 等,用于不同类型的文本对比。ROUGE 指标常被用来评估生成模型的覆盖率,特别是在自动文本生成任务中。

ROUGE-N:ROUGE-N 通过 N-gram 重叠评估生成文本的质量,类似于 BLEU 中的 N-gram 计算,但更注重召回率。

ROUGE-L:ROUGE-L 关注生成文本与参考文本之间最长公共子序列(Longest Common Subsequence,LCS)的匹配情况,用以评估文本整体的语义连贯性。

ROUGE 的优势在于适用于长文本评估和召回率导向,即 ROUGE 特别适合评估长文本生成任务,如自动摘要和问答系统,因为它能够捕捉生成文本与参考文本在句子级别的匹配度。此外,由于 ROUGE 更关注生成内容的全面性,因此在某些任务中能比 BLEU 更好地反映模型性能。但 ROUGE 与 BLEU 类似,虽然 ROUGE 和 BLEU 在理念上有所不同,但两者在实际应用中可能会产生类似的结果,尤其是在短文本生成任务中。另外,ROUGE 的表现也依赖参考文本的质量和数量。在参考文本较少的情况下,ROUGE 分数可能无法全面反映模型性能。

3.4 本章小结

本章从传统机器学习模型、深度学习模型和模型评价指标等方面介绍了模型基础。传统机器学习模型包含 Logistic Regression、支持向量机、最大熵模型、随机森林、条件随机场、集成学习;深度学习模型包含感知机、前馈神经网络、递归神经网络、卷积神经网络、生成式对抗网络、注意力机制、Transformer 模型、预训练模型;模型评价指标包含正确率、准确率、召回率、F1 值、混淆矩阵、ROC 曲线、AUC 曲线、BLEU、ROUGE。

习题 3

(1)(多选题)下列关于最大似然估计(Maximum Likelihood Estimation,MLE),说法正确的是(　　)。

A. MLE 可能并不存在

B. MLE 总是存在

C. 如果 MLE 存在,那么它的解可能不是唯一的

D. 如果 MLE 存在,那么它的解一定是唯一的

(2)(多选题)机器学习的模块有(　　)。

A. 损失函数　　B. 模型类型　　C. 学习算法　　D. 算法复杂度

(3)(多选题)常用激活函数有(　　)。

A. sigmoid　　B. tanh　　C. ReLU　　D. GeLU

(4)(多选题)k-最近邻算法(k-NN)中相似度度量计算主要有(　　)。

A. Euclidean 距离　　B. 加权 Euclidean 距离

C. 余弦相似度　　D. 正弦相似度

(5)(多选题)k-最近邻算法中由投票策略(　　)确定标签。

A. 多数投票　　B. 加权投票　　C. 少数投票　　D. 非投票

(6)(多选题)在深度学习中,网络层数增多会伴随的问题有(　　)。

A. 计算资源的消耗(GPU)

B. 模型容易过拟合(Dropout)

C. 梯度消失/梯度爆炸问题产生(批量归一化(BN))

D. 退化(degradation)问题

(7)(多选题)池化层的作用主要有(　　)。

A. 可以扩大感知野

B. 实现非线性

C. 降维

D. 可以实现不变性:平移不变性、旋转不变性、尺度不变性

(8) BLEU 是如今在评估机器翻译质量时常用的一个指标。

对于下述句对,计算 bigram BLEU 的得分:

Candidate: the cat the the the the
Reference: the cat is now on the mat

(9) 假设二元分类的输出是概率值,一般地,设定输出概率大于或等于 0.5,则预测为正类;若输出概率小于 0.5,则预测为负类。那么,如果将阈值 0.5 提高,例如提高到 0.6,大于或等于 0.6 的阈值才预测为正类,则精确率(Precision)和召回率(Recall)会发生什么变化?(　　)

A. 精确率增加或者不变　　B. 精确率减小

C. 召回率减小或者不变　　D. 召回率增大

(10) 数据库中有 500 个文档,其中 50 个符合问题的定义。系统检索了 75 个文档,其中只有 45 个符合定义的问题。请计算精确率和召回率。

(11) 传统机器学习算法有哪些?这些算法的核心思想是什么?

(12) 有哪些深度学习模型?常用的深度学习模型有哪些?

(13) 简要分析现有评价指标的作用。

(14) 分类模型评价的常用指标有哪些?