01 一些往年题的汇总

一些必考概念

独立同分布

独立同分布是机器学习模型有效性的假设性前提：在满足独立同分布的前提下，学习到的机器学习模型才是有效的
“独立”:针对样本来源而言的。如果样本集（训练集、测试集）中的任意两个样本之间，都是不相关的，则样本的获得或者生成满足独立性条件
“**同分布”**：在表示样本的特征确定以后，训练样本集的分布是具体而确定的，称之为“基于训练样本集的分布”；样本全集（通常都是无法获取的，是个理论概念），如果能够获取的话，同样对应着一个具体而确定的分布，称之为“基于样本全集的分布（实际上无法获得）”；
严格意义上的同分布：基于训练样本集的分布与基于样本全集的分布完全相同

在机器学习的实践中，绝对意义上的独立、同分布假设是很难做到的。无法确保严格的独立、同分布，是影响模型泛化性能的根本原因，也是机器学习模型在开放场景应用的最大挑战。

机器学习

机器学习就是让计算机从数据中自动学习规律，并利用学到的规律处理新问题。

简答题 8分x9

（不仅要说明机器学习范式的基本思想，还要复述上课讲过的内容，并简述基本思路，例如：半监督学习方法有自训练与协同训练，需简述两种方法的基本思想。遗漏章节重要内容会酌情扣分）

1、什么是半监督学习
2、什么是无监督学习
3、什么是迁移学习
4、什么是度量学习
5、什么是在线学习
6、什么是元学习
7、什么是联邦学习
8、什么是多示例多标记学习

02 半监督学习

> Semi-Supervised Learning

概念

- 无监督学习（Unsupervised Learning）
  - 训练数据： X=(x1,...,xn)
  - 目标：在数据X中找到有意义的结构，发现潜在的有价值的信息
  - 典型应用：聚类、降维、离群点检测等
- 监督学习（Supervised Learning）
  - 训练数据：X={(x1,y1),...,(xn,yn)}
  - 目标：学习一个从特征到输出的映射函数，该函数可以对新的未标记的数据进行相对准确和可靠的输出预测
  - 应用：分类、回归
- 半监督学习（Semi-Supervised Learning，SSL）
  - 应用场景：在某些领域，给样本打标记一般都很费时，而且成本很高
  - 一种介于监督学习和无监督学习之间的一种学习范式
  - 目标：使用标记和未标记的数据来学习更好的学习器，这个学习器比仅使用标记数据训练得到的学习器或仅使用无标记数据训练得到的学习器更好
  - 形式化表达：
    - 输入实例 x，标记 y
    - 实例空间 $\mathcal{X}$，标记空间 $\mathcal{Y}$，其中 $x \in \mathcal{X}$，$y \in \mathcal{Y}$
    - 有标记的数据集 $D_l = \{(x_{1:l}, y_{1:l})\}$，无标记的数据集 $D_u = \{x_{l+1:n}\}$
    - 训练阶段: 通过 D_l 和 D_u 在去获得一个映射 $f : \mathcal{X} \mapsto \mathcal{Y}$
    - 测试阶段: 利用 f 来预测 $D_t = \{(x_{n+1:}, y_{n+1:})\}$

      $D_t = \{(x_{n+1:}, y_{n+1:})\}$ 是新数据，未参与训练

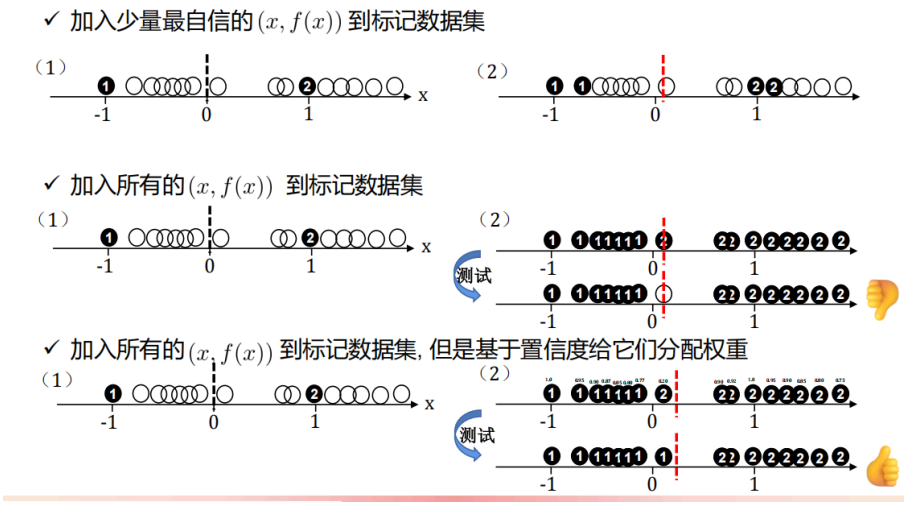
    在很多现实应用当中，标记数据很难获得或标记代价昂贵，而无标记数据的获得容易活廉价，因此： 通常 $l \ll n$
  - 假设/半监督学习何时起作用
    - 所有的无标记数据必须属于有标记数据的某一类
      - 模型同时利用有标签数据计算监督损失、利用无标签数据计算无监督损失，两部分共同组成半监督学习的优化目标，从而实现利用少量标注数据和大量未标注数据提升模型性能
    - 光滑性假设
      - 如果一个高密度区域的两个点x1，x2很接近，那么相应的输出y1，y2也应该很接近

- 聚类假设
  - 如果两个点在同一个簇中，那么它们很可能属于同一类
- 流形假设
  - 高维数据大致位于低维流形上
    - 高维数据并不是杂乱无章地分布在整个空间中，而是集中在某些低维结构上

代表性方法

自训练（Self-training）

1. 假设：具有高置信度的预测是正确的
2. 自训练算法：
   1. 基于 (Xl,Yl) 来训练f
   2. 利用f来预测x∈Xu
   3. 将 (x,f(x))加入有标记数据
   4. 重复
3. 变种：



第一种：只加入少量最自信的样本

- 模型只挑自己最有把握的无标签样本加入训练集

第二种：加入所有伪标签样本👎

- 加入所有的 (x,f(x))到标记数据集
- 模型把所有无标签样本都预测一个标签，然后全部加入训练集

第三种：加入所有伪标签样本，但按置信度加权👍

- 加入所有的 (x,f(x))到标记数据集，但是基于置信度给它们分配权重
- 既能充分利用无标签数据，又能降低错误伪标签对模型的负面影响

4. 优缺点

优点：

- 很容易理解
- 框架性思路：Self-Training+各种机器学习方法

- Self-Training 本身不是一个具体分类器，它只是一个训练框架；真正负责学习分类规律的是里面用的机器学习模型

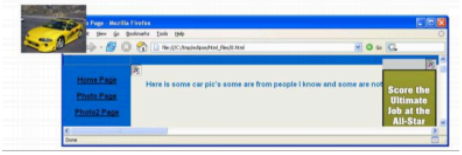
  缺点：

- 错误的伪标记会影响模型训练效果，尤其是训练过程早期的错误
- 伪标记带来的问题会在训练过程中“放大”

协同训练（Co-training）

1. 思路

   ✓ 汽车作为一种对象，可以用图像表达，也可以用文字表达，图像、文字可看作表达汽车这一对象的两个不同的视图

   - 每一个实例由两组特征表示 $x=[x^{(1)};x^{(2)}]$
   - $x^{(1)}$:是图片数据对应的特征，$x^{(2)}$:是文本数据对应的特征 。 两种特征是天然分离的。

   可以用多种形式或者来源的数据表达同一个对象，不同形式或来源的数据是天然分离的、相互独立的

2. 算法步骤

   协同训练算法: 视图1 视图2

   ① 训练两个分类器: $f^{(1)}$ from $\left(X_l^{(1)}, Y_l\right)$, $f^{(2)}$ from $\left(X_l^{(2)}, Y_l\right)$.

   ② 基于 $f^{(1)}$ 和 $f^{(2)}$ 分别对 X_u 分类.

   ③ 加入 $f^{(1)}$ 的k个最自信的 $(x, f^{(1)}(x))$ 到 $f^{(2)}$ 的有标记集合中

   ④ 加入 $f^{(2)}$ 的k个最自信的 $(x, f^{(2)}(x))$ 到 $f^{(1)}$ 的有标记集合中

   ⑤ 重复。

3. 假设

   $x=[x^{(1)};x^{(2)}]$

   单独使用特征$x^{(1)}$或者$x^{(2)}$已经足够去训练好一个好的分类器

   特征$x^{(1)}$和$x^{(2)}$在分类时相互独立

4. 优缺点

   优点：

   - 好理解
   - 框架性思路：Co-Training+各种机器学习方法
   - 和自训练方法比，协同训练对错误的伪标记不太敏感

   缺点：

   - 在有些应用或者任务中，未必满足或者未必能创造出多视图条件

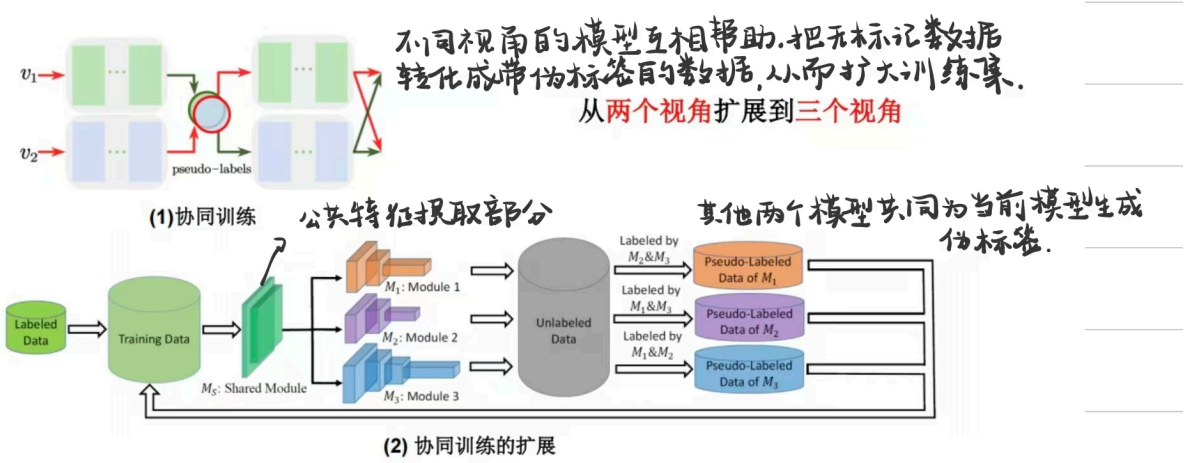
5. 协同训练风格

   只要目标是**训练出两个不同的分类器**，让它们互相提供信息，就可以看作协同训练风格

   现实中不一定总能找到严格意义上的两个天然视图，所以可以把 Co-training 的思想扩展一下：

- 特征划分：将所有特征划分为两个特征子集，将两个不同的特征子集视为不同的视图，进行协同训练
- 数据集划分：将有标记数据集划分为两个子集，将划分后的两个有标记数据子集视为不同的视图，进行协同训练
- 不划分数据集、不划分特征，但训练两个不同的机器学习模型

6. 视角拓展

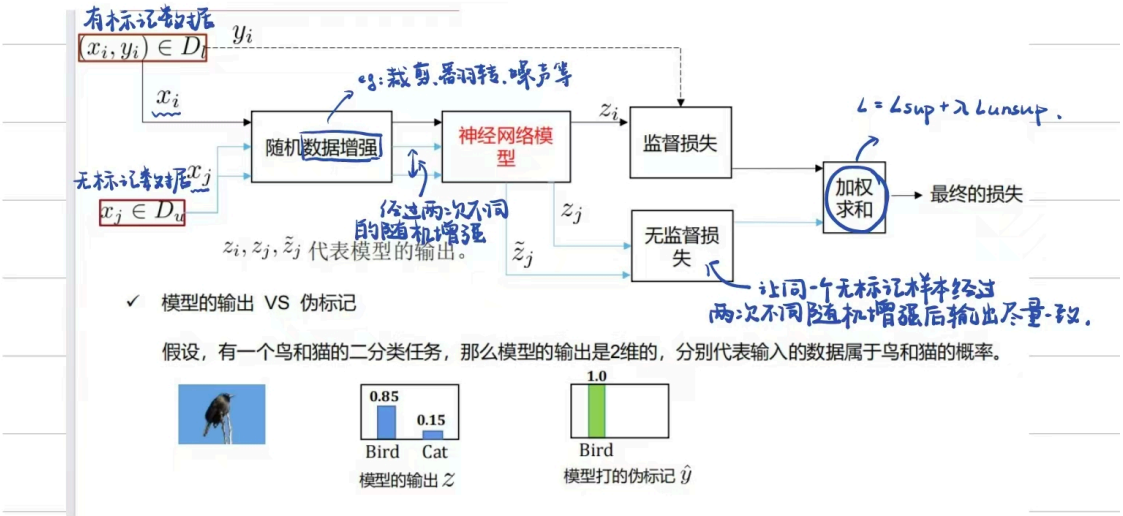


一致性正则化（Consistency Regularization）

1. 思想：同一个样本，即使经过不同的数据增强、扰动或不同训练时刻的模型处理，模型输出也应该保持一致。

> 希望模型对同一个样本在不同扰动下的预测结果尽量一致

2. 代表方法
   - Π模型
     - 让**同一个模型**在两次不同随机增强或扰动下的输出保持一致



     - 左边输入了两类数据：有标记数据和无标记数据。它们都会经过随机数据增强和同一个神经网络模型。有标记数据因为有真实标签，所以可以计算监督损失；无标记数据没有真实标签，所以不能算交叉熵，只能让同一个无标记样本的两次增

强输出保持一致，形成无监督一致性损失。最后，监督损失和一致性损失加权求和，得到最终损失。

- 时序集成
- 平均教师
- ...

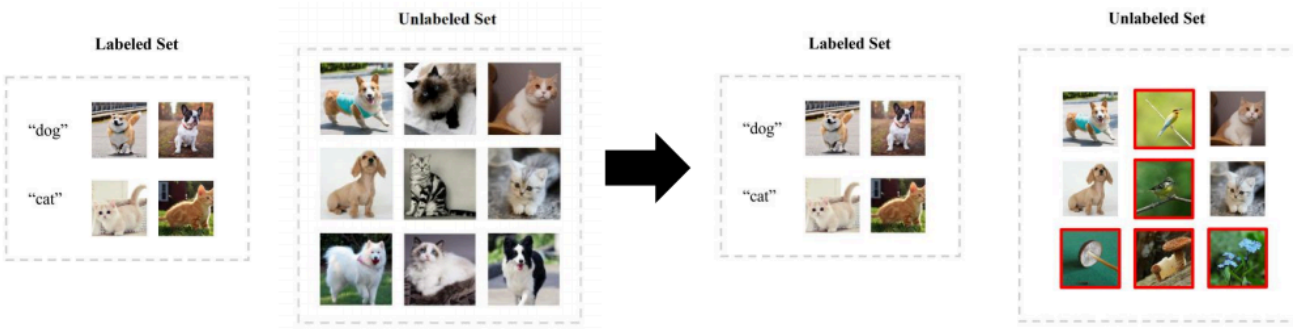
3. 优缺点
   优点：

- 使用起来简单
- 对噪音不敏感
  缺点：
  模型训练需要花费更多的时间，效率低

挑战

开放场景下，如何保证半监督学习方法的安全性？

> 这里的安全性的含义：
> 使用无标记数据以后，模型性能不能比只用有标记数据时更差。



开放场景下的半监督学习面临的主要挑战是**类别分布不匹配**。传统半监督学习通常假设有标记数据和无标记数据来自相同类别空间，例如有标记数据只有 dog 和 cat，那么无标记数据也默认属于 dog 或 cat。但在开放场景下，无标记数据中可能混入 bird、flower、airplane 等未见类样本。如果仍然采用普通伪标签或一致性正则化方法，模型会被迫把这些未见类样本划分为已知类别，从而产生错误伪标签，造成错误传播和负迁移，最终可能使模型性能低于只使用有标记数据的监督学习模型。

因此，开放场景下半监督学习的安全性要求是：**引入无标记数据后，模型性能不能因为未见类样本而下降**。为保证安全性，不能简单地使用全部无标记数据，而应当对无标记数据或模型参数进行筛选和约束。

- 一方面，可以在数据层面识别并降低未见类样本的影响，例如过滤低置信度样本，避免给明显不属于已知类别的样本赋错误伪标签
- 另一方面，可以在参数层面控制模型学习过程。
  - 不是所有模型参数都应该被同等更新，其中一部分参数有利于已见类分类，称为 safe parameters，而另一部分参数容易拟合未见类或无关数据，可能损害模型性

能，称为 harmful parameters。通过发现并保留安全参数、抑制有害参数，从而减弱类别分布不匹配带来的负面影响

开放场景下保证半监督学习安全性的核心思想是：**既要利用无标记数据提供的有用信息，又要防止未见类无标记样本产生错误伪标签或错误梯度；只有当无标记数据对已知类别分类有帮助时才充分利用，对可能有害的信息则进行过滤、降权或参数抑制。**

03 无监督学习

> Unsupervised Learning

概念

1. 什么是无监督学习（Unsupervised Learning,UL)？

已知数据X，在没有其标签Y的情况下，从数据本身挖掘**有用的信息**即丰富的模式

- 判别任务：学习语义上有意义且通用的表征
- 生成任务：学习潜在的数据分布，重建原始数据分布

2. 为什么需要无监督学习

1. **解决大量无标签数据存在和数据标注困难的问题**
   - 在现实世界中，数据的数量庞大，但大多数数据是未标注的
   - 监督学习依赖于人工标注数据，而数据标注成本极高
   - 无监督学习可以利用大量无标签数据，使得机器可以自主学习
2. 无监督学习更符合人类的学习方式
   - 人脑主要依赖于从环境中自动提取信息，而不是依靠大量的人工标签
3. 无监督学习可以提高模型的泛化能力
   - 基于监督学习的模型倾向于记忆标记的数据并结合注释者的偏见，削弱了它们对具有未知数据分布的新场景的泛化能力
   - 无监督学习可以学习更通用的特征表示：UL学习的表示不仅要捕获底层语义信息，而且还可以转移到处理看不见的下游任务，如视觉识别、检测和分割等等

3. 无监督学习形式化问题定义

$$\min_{\theta} \sum_{x \in \mathcal{D}} \mathcal{L}(x, \theta)$$

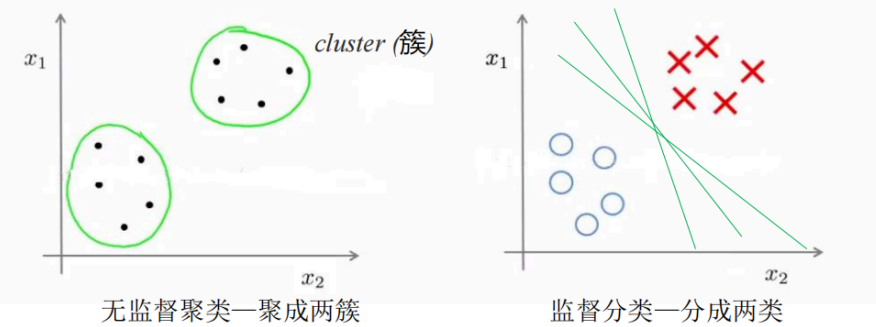
其中

- θ代表模型参数
- $\mathcal{D}$是无标签的数据集，其中x代表一个输入样本
- $\mathcal{L}(x, \theta)$是无监督损失函数
  - K均值聚类：样本间的相似度度量
  - 主成分分析：信息损失
  - 自编码器：重构误差

代表性算法

K均值聚类/K-means

1. 聚类(clustering):将样本分为若干组（通常称为簇），以最小化样本到簇中心的距离，使得簇内（inter-group）相似度高而簇间（intra-group）相似度低



无监督聚类—聚成两簇　　监督分类—分成两类

2. 工作流程
    1. **初始化K**：选择K个样本作为K个簇的簇中心，K是根据一些先验知识人工设定的
    2. **分配簇中心**：基于期望最大化（EM）算法的两步迭代过程
        - Expection-Step（E步）：计算每个样本与k个簇中心的距离（通常是欧几里得距离），选择最小距离的簇中心作为其隶属的簇
        - Maximization-Step（M步）：计算每个簇的所有点的平均值并重新分配簇中心
        - 重复上述过程直至收敛或达到最大迭代次数
3. 目标:最小化所有样本点到其簇中心的欧几里得距离的平方和

$$J = \sum_{i=1}^{N}\sum_{j=1}^{K} w_{ij}\|x_i - c_j\|^2$$

x_i 表示第 i 个数据样本;
c_j 表示第 j 个簇中心;
$w_{ij} = \begin{cases} 1, & \text{如果 } x_i \text{ 被分配到第 } j \text{ 个簇;} \\ 0, & \text{否则。} \end{cases}$

4. 性能分析
   随机初始化可能会产生不一致的结果，如果对相同数据执行多次该算法，则簇中心分配可能会有不同
   极端情况：如果在初始化过程中很不巧的将每个（或大多数）簇中心都初始化到了一个簇中，那么在这种情况下该算法很大程度都不会收敛到全局最小值。

> 当簇中心初始化的位置不得当时，聚类结果将会出现严重的错误

5. 优化方法：K-means++
   与K-means的区别在于：**优化初始簇中心的选择**，其他地方相同

- 从数据集中随机选取一个样本点作为第一个初始聚类中心

- 接着计算每个样本与当前已有聚类中心之间的最短距离（即与最近一个聚类中心的距离），用D(x)表示；然后计算每个样本点被选为下一个聚类中心的概率$\frac{D(x)^2}{\sum D(x)^2}$最后选择最大概率值所对应的样本点作为下一个簇中心；
- 重复第二点直到选择出K个聚类中心

> 核心：距离现有簇中心越远的样本点，越可能被选为下一个簇中心、

6. 优缺点
   应用场景：维度少、数字多且易于分割的机器学习数据
   优点：

- 简单：原理简单，容易实现
- 快速：K 均值聚类采用计算简单的迭代方法设计，相比其他聚类算法更快
  缺点：
- 依赖输入参数：错误的簇中心初始化可能导致运行时间增加和聚类分配质量的低下
- 显著的异常值影响：少量的离群点和噪声点可能对求平均值产生极大影响，从而影响聚类结果

主成分分析PCA

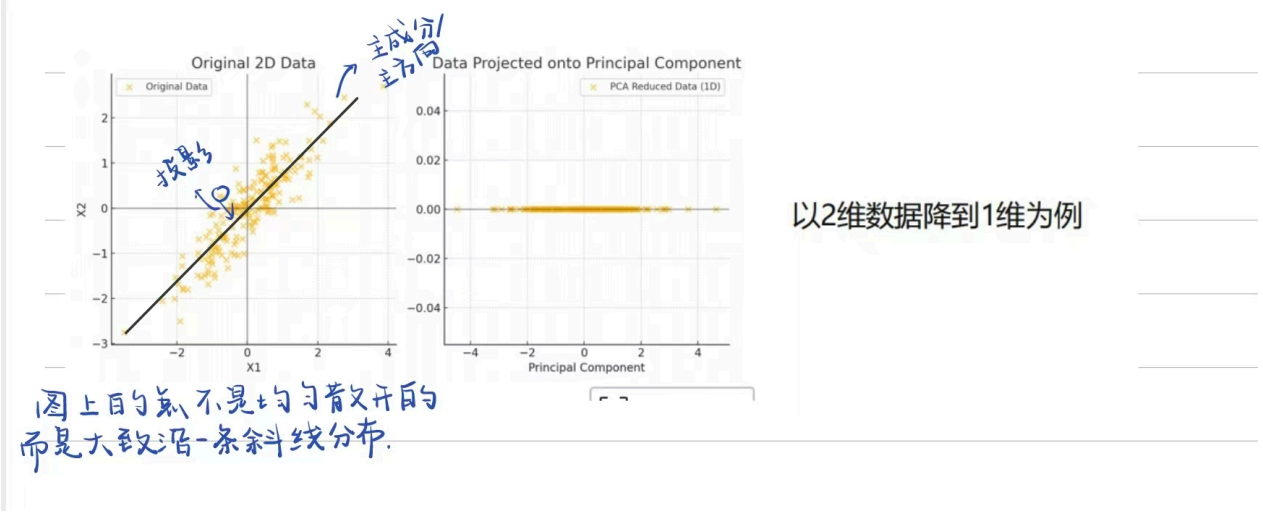
1. 核心思想：**降维**
   背景知识：
   在实际应用中，存在大量高维数据

- 图像数据：每个像素都是一个特征
- 文本数据：词向量表示通常是高维的
  高维数据存在一些问题：
- 计算开销大：高维度意味着更高的存储和计算成本
- 维度灾难：高维数据会导致数据稀疏，影响模型性能
- 难以可视化：超过3维的数据无法直观展示

主成分分析（Principal Component Analysis，PCA）核心思想：
找到数据的最重要的特征，降低维度，同时尽可能保留数据的主要信息

2. 目标
   - 找到数据变化最大的方向（方差最大）
   - 构造一组新的正交坐标轴（主成分）
   - 舍弃重要程度较小的主成分，降低数据维度
     找到数据最主要的变化方向，把数据投影到这些方向上，保留重要方向，舍弃不重要方向，从而实现降维。

- 寻找**数据中方差最大的方向**，因为方差越大表示该方向上数据变化越明显，包含的信息量越多。PCA 会根据数据分布构造一组新的相互正交的坐标轴，这些坐标轴称为主成分，其中第一主成分是数据方差最大的方向，第二主成分是在与第一主成分正交的条件下方差最大的方向。降维时，PCA 保留方差较大的前几个主成分，舍弃方差较小、重要性较低的主成分，从而在尽量减少信息损失的情况下降低数据维度。



3. 计算过程
   假设我们的数据集是$\mathcal{D}=\{x_1,x_2,\ldots,x_m\}$,每个样本$x_i$是n维特征:

$$x_i=(x_{i1},x_{i2},\ldots,x_{in})^T$$

   1. 去中心化
      1. 把数据整体平移到以原点为中心的位置，让 PCA 只关注数据的“分散方向”，而不受数据整体位置影响
   2. 计算样本的协方差矩阵：捕捉特征之间的相关性，协方差大的特征对数据分布影响更大
   3. 对协方差矩阵做特征值分解
      - 特征向量 v_i 代表一个主成分方向，而特征值 λ_i 代表数据在该方向上的方差信息

$$\Lambda=\begin{bmatrix}\lambda_1 & 0 & 0 & \ldots & 0\\ 0 & \lambda_2 & 0 & \ldots & 0\\ 0 & 0 & \lambda_3 & \ldots & 0\\ \vdots & \vdots & \vdots & \ddots & 0\\ 0 & 0 & 0 & 0 & \lambda_n\end{bmatrix} \qquad V=(v_1,v_2,\ldots,v_n)$$

   4. 选取前d'个主成分
   5. 输出数据降维

4. 优缺点
   应用场景:

- 图像压缩：PCA 可降低图像维数，同时保留基本信息。它有助于创建紧凑的图像表示，使其更易于存储和传输
- 数据可视化：PCA 通过将高维数据投影到低维空间（例如 2D 或 3D 图）来帮助可视化高维数据。这简化了数据解释和探索
- 噪声过滤：PCA 可以通过关注捕捉潜在模式的主要成分来消除数据中的噪音或冗余信息

优点:

- 降低数据维度，简化模型

- 保留数据中的主要变化模式
- 去除噪声和冗余信息

缺点：

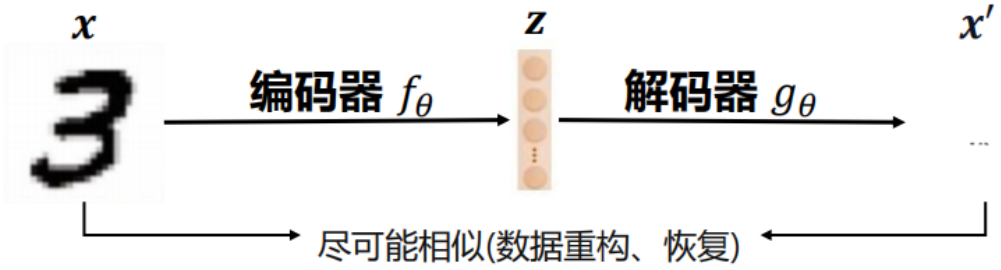
- 对缺失值和异常值敏感
- 对数据的线性关系敏感，可能丢失非线性结构信息

> PCA 本质上是一种**线性降维方法**。

自编码器AE

> AutoEncoder

1. 核心思想：先把输入数据压缩成一个低维表示，再尽量把它还原回来



其中：

- 输入数据 x 经过编码器（Encoder）转换为低维表示 z：$z = f_\theta(x)$
- 解码器（Decoder）试图从 z 重构出原始数据：$x' = g_\theta(z)$
- 训练目标是最小化重构误差： $Loss = \sum_{i=1}^{N} (x_i - x_i')^2$

自编码器真正有价值的地方不只是输出 x'，而是中间的低维表示：z

预测输出图像的期望像素值，就是指解码器对每个像素位置给出一个**平均意义上的预测强度值**

2. 目标：重建输入图像来学习数据的低维表示

> 前面 AE 的基本目标是：
>
> $$x \to z \to x'$$
>
> 也就是把原始数据 x 编码成低维特征 z，再从 z 重构出 x'。
> 这一页进一步说明：**不是随便设计一个 AE 就能学到好的特征表示，需要合理设计编码器、解码器和重建任务。**

如何优化特征表示（主要是指中间的隐变量z）：

> 让 z 尽可能保留原始数据中的关键信息，同时去掉不重要的信息

- 选择合适的编码器和解码器

- AE 的结构设计会影响它学到的特征质量

- 根据不同的任务设计重建任务

  普通 AE 的重建任务是：

  $$输入原图 \rightarrow 重构原图$$

  也就是：

  $$x' \approx x$$

  但是不同任务下，重建目标可以不一样。

  比如：

  如果任务是**降维**，可以让 AE 学会用较低维 z 尽量重构原始数据；

  如果任务是**去噪**，可以输入带噪声的图片，但要求输出干净图片：

  $$\tilde{x} \rightarrow x$$

  这就是去噪自编码器。

  如果任务是**异常检测**，可以让 AE 学习正常样本的重构规律。测试时，如果某个样本重构误差很大，就可能是异常样本。

  所以这一点是在说：

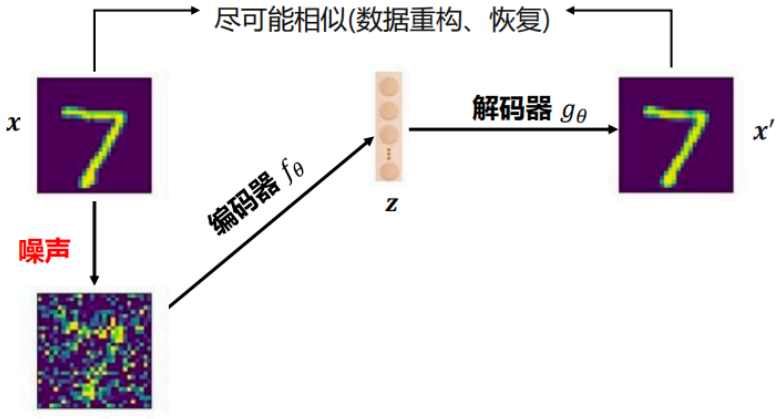
  AE 不只是固定地重构输入，而是可以根据任务目标设计不同的输入和输出，从而让模型学到更有用的特征表示。

3. 优化

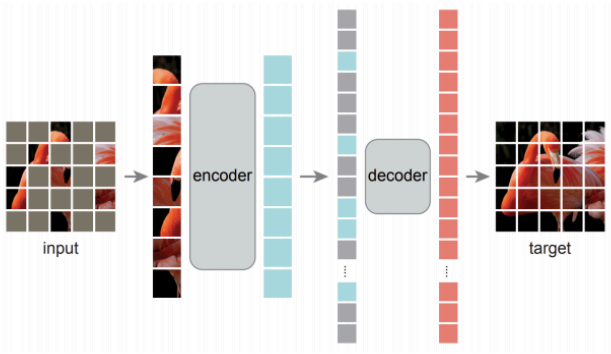
   对于图像而言，图像像素的冗余性高，普通自编码器容易记住输入数据的细节，无法学习其更鲁棒的特征

   如何让AE学习更鲁棒的特征，防止过拟合？

- 去噪自编码器（Denoising Autoencoder，DAE）：在输入数据中**加入随机噪声**，然后训练自编码器恢复图像



- 掩码自编码器（Masked Autoencoder，MAE）：通过**随机遮挡部分输入**，让模型预测被遮挡的信息



- 模型首先将输入图像划分为若干图像块，并随机遮挡其中一部分 patches，得到带有掩码的输入。编码器只根据可见的图像块提取隐表示，解码器再根据该隐表示尝试重建完整图像，尤其是恢复被遮挡的区域。训练目标是最小化重建图像与原始目标图像之间的误差

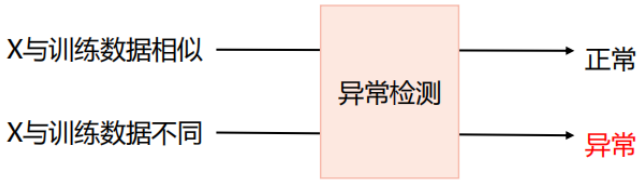
> 需要注意的是AE、DAE和MAE其目的都是为了**学习中间的低维特征表示**，从而用于下游任务如分类、检测或分割
>
> AE、DAE、MAE 训练完成后，可以把解码器去掉，只保留编码器：
>
> $$x \to z$$
>
> 然后把这个 z 或编码器输出的特征接到分类器、检测头、分割头后面。
>
> 也就是说：
>
> **先用大量无标签数据训练编码器学特征，再把学到的特征迁移到具体任务中。**

算法比较

算法	K-means聚类	主成分分析PCA	自编码器AE
方法类型	聚类	线性降维	非线性降维
核心思想	通过计算点与聚类中心的距离，将数据点分配到最近的簇	计算数据的主成分方向，将数据投影到低维空间	通过神经网络编码-解码过程学习数据的低维表示
关键步骤	选择初始聚类中心 → 计算距离 → 重新分配数据→ 更新中心 → 迭代直到收敛	计算协方差矩阵 → 特征值分解 → 选择前 d’ 个主成分 → 数据投影	训练神经网络（编码器 +解码器） → 最小化重构误差
计算复杂度	适中，取决于数据点和聚类数 K	低，主要涉及矩阵运算	高，训练神经网络需要较长时间
缺点	1. 易受初始中心点选择影响 2. 受异常值影响较大	1.仅适用于线性降维 2. 无法捕捉复杂的数据结构	1. 训练过程较慢 2. 需要大量的数据 3. 可能过拟合

实际应用：异常检测

异常检测（Anomaly detection）识别与大多数数据显著不同的数据点
给定一组训练数据，我们想找到一个函数来检测输入 x 是否与训练数据相似



实现：

- K-means
  - 使用k均值进行聚类，计算每个数据点到最近的聚类中心的距离
  - 易于实现，但高维数据可能表现不佳
- PCA
  - 计算数据点的PCA降维后的重构误差
  - 适合高维数据，但是对非线性数据的异常点不够敏感

$$\| X - \hat{X} \|^2$$

重构误差

- Auto-Encoder
  - 使用自动编码器重构原始数据，计算重构误差
  - 适合复杂、非线性数据，但需要大量数据和较高的计算资源需求

当距离或者误差大于人工设置的阈值时，认定为异常数据

04 度量学习

> Metric Learning

核心问题：什么是度量学习

度量学习是一类学习样本间距离或相似性的机器学习方法。利用对象之间的相似性关系，为样本学到有效的特征表示，使得新表示空间中样本之间的距离度量能够精确反映样本之间的相似与不相似关系。

它的基本思想是：首先给定一个带标签的训练集，每个样本由特征 x_i 和标签 y_i 组成。根据标签是否相同，可以构造相似样本对集合 S_x 和不相似样本对集合 D_x。如果两个样本标签相同，就希望它们在学习后的度量空间中距离更近；如果两个样本标签不同，就希望它们的距离更远。因此，度量学习的目标就是学习一个矩阵 A，定义新的距离函数 $D_A(x_i, x_j)$，使**同类样本距离最小化、异类样本距离最大化**。简单来说，度量学习学习的不是具体类别，而是学习一种更合理的相似性判断方式。

相比普通欧氏距离，度量学习不再假设所有特征同等重要，而是根据具体任务学习更合适的相似性度量。

- LDA 可以看作一种线性度量学习思想，它通过投影让类间距离变大、类内距离变小
- 流形学习则关注高维数据背后的低维非线性结构
- 深度度量学习进一步使用神经网络学习复杂的特征映射，使样本在新空间中的距离更加符合语义或类别关系。

概念

导引

常见的机器学习任务：

- 分类
  - 判断一个新样本距离哪一类最近
- 聚类
  - 判断哪些点应该分到同一簇
- 回归
  - 也可能用样本之间的距离做预测
- 降维
  - 尽量保持原空间中的相似关系

    分类、聚类、回归、降维虽然任务形式不同，但它们都可能依赖样本之间的距离或相似性判断。因此，如果距离定义得不好，后续机器学习任务的效果也会受到影响。

    度量学习要解决的问题：**既然很多任务都依赖距离，那我们能不能不要手工规定距离，而是让模型自己学习一个更合适的距离？**

欧氏距离是：

$$D(x,\mu)=\sqrt{(x-\mu)^T I(x-\mu)}$$

马氏距离是：

$$D_M(x,\mu)=\sqrt{(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

它们的区别在于中间的矩阵不同。

欧氏距离用的是单位矩阵 I，相当于认为每个方向一样重要。

马氏距离用的是协方差矩阵的逆 Σ^{-1}，相当于根据数据分布调整不同方向的重要性。

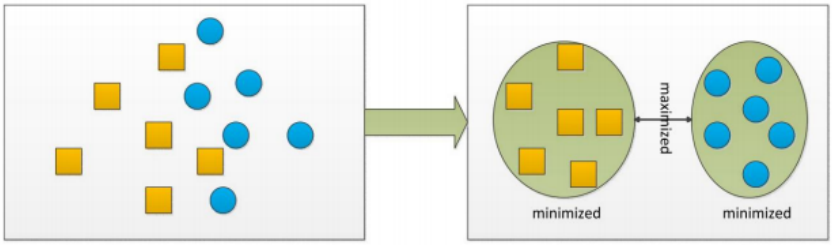
而度量学习进一步推广为：

$$D_A(x_i,x_j)=\sqrt{(x_i-x_j)^T A(x_i-x_j)}$$

也就是说，度量学习不再只根据数据整体分布得到 Σ^{-1}，而是直接学习一个更适合任务的矩阵 A。

定义

1. 问题定义：利用对象之间的相似性关系，为样本学到有效的特征表示，使得新表示空间中样本之间的距离度量能够精确反映样本之间的相似与不相似关系
2. 形式化定义：
   给定一个训练集：$T=\{(x_i,y_i)\}_{i=1}^m$
   相似集：$S_{\mathbf{x}}=\{(x_i,x_j)\mid y_i=y_j\}$
   不相似集：$D_{\mathbf{x}}=\{(x_i,x_j)\mid y_i\neq y_j\}$



在原始空间中，两类样本可能混在一起，距离关系不一定清楚。比如有些蓝色点离黄色方块很近，有些同类点之间又比较远。
右边是经过度量学习之后的效果。
它希望做到两件事：

- 第一件事：同类样本靠近
- 第二件事：不同类样本远离

> 度量学习希望改变样本之间的距离关系，使同类更紧密，异类更分离。

因此，所要学习的度量课定义为：

$$D_A(x_i,x_j)=\sqrt{(x_i-x_j)^T A(x_i-x_j)}$$

- 其中 D_A 满足如下条件:
  - 非负性(non-negativity) $d(x_i, x_j) \geq 0$
  - 自反性(reflexivity) $d(x_i, x_j) = 0 \; if \; x_i = x_j$
  - 对称性 (symmetry) $d(x_i, x_j) = d(x_j, x_i)$
  - 三角不等式 (triangular inequality) $d(x_i, x_l) \leq d(x_i, x_j) + d(x_j, x_l)$

度量学习不是直接用固定的欧氏距离，而是学习一个矩阵A，让距离计算更符合任务需求。

- 度量学习虽然要学习矩阵 A，但是 A 不能乱学。为了让学出来的距离函数 D_A 是合理的距离，必须约束 A 是半正定矩阵，即 $A \succeq 0$（保证非负性）。

优化目标:

$$\min_A \sum_{(x_i,x_j)\in S_x} d_A(x_i, x_j) \quad \text{and} \quad \max_A \sum_{(x_i,x_j)\in D_x} d_A(x_i, x_j)$$

对比普通欧式距离:

$$D(x_i, x_j) = \sqrt{(x_i - x_j)^T I (x_i - x_j)}$$

- 普通欧氏距离默认每个特征都一样重要
- 度量学习就希望学出一个 A，让重要特征的差异被放大，不重要特征的差异被减小
  - A的作用：调整不同特征、不同方向在距离计算中的重要程度

3. 度量学习的基本定义
   首先给定一个带标签的训练集，每个样本由特征 x_i 和标签 y_i 组成。根据标签是否相同，可以构造相似样本对集合 S_x 和不相似样本对集合 D_x。如果两个样本标签相同，就希望它们在学习后的度量空间中距离更近；如果两个样本标签不同，就希望它们的距离更远。因此，度量学习的目标就是学习一个矩阵 A，定义新的距离函数 $D_A(x_i, x_j)$，使**同类样本距离最小化、异类样本距离最大化**。简单来说，度量学习学习的不是具体类别，而是学习一种更合理的相似性判断方式。

$$\min_A \sum_{(x_i,x_j)\in S_x} d_A(x_i, x_j) \quad \text{and} \quad \max_A \sum_{(x_i,x_j)\in D_x} d_A(x_i, x_j)$$

相似集　　不相似集

$$\text{s.t.} \quad A \succeq 0$$

半正定

距离度量 $D_A(x_i, x_j) = \sqrt{(x_i - x_j)^T A (x_i - x_j)}$

经典算法

马氏度量学习（LDA）

> 线性判别分析(Linear Discriminant Analysis，LDA)

二分类问题为例：
找一个投影方向 w，把数据投影到一条直线上，使得不同类别之间尽量分开，同一类别内部尽量紧凑

- 类间散度大：把两类样本投影到直线以后，希望两类的均值距离尽可能大
- 类内散度小：同一类样本投影到直线之后，希望它们不要分得太散

	原始空间	新空间
均值	$u_i = \frac{1}{N_i}\sum_{x\in D_i} x, \quad (i=1,2)$	$\widetilde{\mu_i} = \frac{1}{N_i}\sum_{x\in D_i} w^T x, \qquad (i=1,2)$
方差	$S_i = \sum_{x\in D_i}(x-u_i)(x-u_i)^T, \quad (i=1,2)$	$\widetilde{S_i} = \sum_{x\in D_i}(w^T x-\widetilde{\mu_i})^2, \qquad (i=1,2)$
类内散度	$S_w = S_1 + S_2$	$\widetilde{S_w} = \widetilde{S_1}+\widetilde{S_2}$
类间散度	$S_b = (u_1-u_2)(u_1-u_2)^T$	$\widetilde{S_b} = (\widetilde{\mu_1}-\widetilde{\mu_2})^2$

优化目标 $\max J(w) = \frac{(\widetilde{\mu_1}-\widetilde{\mu_2})^2}{\widetilde{S_1}+\widetilde{S_2}}$

> 让类间距离尽可能大，同时让类内散度尽可能小。

流形学习

流形是一个局部点近似于欧氏空间的拓扑空间

> 流形就是一种弯曲的低维空间。虽然整体可能是弯的、复杂的，但如果你只看很小的一块，它就像普通的平面或直线一样

深度度量学习

在深度学习里，我们不一定显式学习一个矩阵A，而是可以用深度神经网络学习一个“空间变换”，把样本从原始空间映射到新的特征空间中，让相似样本更近。

> 度量学习的思想可以用深度神经网络实现。传统方法学习矩阵 A，深度方法学习网络参数，通过神经网络把原始样本映射到新的特征空间中，使相似样本距离更近、不相似样本距离更远。

可以简单理解为：

传统度量学习：学一个矩阵A来改距离

深度度量学习：学一个神经网络f来改表示

最后都是为了：

同类近，异类远

05 迁移学习

> Transfer Learning

概念

> 引入：迁移学习是机器学习领域用于解决标记数据难获取这一基础问题的重要手段

什么是迁移学习？

迁移学习是利用**数据，任务或模型**之间的**相似性**，将在旧问题学习过的模型，应用于新问题的一种学习过程

- 旧问题：源域
- 新问题：目标域
  核心：利用源域和目标域之间的相似性

形式化定义：
给定一个（多个）源域 $D_s = \{x_i, y_i\}_{i=1}^{n_s}$，和一个（多个）目标域 $D_t = \{x_i, y_i\}_{i=1}^{n_t}$，其中 $x \in \mathcal{X}, y \in \mathcal{Y}$，$P_S$ 表示源域分布，P_t 表示目标域分布，迁移学习的目的利用源域数据来提高目标域上的性能。

- 分布：数据整体长什么样，数据有什么特征规律。
  - 数据分布可以理解为数据整体呈现出来的规律，包括数据的外观、来源、风格、取值范围、类别比例、特征组合方式等。
    三种设置：
- $0 \leq n_t \leq n_s$（通常情况下，目标域数据比源域少，甚至可能没有标签数据）
- $\mathcal{X}_s \neq \mathcal{X}_t$ or $\mathcal{Y}_s \neq \mathcal{Y}_t$（源域和目标域之间至少在输入特征空间或输出标签空间上有差异）
- $\mathcal{Y}_t$ 可知或不可知（目标域标签可能有，也可能没有）

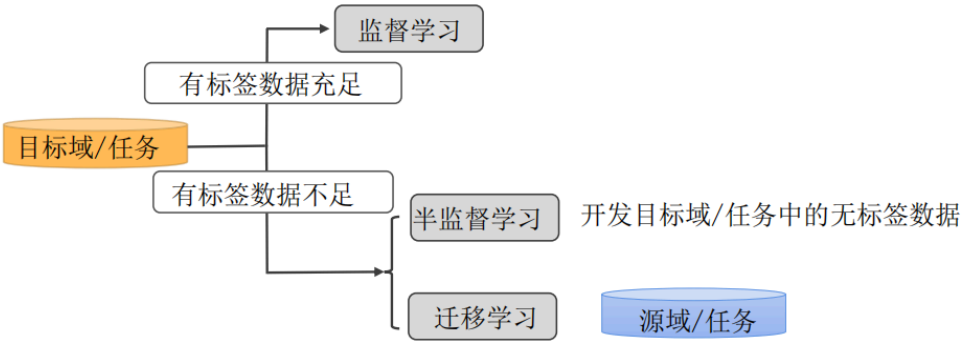
问题：**负迁移**
定义：在源域中学习的知识对目标域的学习有负面作用
原因

- **数据**：源域数据和目标域数据本质上不相似
- **方法**：源域数据和目标域数据本质上相似，但是由于迁移方法不合理，难以找到可迁移的成分

为什么有迁移学习？

- 特定应用中标注数量少eg:医学图像
  - 半监督学习：少量有标签数据 + 大量无标签数据，仍然主要使用目标域自己的数据

- 迁移学习：利用源域/源任务中已有的标注数据或预训练模型，帮助目标域/目标任务学习
  - 利用迁移学习，我们可以利用已经标注好的相似数据，并且利用它们来进行模型的学习。



利用迁移学习，我们可以利用已经标注好的相似数据，并且利用他们来进行模型的学习.

- 半监督学习 vs 迁移学习

方法	主要利用什么数据	数据来自哪里	核心思想
半监督学习	无标签数据	目标域/目标任务本身	用目标任务中的大量无标签数据辅助训练
迁移学习	已有标注数据或预训练知识	源域/源任务	把别的任务学到的知识迁移到当前任务

- 机器学习模型需要鲁棒性
  - 在新场景下，源域模型可以利用迁移学习的思想需要在使用前进行调整
  - 当源域和目标域分布不同，但任务相关时，需要通过迁移学习把源域学到的知识调整到目标域中。
- 特定应用需求
  - 利用迁移学习，寻找相似领域的模型或者用户在其他领域的数据来进行系统的初始化，有效缓解冷启动问题
    - 一个新系统、新用户、新商品刚出现时，由于历史数据太少，模型不知道该怎么推荐或预测。

方法

如何进行迁移学习？

- 基于示例的迁移学习
  - 基本假设：源域样本和目标域样本中存在交集
  - 做法：调整源域数据示例的权重，增加有利于目标域任务的实例权重，降低不利用目标域任务的实例权重
- 基于特征的迁移学习
  - 基本假设：源域特征和目标域特征之间存在交叉部分（共同特征）
  - 做法：将源域和目标域数据转换到更相似的空间，然后在这个空间中找到共同的特征来进行迁移
- 基于模型的迁移学习

- 基本假设：源域模型和目标域模型之间存在相似部分
- 做法：利用源域中已经训练好的模型参数、结构或先验知识来初始化或约束目标域模型，再根据目标域数据进行微调，使模型适应目标任务。

一些迁移学习的子类

域泛化

> Domain generalization

和迁移学习的关系

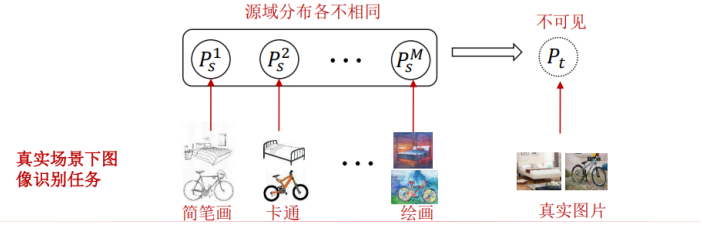
迁移学习/领域自适应通常可以看到一些目标域数据，哪怕没有标签；
域泛化更难，因为训练时完全看不到目标域数据。

定义

给定：M个训练的域 $D=\{D_S^i \mid i=1,\dots,M\}$, where $D_S^i=\{(x_j^i,y_j^i)\}_{j=1}^{n_i}$
条件：

- 分布之间均不同：$P_S^i \neq P_S^j,\ 1\leq i\neq j\leq M$
- 在训练过程中目标域不可得
  目标：在分布和源域均不同的目标域上，得到一个性能好的模型

> 在目标域上的测试误差最小



方法

- 数据操纵
  - $\min_h \mathbb{E}\left[l(h(x),y)\right]+\mathbb{E}\left[l(h(x'),y)\right]$

所以第一项：

$$\mathbb{E}[l(h(x), y)]$$

表示模型在真实源域数据上的平均损失。

第二项：

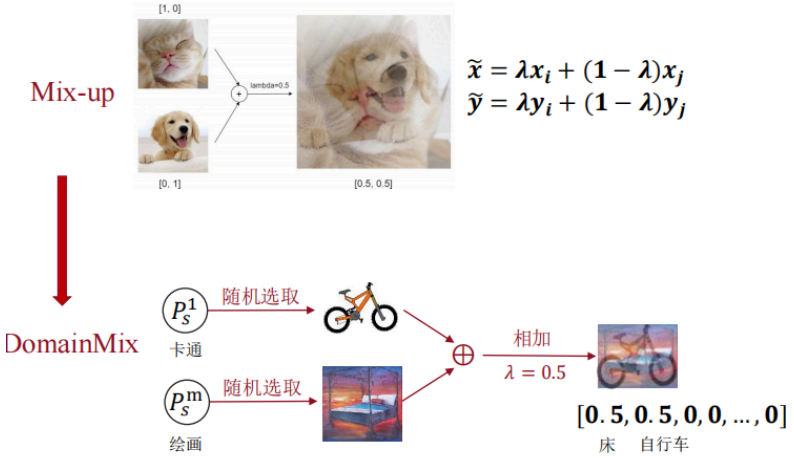
$$\mathbb{E}[l(h(x'), y)]$$

表示模型在生成数据上的平均损失。

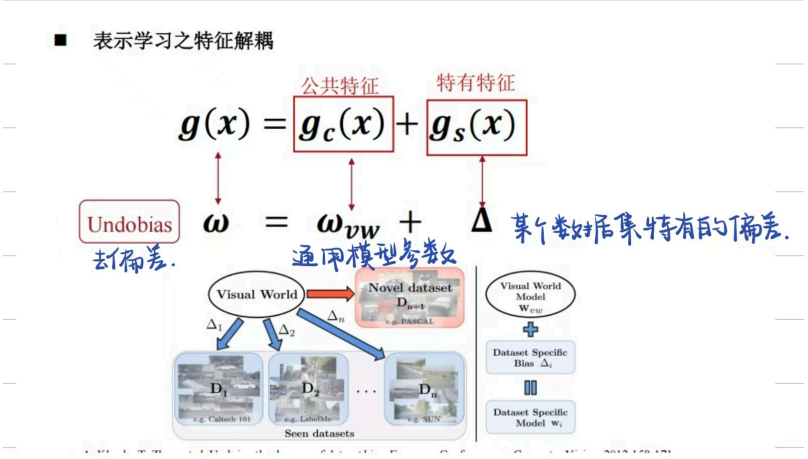
整个目标是：

> 同时最小化真实数据上的错误和增强数据上的错误。

- 如何生成高质量有利于泛化的数据
  - 各种数据增强 eg:去纹理/去颜色/边缘增强/翻转/旋转等等
  - DomainMix：把两个样本按比例混合，得到一个新的样本
    - 它不是随便混两个样本，而是从不同源域中各选一个样本进行混合



- 表示学习
  - 特征解耦：在域泛化中，模型学到的表示可以分解为**公共特征**和**特有特征**。公共特征是不同域之间共享的、有利于泛化的本质信息；特有特征是某个数据集或某个域中特有的偏差信息。特征解耦的目标就是把这两部分分开，削弱数据集偏差，让模型更多依赖公共特征，从而在未知目标域上表现更好。



- 学习策略
  - 集成学习：利用训练数据构建一组基学习器，然后通过对每个基学习器的预测进行投票来产生最终预测

- 基学习器的条件
  - 性能不能太差 ⬅️ 每个源域和目标域的分布是相似的
  - 多样性：表现尽量不同 ⬅️ 每个源域的分布是不完全相同的
    - 不同模型从不同角度学习数据，它们的错误可以互相补充、互相抵消。

> 不只训练一个模型，而是用不同源域的数据训练多个基学习器，然后把这些模型的预测结果综合起来，得到最终输出。

多任务学习

> (multi-task learning，MTL)

和迁移学习的关系

联系：核心是知识迁移，一种迁移学习的思想和模式
区别：

- 迁移学习的大部分算法重点**在于提升目标域上的性能**
- 多任务学习中无源域和目标域的定义，其大部分算法的重点**在于提升所有相关任务上的性能**

基本概念

1. 引入/应用：如何训练自动驾驶汽车的模型，能同时预测车辆行驶方向（向前行驶还是停止）、行人位置（站在马路对面还是在斑马线上）、交通信号灯状态（红/绿）这三个任务？

   > 使用一个共享的主干网络先提取道路场景中的公共特征，然后分别接三个任务的输出分支。

   可以写成：

   $$输入图像 \rightarrow 共享特征提取网络 \rightarrow 三个任务输出$$

   例如输入一张道路图像，模型先共同提取一些基础信息：

   道路边界、车辆位置、行人轮廓、斑马线、交通灯区域、场景光照等。

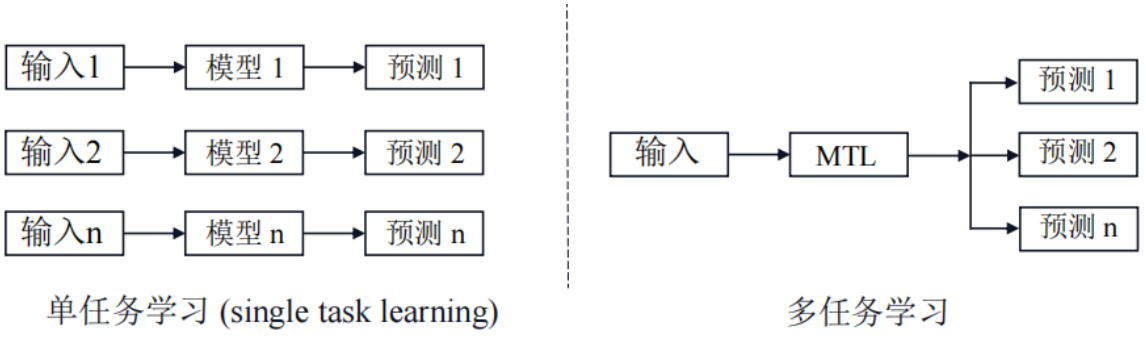
   然后分成三个任务头：

   任务一：判断车辆是继续行驶还是停止；
   任务二：判断行人在马路对面还是斑马线上；
   任务三：判断交通信号灯是红灯还是绿灯。

   为什么这三个任务适合一起学呢？➡️存在明显关联

- 如果信号灯是红灯，那么车辆更可能需要停止；

- 如果行人在斑马线上，那么车辆也更可能需要停止；
- 如果信号灯是绿灯，且斑马线上没有行人，那么车辆更可能继续行驶。



单任务学习 (single task learning)　　多任务学习

2. 定义：
给定n个任务，其中所有或部分任务是相关但不相同的，多任务学习的目的是利用这些n个任务的知识来提高所有任务的表现

3. 优点
    1. 方便性【时间成本、计算成本、存储成本】
        - *eg*:多任务学习可以让多个任务共享一部分模型结构，比如共享底层特征提取网络，只在最后为不同任务设置不同输出头
    2. 增强鲁棒性
        - 多个任务一起训练时，模型会看到更多样的数据变化和噪声形式，就不容易被某一种噪声误导
    3. 窃听机制
        - 一个任务可以借助另一个任务学到的有用信息
        - 困难任务可以从相关的简单任务中获得额外监督信号，从而更容易学到关键特征
4. 如何进行多任务学习

- 软共享
    - 每个任务都有自己独立的网络，但不同任务的网络之间会进行信息交互或参数约束
- 硬共享
    - 多个任务共享同一个底层特征提取网络，只在最后输出层分成不同任务

域自适应

> Domain adaptation

背景

监督学习 vs 域自适应

监督学习：假设训练和测试数据同分布，直接通过标注数据优化模型
域自适应：需处理分布差异，通过迁移源域知识适应目标域

- **协变量偏移**：输入特征分布变了，但输入到标签的对应关系没有变。
    - $P_{\text{source}}(x) \neq P_{\text{target}}(x)$

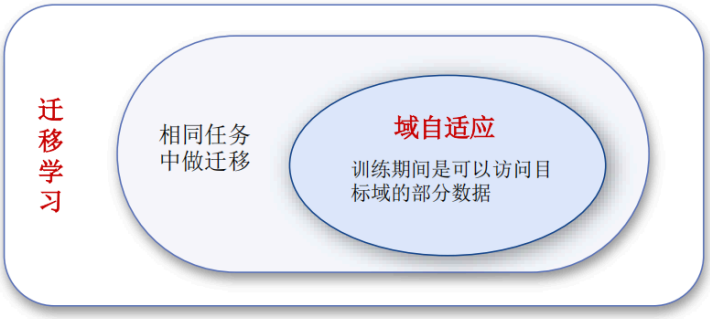
- $P(y \mid x)$

	监督学习	域自适应
数据分布假设	训练集和测试集**独立同分布**	源域和目标域数据分布不同
数据标签需求	所有数据均有标注	源域有标注，目标域可能无标注
核心目标	在同一分布下最大化模型性能	解决分布差异，实现跨域的模型泛化
方法设计	直接优化损失函数	伪标签生成、特征对齐或对抗训练等域自适应方法

域自适应与迁移学习的关系

联系：域自适应是迁移学习的一个子类
差别：

- 任务：域自适应假设模型是**在相同的任务**中做迁移
- 数据：域自适应假设模型**在训练期间是可以访问目标域的部分数据**



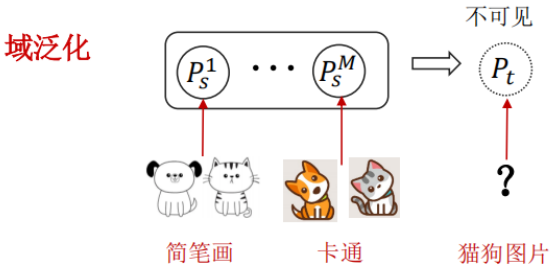
域自适应 vs 域泛化 vs 多任务学习

域自适应：将从源域上学到的模型迁移到不同但相关目标域



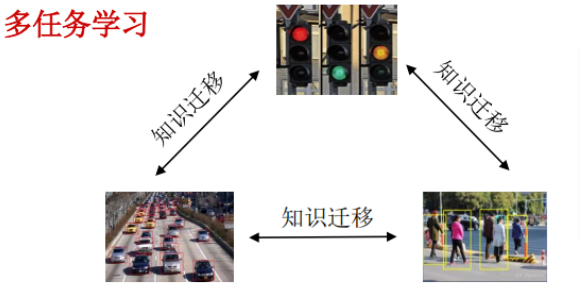
- 一个源域和一个目标域
- 源域域目标域数据分布不同
- 训练时可以获得部分目标域数据

域泛化：将从多个源域数据训练的模型泛化到未知数据分布的目标域



- 多个源域
- 多个源域之间分布不同
- 训练时目标域不可获取

多任务学习：通过使用多个相关任务中包含的知识来帮助提升各个任务的性能



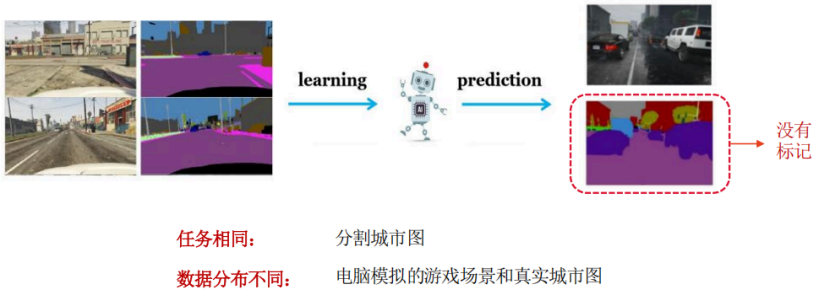
- 存在多个任务
- 多个任务之间具有相关性
- 训练时多个任务的数据都可以获取

定义

域自适应：在机器学习中，通过学习源域和目标域之间的差距，实现将源域的模型迁移到不同但相关目标域

> 域自适应旨在解决源域与目标域之间的分布差异问题

应用场景：主要用于任务相同，只是数据分布不同的场景中

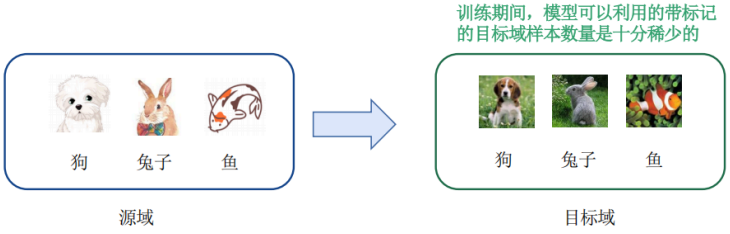


分类

监督域自适应 (Supervised Domain Adaptation)

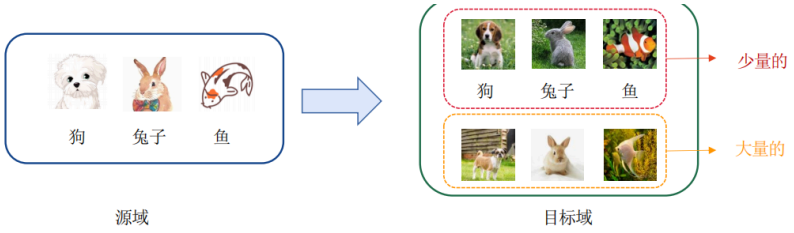
源域P_S：m 个带标记的样本 $\{x_i, y_i\}_{i=1}^m$ 来自分布 $Q(X,Y)$；训练集
目标域P_t：少量带标记的样本 $\{x_i, y_i\}_{i=1}^k$ 来自分布 $P(X,Y)$；一部分训练集+测试集

目标：在分布差异下(即 $Q \neq P$)，利用**带标记的源域样本和少量带标记的目标域样本**学习一个在目标域中表现性能好的模型 $h: x \rightarrow y$



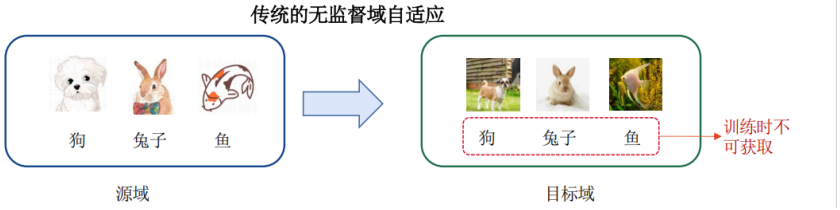
半监督域自适应 (Semi-supervised Domain Adaptation)

源域P_S：m 个带标记的样本 $\{x_i, y_i\}_{i=1}^{m}$ 来自分布 $Q(X, Y)$；
目标域P_t：少量带标记的样本 $\{x_i, y_i\}_{i=1}^{k}$ 来自分布 $P(X, Y)$；
目标：在分布差异下(即 $Q \neq P$)，利用**带标记的源域样本，少量带标记的目标域样本和大量无标记的目标域样本**学习一个在目标域中表现性能好的模型 $h: x \rightarrow y$



无监督域自适应 (Unsupervised Domain Adaptation)

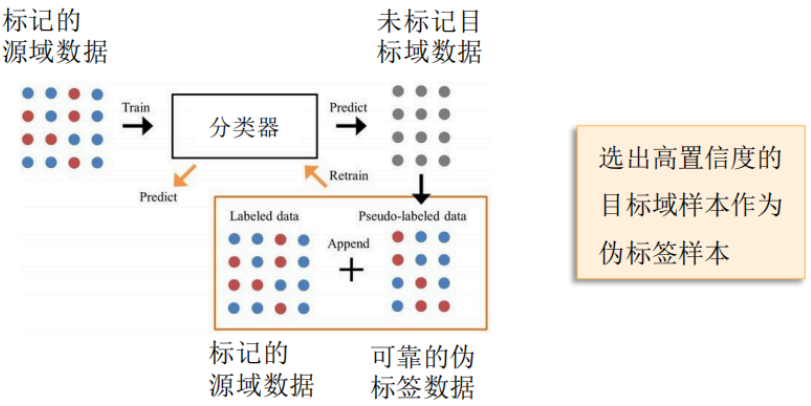
源域P_S：m 个带标记的样本 $\{x_i, y_i\}_{i=1}^{m}$ 来自分布 $Q(X, Y)$；
目标域P_t：少量带标记的样本 $\{x_i, y_i\}_{i=1}^{k}$ 来自分布 $P(X, Y)$；
目标：在分布差异下(即 $Q \neq P$)，利用**带标记的源域样本和无标记的目标域样本**学习一个在目标域中表现性能好的模型 $h: x \rightarrow y$



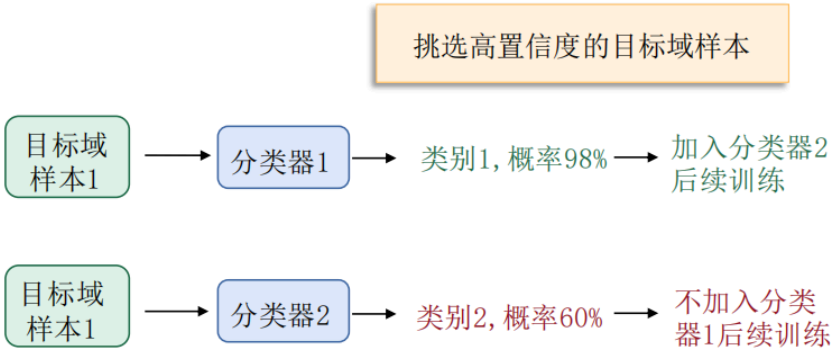
方法

示例层面：基于伪标签的方法

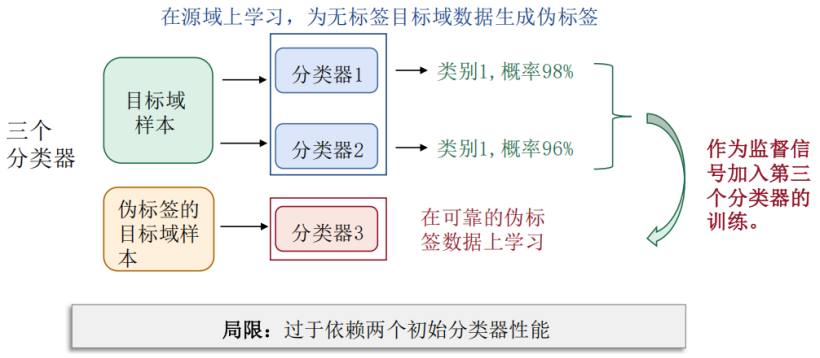
- 一个分类器



  - 核心思想：尝试为目标域样本分配伪标签，并选择**可靠的伪标签**来完成后续的监督训练
  - 局限：可能会过拟合到错误的预测标签
- 二个分类器



  - 核心思想：训练两个独立的模型，选择可靠的伪标签加入另一个模型的后续训练
  - 局限：两个分类器训练到最后可能会趋于一致
- 三个分类器

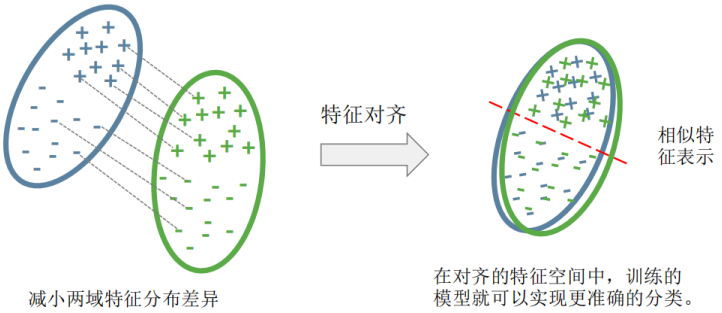


  - 核心思想：使用三个独立训练的模型，第三个模型所获得新标记样本都由其余两个模型协作提供
  - 局限：过分依赖两个初始分类器性能

特征层面：基于特征对齐的方法

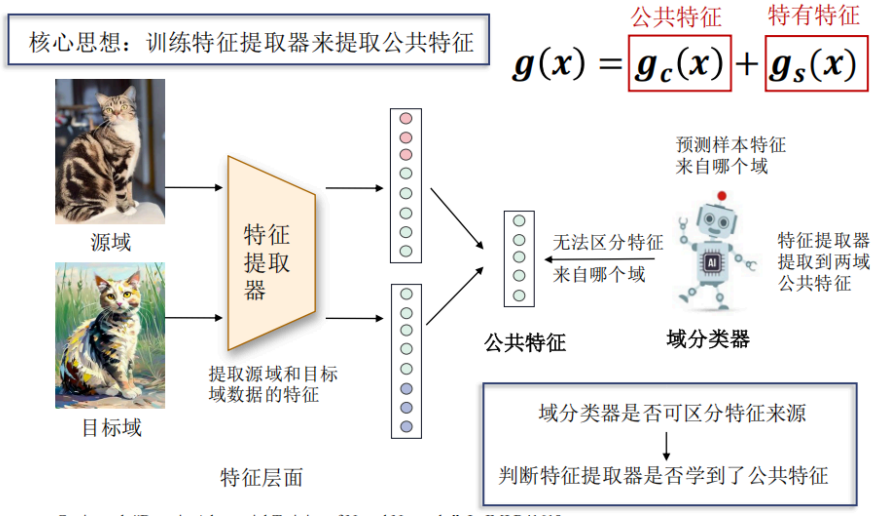
思想：既然源域和目标域在原始样本层面差异很大，那能不能通过特征提取，把它们映射到一个共同的特征空间里，让它们的表示变得接近？
也就是说：不要求原始图片完全一样，而是希望模型提取到的特征表示相似

- 特征对齐的目标：通过学习一个特征映射，把源域和目标域映射到同一个特征空间，使它们的特征分布尽量接近
    - 度量两域特征分布之间的距离
        - 最大均值差异（Maximum Mean Discrepancy，MMD）
            - 将源域和目标域数据映射到一个共同的特征空间，使得两域数据在该空间中的分布差异最小化
            - 可以通过最小化 MMD 来训练特征提取器，使源域和目标域在特征空间中的分布尽量对齐，从而提取更具迁移性的域不变特征。



模型层面：基于对抗的方法

核心思想：训练特征提取器来提取公共特征

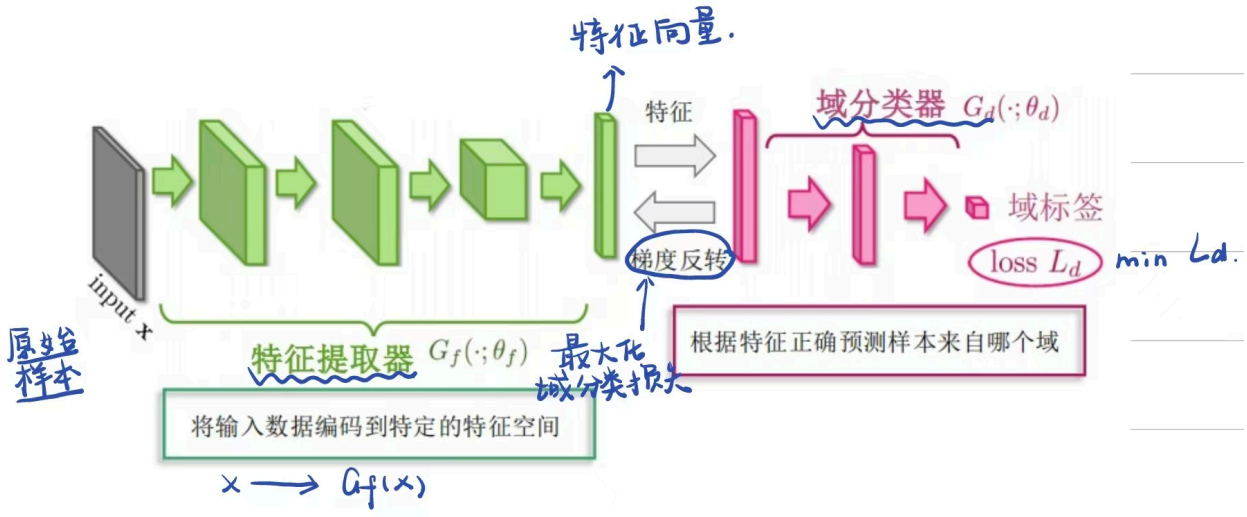


域对抗神经网络（Domain Adversarial Neural Network）

- 特征提取器：学习提取公共特征，域分类器分辨不出公共特征来自哪个域
- 域分类器：正确分类特征提取器提取的特征来自哪个域

  （对抗训练）

  最终目标：当域分类器无法准确预测特征提取器提取的特征来自哪个域➡️特征提取器提取到两个域的公共特征



特征提取器 G_f 将输入样本映射到特征空间，域分类器 G_d 根据这些特征判断样本来自源域还是目标域。通过梯度反转，特征提取器被训练成提取让域分类器无法区分来源的域不变特征，从而实现源域和目标域的特征对齐。

新兴域自适应方向（拓展）

- 噪声环境下的域自适应性
  - 关键挑战：如何减小噪声对模型的影响，以提高模型在目标域上的性能和泛化能力
    - ans：学习更加鲁棒的域不变特征，过滤或降低噪声样本的权重，引入数据增强模拟噪声环境，利用正则化方法防止模型过拟合噪声，以及通过对抗训练或特征对齐方法减小源域和目标域在噪声条件下的分布差异。
      - 在训练模型时，给模型加一些“限制”或“约束”，防止模型把训练数据记得太死，从而提升它在新数据上的泛化能力。
- 通用域自适应
  - 通用域自适应是在目标域类别信息完全未知的情况下的一种域自适应。
    - Closed Set DA 闭集域自适应：源域和目标域的类别集合完全相同
    - Partial DA 部分域自适应：目标域类别集合是源域类别集合的子集
    - Open Set DA 开放集域自适应：源域和目标域有一部分共有类别，但目标域还可能出现源域没有的新类别
    - Universal DA 通用域自适应：不预先假设源域和目标域类别集合之间的具体关系
  - 关键挑战：关于新类别是否出现这个信息需要事先知道➡️要有拒识能力
    - 第一，识别源域和目标域的**共有类别**，并对这些共有类别进行特征对齐
    - 第二，对于目标域中不属于源域共有类别的样本，要把它们识别为**unknown**，避免强行分到源域已有类别中
    - 高置信目标样本用于迁移，低置信或离群样本作为未知类别处理
- 源域无关域自适应
  - 关键挑战：在没有源域信息的情况下，有效地利用目标域数据进行模型适应
    - ans:需要从源域模型中挖掘可迁移的知识，并结合目标域数据的分布结构，通过伪标签、自训练、熵最小化、目标域特征聚类或模型参数调整等方式，使模

型逐渐适应目标域。其核心难点是既要避免由于缺少源域数据而造成的对齐困难，又要防止目标域无标签数据中的伪标签错误不断累积，从而提高模型在目标域上的性能和泛化能力。

- 多源域无关域自适应
  - 关键挑战：有效利用各个源域模型，使它在新的目标域数据中获得好的性能
    - 利用无标签目标域数据观察各个源域模型在目标域上的预测表现。对于在目标域样本上预测置信度高、输出稳定、熵较低、与其他模型预测一致性较强的源域模型，可以认为其与目标域更相关，分配较大的权重；对于预测不确定、与目标域差异较大或容易产生错误预测的模型，则降低其权重，避免负迁移。最后，将多个源域模型按照权重进行集成

06 多示例多标记学习

> Multi-Instance Multi-Label Learning

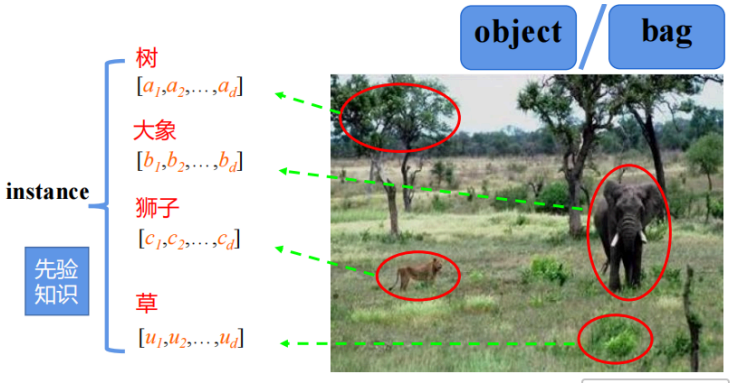
传统的监督学习场景：单示例单标记
样本(x_i, y_i) 被描述为一个d维向量 $x_i \in \mathbb{R}^d$，并拥有一个标记(label) y_i。 学习任务旨在基于n个样本组成的数据集 $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ 上学习 $y = f(x)$ 的映射函数 f，使得 f 能对新样本进行正确的预测。

- 对分类问题：y_i是离散值
- 对回归问题：y_i是连续值
  `object=instance`:输入的对象看作一个不再分割的整体

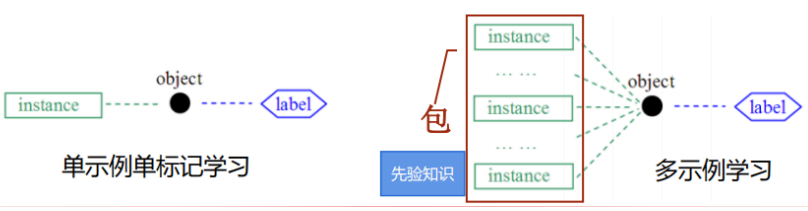
多示例学习

定义

1. 基本定义
   多示例学习（multi-instance learning，MIL）：机器学习的任务不再是基于单一的instance做输出预测，而是基于**多个instance**组成的输入（object）做出预测



- 如果机器学习的任务是预测该图片的标记是否为“树”，那么只要一个包（一张图片）中有一个示例（instance，区域）是树，则输出该图片（包）的标记为“树”；如果一个包（一张图片）中没有任何一个示例（instance，区域）是树，输出该图片（包）的标记为“非树”
- 多示例学习中，模型的预测输出为单标记，标记是包层面的
  - 预测的是包的标记而非包中某一具体的示例的标记



2. 形式化描述
   给定数据集$\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$,多示例学习的目标是学得映射：

$$2^{\mathcal{X}} \to \mathcal{Y}$$

其中$\mathcal{X},\mathcal{Y}$分别为示例空间和标记空间
$X_i \subset \mathcal{X}$为来自同一包的一组示例$\{x_{i1}, x_{i2}, \ldots, x_{in_i}\}, x_{ij} \in \mathcal{X}\ (j = 1, 2, \ldots, n_i)$
n_i为X_i中所包含的示例个数
$y_i \in \mathcal{Y}$是X_i对应的类别标记（单标记）

2. 为什么映射是 $f : 2^X \to Y$?

这里的 X 表示**示例空间**，也就是所有可能的 instance。

而 2^X 表示 X 的所有子集，也就是"一组 instance"。

所以：

$$f : 2^X \to Y$$

意思是：

> 模型不是判断单个 instance 的标签，而是判断一整个 instance 集合，也就是一个"包"的标签。

比如传统监督学习是：

$$f(x) \to y$$

多示例学习是：

$$f(\{x_1, x_2, x_3\}) \to y$$

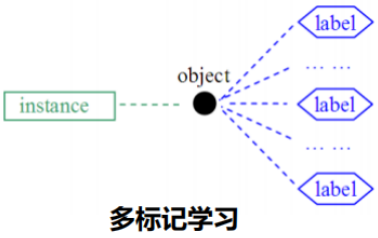
常见思路

- 基于示例的方法
  - 虽然训练时只知道"整封邮件/整个包"的标签，但我们可以尝试从包里的具体示例中找出关键部分，再判断整个包的类别
  - 重点找包里面哪个小区域最关键
- 基于包的方法
  - 不单独判断每个小区域，而是把整张图片/整个包作为整体来比较和分类
  - 重点比较整个包和整个包之间的差异

多标记学习

定义

1. 基本定义
   多标记学习（multi-label learning，MLL）：每个输入由**一个示例**来表示，该示例具有**多个类别标记**；学习的任务是训练后的模型有能力对新示例正确地预测其应具有的多个类别标记



多标记学习

2. 形式化描述
   给定数据集$\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$,多标记学习的目标是学得映射:

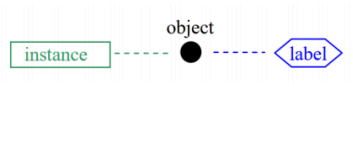
$$\mathcal{X} \to 2^{\mathcal{Y}}$$

   其中$\mathcal{X}, \mathcal{Y}$分别为示例空间和标记空间（多标记）
   $x_i \in \mathcal{X}$ 为样本向量
   $Y_i \subset \mathcal{Y}$ 为样本x_i对应的标记集合，$Y_i = \{y_{i1}, y_{i2}, \ldots, y_{il_i}\}$，　$y_{ik} \in \mathcal{Y}\ (k = 1, 2, \ldots, l_i)$
   l_i 为Y_i中所含类别标记个数

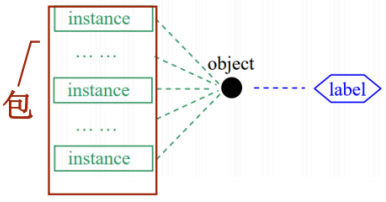
常见思路

- 二分类问题
  - 把一个多标记问题，拆成多个二分类问题
  - 每个标签单独判断，多个标签对应多个而分类器
- 排序问题
  - 按照一定顺序，一个接一个地预测，并把前面预测出的标签结果作为后面分类器的输入

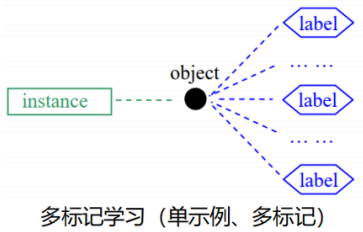
多示例多标记学习



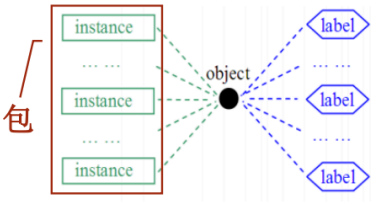
传统监督学习（单示例、单标记）



多示例学习（多示例、单标记）



多标记学习（单示例、多标记）



多示例多标记学习，MIML

背景：在场景分类问题中，一幅图像被表示为多个图像小块(patch)，每个小块对应一个示例，与此同时该图像拥有多个标记

定义

什么是多示例多标记学习？

基本定义：在MIML框架中，一个样本用多个示例来表示并同时拥有多个标记
形式化描述：
给定多示例多标记数据集$\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$,多示例学习的目标是学得映射：

$$2^{\mathcal{X}} \to 2^{\mathcal{Y}}$$

其中$\mathcal{X}, \mathcal{Y}$分别为示例空间和标记空间
$X_i \subset \mathcal{X}$为来自同一包的一组示例$\{x_{i1}, x_{i2}, \ldots, x_{in_i}\}, x_{ij} \in \mathcal{X}\ (j = 1, 2, \ldots, n_i)$
$Y_i \subset \mathcal{Y}$ 为X_i对应的标记集合，$Y_i = \{y_{i1}, y_{i2}, \ldots, y_{il_i}\}$，　$y_{ik} \in \mathcal{Y}\ (k = 1, 2, \ldots, l_i)$

为什么会有多示例多标记学习？

1. 概念更广泛
   - 真实场景中的学习任务往往是复杂的，传统监督学习框架下有可能会导致信息的利用不充分；
2. 多标记学习的不足
   - 只能知道一个样本有哪些标签，但不清楚标签与具体示例/局部区域之间的对应关系。
   - 模型通常会把整张图或多个区域的特征融合起来，然后预测整张图有哪些标签，但是融合之后容易丢失局部信息
     多示例多标记学习框架不但解决了输入空间（表示空间）的复杂性，也解决了输出空间（标记空间）的多义性

常用算法（不要求）

- 转化为多标记学习
  - 首先通过一种映射函数把多示例的样本转化为传统单个向量的表示形式，由此 MIML 学习问题转化为一个多标记学习问题，然后利用已有的多标记学习算法 MLSVM建立每个标记的预测模型。
- 转化为多示例学习
  - 把MIML问题转化为多示例单标记问题，利用多示例算法法MIBoosting进行求解

优点：思想简单、易于实现
缺点：信息损失

性能度量指标

- 海明损失
  - $HammingLoss = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{L}\left(y_i \oplus \hat{y}_i\right)$
  - HammingLoss指标值越小，算法性能越好
- 首标记错误/1-错误率
  - 统计在输出结果中排序第一的标记（预测概率值最大的标签）并不属于真实标记集的次数
  - $OneError = \frac{1}{N}\sum_{i=1}^{N}\left(\arg\max_y f(x_i, y) \notin y_i\right)$

- 指标值越小，算法性能越好

- 覆盖距离
  - 表示所有样本中排序最靠后的真实标签的排序均值
  - Coverage（覆盖距离）越小，预测结果越好
  - $Coverage = \frac{1}{N}\sum_{i=1}^{N} \max_{y \in y_i} \text{rank}(\hat{y}_i)$

$$\text{Coverage} = \frac{1}{N}\sum_{I=1}^{N} \max_{y \in y_i} \text{rank}(\hat{y}_i)$$

例如：假定某样本的预测概率[0.8, 0.4, 0.6, 0.5, 0.2]
真实标记[1, **1**, 0, 0, 0]。
首先对预测概率降序排序，结果为：
[0.8,0.6,0.5,**0.4**,0.2]
容易得出该样本最靠后的真实标签对应概率的序号为**4**

- 排序损失
  - 统计存在的标签的预测概率值比不存在的标签预测概率值小的次数，即按预测概率降序排序后，有多少不存在的标签排序高于存在的标签

例如：假定某样本的预测概率[0.8, 0.4, 0.6, 0.5, 0.2]
真实标记[1, **1**, 0, 0, 0]。
首先对预测概率降序排序，结果为：
[0.8,0.6,0.5,**0.4**,0.2]
答：0 + 2 = 2 个
容易得出：对第一个真实标签，高于它的不存在的标签为0个；对于第二个真实标签，高于它的不存在的标签为2个

- 平均精度
  - 给定某个样本，对预测结果排序（降序），针对每个真实标签，计算排序排在该标签前面（**包括该标签**），且属于真实标签集的概率P。对该样本所有真实标签求该概率值P，然后进行平均，得到精度。所有样本的精度均值便为平均精度。

例如：假定某样本的预测概率[0.8, 0.4, 0.6, 0.5, 0.2]
真实标记[1, **1**, 0, 0, 0]。
首先对预测概率降序排序，结果为：
[0.8,0.6,0.5,**0.4**,0.2]
第一个真实标签：$\frac{1}{1}$
第二个真实标签：$\frac{2}{4}$
容易得出：对于第一个真实标签，序号大于等于它的序号，且为真实标签的概率为1；对于第二个真实标签，概率为0.5,因此，该样本对应的精度为0.75.

07 在线学习

> Online Learning

背景

- 大数据
  - 场景：海量数据+快速且变化的数据流
  - 面临的挑战
    - 实时性要求高
    - 快速适应新数据
- 批量学习（Batching Learning）：以基于**离线批量数据**的算法来进行学习
  - 在批量学习中，系统必须用所有可用数据进行训练。这通常会占用大量时间和计算资源，所以一般是**线下**做的
  - 过程：首先进行训练，然后部署在生产环境且停止学习，随后只使用已经学到的策略
    ➡️离线学习
    - 特点：离线数据集：假设所有数据提前获得
    - 训练/测试阶段分离
    - 独立同分布的假设
  - 局限性：当数据规模非常大，这种模式带来的问题
    - 高开销-计算成本与存储成本
    - 响应慢，实时性差
      - 如果想让一个批量学习系统了解新数据（例如垃圾邮件的新类型），就需要从头训练一个系统的新版本，使用全部数据集（不仅有新数据也有老数据），然后停掉老系统，换上新系统

> 更加的解决方案：在线学习
> 优点：可以降低学习算法的样本存储开销与训练时间开销，实时性强

在线学习

定义

对于批量学习而言，该模式下的算法一般只关心**整个**学习过程**结束后得到的模型性能**，因此学习目标是**最小化总体损失**，关注的指标是训练结束后**在测试集上的总体表现**
而在线学习的学习过程、优化目标及常用的评价指标，与批量学习皆有所不同。

在线学习的特点：

- 流模式的数据集
- 每一步学习：只学习单步数据，旧数据学完即丢掉
- 只计算当前数据的损失，然后即更新得到新模型

在线学习的学习过程

在线学习可被看作是在已知（或部分已知）之前问题的答案与其他可用信息的情况下，询问及被回答一系列未知问题的过程
For $t = 1, 2, \ldots$

- 接收到一个问题 $x_t \in \mathcal{X}$；（是否是垃圾邮件）
- 选择解决方案 w_t，预测问题答案 $\hat{y}_t$；
- 被揭示问题的真实答案 $y_t \in \mathcal{Y}$；
- 遭受损失 $f_t(w_t) = l(\hat{y}_t, y_t)$，并更新解决方案；

在线学习的评价指标

每一步更新时，是**基于单步损失更新模型**，但在线学习可以用**累计损失**来评价模型的整体性能
假设算法共迭代T次，那么累计损失：$\sum_{t=1}^{T} f_t(w_t)$
遗憾：通过比较在线算法的累计损失与最优算法（假设）的损失，将其差值定义为遗憾

$$\text{regret} = \boxed{\sum_{t=1}^{T} f_t(w_t)} - \boxed{\min_{w \in \mathcal{W}} \sum_{t=1}^{T} f_t(w)}.$$

- 累计损失越小，遗憾越小

分析并比较不同的在线学习方法：

- 实验验证
- 理论支持：遗憾界 eg: $\mathcal{O}(log\mathcal{T})$ $\mathcal{O}(\sqrt{\mathcal{T}})$

在线学习 vs 批量学习

	批量学习	在线学习
训练数据	整个数据集 独立同部分	当前步的数据 分布不做限制
学习模式	训练/测试分离	预测-训练迭代进行
计算成本	高	低
存储成本	高	低
更新速度	需重新训练	快速更新一步
关注点	学习结束后的整体表现	更关注当前/最近表现

分类

- 完全信息在线学习

    - 学习器可以观测到完整的损失函数$f_t(\cdot)$，因此可以利用函数的信息，比如梯度，来更新模型
- 部分信息在线学习/赌博机在线学习
    - 学习器通常只能观测到损失函数在所选决策 w_t 上的值$f_t(w_t)$，不能观测到损失函数在其它决策上的数值
    - 困境：由于观测信息不充分，学习器面临着**探索**与**利用**之间的折中问题
        - 一方面，利用意味着根据目前所观测到的数据，选择能得到更高奖励的摇臂
        - 另一方面，由于目前观测到的数据并不充分，不足以让学习器做出最佳决策，此时探索意味着不要轻易的做出最佳决策，而是多去尝试不同的决策。
        - 解决方案：置信上界算法
            - 核心思想：每一步迭代中，为每个摇臂维持一个奖励的置信上界，算法通过选择具有有最大置信上界的摇臂，从而自动的在探索和利用之间进行折中

可以这样记：

完全信息：知道所有选择的损失

部分信息：只知道自己选的那个损失

比如第 t 轮有 3 个决策 A、B、C：

类型	选择 A 后能看到什么
完全信息在线学习	A、B、C 三个决策的损失都能看到
部分信息在线学习	只能看到 A 的损失，看不到 B、C 的损失

持续学习

> （Continual Learning）也叫做增量学习，终身学习）

持续学习是在线学习的一种特殊范式，可以理解为一种在线的批量学习，但持续学习的意义更广

> 任务层面上的在线学习，在每个任务内进行批量学习

关键问题与挑战：灾难性遗忘

	批量学习	在线学习	持续学习
训练数据	整个数据集 独立同部分	当前步的数据 分布不做限制	当前批数据 分布不做限制
学习模式	训练/测试分离	预测-训练迭代进行	训练/测试分离且在任务层面上迭代进行
计算成本	高	低	适中
存储成本	高	低	适中

	批量学习	在线学习	持续学习
更新速度	需重新训练	快速更新一步	适中
关注点	学习结束后的整体表现	更关注当前/最近表现	整体表现

08 元学习

> Meta Learning

为什么（背景与动机）



传统学习：面对新任务，使用一个手动设计的算法从头学起
元学习：希望从众多学习过的任务中获得经验，提高学习算法自身的能力

- 元学习的核心思想是“**学会学习**”。与传统学习面对新任务时从头训练不同，元学习通过在多个已学习任务中积累经验，学习一种更有效的学习策略或初始化方式，从而提高模型在新任务上的快速适应能力。它的优势在于能够减少对大量标注数据的依赖，并提升模型面对新任务时的泛化能力。

是什么

元学习，或者说“**学习如何学习**”，是一门系统性研究的科学。它通过观察不同机器学习方法在大**量学习任务**上的表现，然后从这些经验或元数据中学习，使模型能够比通常情况更快地学习新任务。
元学习包含所有利用**先前**学习**经验**来更快学习**新任务**的系统

元学习的训练环境：**小样本任务**（few-shot task）
元学习不是直接拿一堆图片训练一个分类器，而是把训练过程组织成很多个“小任务 task”，让模型在任务中学习“如何快速学习”。
符号化：
$x \in \mathcal{X}$：输入数据（input data）
$y \in \mathcal{Y}$：学习目标（supervision target）

- 一个小样本学习任务 T 可以由数据集 D_T 表示：

$$D_T = (D_{train}, D_{test})$$
$$D_{train} = (x_i, y_i)_{i=1}^{M_{train}}, \quad D_{test} = (x_j)_{j=1}^{M_{test}}$$

- 任务 T 的训练集 D_{train} 中，通常包含 N 个类，每个类 K 个样本，即数据集中可用于训练的样本个数为 $M_{train}=NK$，这样的小样本任务又叫做 $N-way\ K-shot\ task$。
- 小样本学习的目标是学习到一个目标预测函数 $f\in\mathcal{F}:\mathcal{X}\to\mathcal{Y}$，以此来处理测试数据 D_{test}。
  - $\mathcal{F}$:候选函数集合
    小样本学习不是只训练一个固定分类器，而是**训练模型在一个个小任务上快速学习。**每个任务都有自己的训练集和测试集，训练集中只有少量标注样本。模型需要利用这些少量样本，快速学到一个预测函数，然后对该任务的测试样本进行分类。

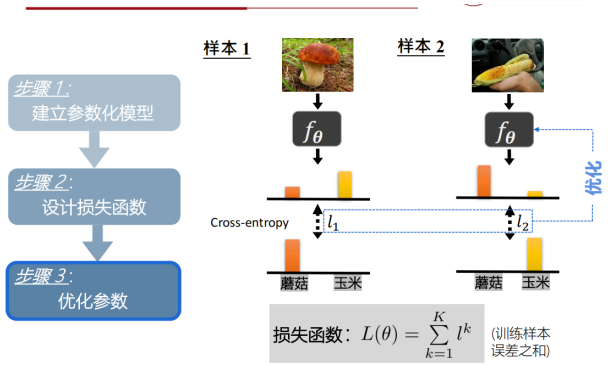
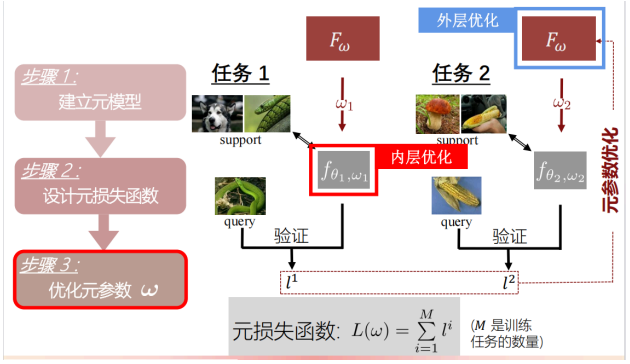


元学习的学习经验：元知识（meta-knowledge w）

- 元知识：模型从大量已学习任务中总结出的可迁移经验，用于帮助模型快速适应新任务

学习范式	主要区别
元学习 Meta Learning	核心是算法层面的学习，通过端到端网络实现手工流程的自动化
自动机器学习 Auto ML	将机器学习流程中传统人工操作的环节自动化，例如数据预处理、算法筛选等
迁移学习 Transfer learning	属于一个研究领域，目标是将源任务知识迁移至目标任务 作为方法论，通过在源任务上常规学习提取先验知识
多任务学习 Multi-task learning	联合学习多个相关任务，依靠参数共享带来正则化增益效果

如何学

常规机器学习	元学习
	

元学习不是只学会一个具体任务，而是学会一种“快速适应新任务的方法”

- 元学习一般遵循双循环框架（a two-loop framework）：

$$\omega^* = \arg\min_{\omega} \sum_{i=1}^{M} L^{\text{meta}}\left(\theta^{*(i)}(\omega), D_{\text{train}}^{\text{query}(i)}\right) \quad \text{外层（outer loop）}$$

$$\text{s.t.} \quad \theta^{*(i)}(\omega) = \mathcal{F}\left(\theta, \omega, D_{\text{train}}^{\text{support}(i)}\right) \quad \text{内层（inner loop）}$$

- 给定新任务 j，元测试过程符号化为：

$$\theta^{*(j)}(\omega) = \mathcal{F}\left(\theta, \omega^*, D_{\text{test}}^{\text{support}(j)}\right)$$

最终，我们使用 f_{θ^*} 对 $D_{\text{test}}^{\text{query}(j)}$ 中的样本做出预测。

对于每个任务 i，先用 support set 经过函数 F 得到任务适应后的参数：

$$\theta^{*(i)}(\omega)$$

然后再用这个参数去 query set 上计算元损失：

$$L^{meta}$$

最后把所有任务的 query 损失加起来：

$$\sum_{i=1}^{M} L^{meta}$$

外层优化的目标就是找到最好的元参数：

$$\omega^*$$

让所有训练任务经过快速适应后，在 query set 上表现都比较好。

元学习训练时，每个任务都被分成 support set 和 query set。内层循环先用 support set 让模型快速适应当前任务，得到任务特定参数；外层循环再用 query set 评估适应后的效果，并优化元参数ω。经过许多任务训练后，模型学到的是一种快速学习能力。测试时，面对新任务，只需要用少量 support 样本进行适应，就可以对 query 样本进行预测。

两大分支方法：

- 基于梯度的元学习：学习优化过程的超参数
- 基于度量的元学习：学习“比较”来解决小样本任务

09 联邦学习

> Federated Learning

背景-人工智能中的数据困境

数据孤岛：有很多数据，但是无法结合使用

- 数据往往属于不同组织，不同部门，想要将这些数据存储到一个中心节点往往是很难做到的
- 人们对于用户隐私和数据安全保护的关注度也在不断提高，用户更加关注他们的隐私信息是否未经自己许可，便被他人出于商业或者政治目的利用

联邦学习

> 联邦学习是一种加密的分布式机器学习技术

定义

定位：联邦学习是隐私保护下机器学习的可实现路径以及“数据孤岛问题”的解决方案
允许从**跨数据所有者分布的数据**中构建集合模型，提供了跨企业的数据使用和模型构建蓝图，适用于B2B和B2C等业务，可被广泛应用于各种领域

联邦学习和分布式学习的关系

1. **用户对于自己的设备和数据有绝对的控制权**。用户可以随时让自己的设备停止参与计算和通信，这就相当于联邦的组成部分，每个邦都有独立权和自治权。
2. **联邦学习的用户节点设备不够稳定**。参与联邦学习的用户的节点往往都是手机，pad或者智能家居这些设备。
3. **联邦学习的通信代价非常大**，其通信代价远大于计算代价。
4. 传统的分布式机器学习技术在数据分布上是一致的（独立同分布），但在联邦学习中所表现的是**数据独立且分布不同**。
5. **联邦学习的节点负载不平衡**。

联邦学习的定义

- 有**两个或以上**的联邦学习参与方协作构建一个**共享的**机器学习模型。每一个参与方都拥有若干能够用来训练模型的训练数据
- 在联邦学习的训练过程中，每一个参与方拥有的数据都不会离开该参与方，即**数据不离开数据拥有者**
- 联邦学习模型相关的信息能够以**加密**方式在各方之间进行传输和交换，并且需要保证任何一个参与方都不能推测出其他方的原始数据

- 联邦学习模型的性能要能够充分逼近理想模型（是指通过将所有训练数据集中在一起并训练获得的机器学习模型）的性能

分类

根据训练数据在不同参与方之间的数据特征空间分布和样本空间的分布情况：

- 横向联邦学习
  - 多个机构拥有的数据“特征空间和标签空间相同”，但是“用户/样本 ID 不同”。它们**不能直接共享原始数据**，于是通过联邦学习共同训练一个模型
  - 可以将横向联邦学习的条件总结如下：

    $$\mathcal{X}_i = \mathcal{X}_j, \quad \mathcal{Y}_i = \mathcal{Y}_j, \quad I_i \neq I_j, \quad \forall D_i, D_j, \quad i \neq j,$$

    式中 D_i 和 D_j 分别表示第 i 方和第 j 方拥有的数据集。假设两方的数据特征空间和标签空间对，即 $(\mathcal{X}_i, \mathcal{Y}_i)$ 和 $(\mathcal{X}_j, \mathcal{Y}_j)$ 是相同的。但是假设两方的客户 ID 空间，即 I_i 和 I_j 是没有交集的或交集很小。
  - 架构
    - 客户-服务器（client-server）架构
    - 对等（Peer-to-Peer，P2P）网络架构
  - 典型例子：医院间联邦学习
- 纵向联邦学习
  - 在数据集上具有相同的样本空间，不同的特征空间的参与方组成的联邦学习
  - 对于纵向的联邦学习系统，我们有：

    $$\mathcal{X}_i \neq \mathcal{X}_j, \quad \mathcal{Y}_i \neq \mathcal{Y}_j, \quad I_i = I_j, \quad \forall D_i, D_j, \quad i \neq j,$$

    式中，$\mathcal{X}$ 表示特征空间；$\mathcal{Y}$ 表示标签空间；I 是样本ID空间；D 表示由不同参与方拥有的数据集。纵向联邦学习的目的是，**通过利用由参与方收集的所有特征，协作地建立一个共享的机器学习模型。**
  - 典型示例：银行和电商的联邦学习
- 联邦迁移学习
  - 特点
    - 参与方的数据集之间可能只有少量的重叠样本和特征
    - 数据集的分布情况可能差别很大
    - 数据集的规模可能差异巨大
    - 某些参与方可能只有数据，没有或只有很少的标注数据
  - 目的：帮助只有**少量数据**（如较少重叠的样本和特征）和**弱监督**（较少标记）的应用建立有效且精确的机器学习模型，并且遵守数据隐私和安全条例的规定
  - 联邦迁移学习旨在为以下场景提供解决方案：

    $$\mathcal{X}_i \neq \mathcal{X}_j, \quad \mathcal{Y}_i \neq \mathcal{Y}_j, \quad I_i \neq I_j, \quad \forall \mathcal{D}_i, \mathcal{D}_j, \quad i \neq j,$$

    式中，$\mathcal{X}_i$ 和 $\mathcal{Y}_i$ 分别表示第 i 方的特征空间和标签空间；I_i 表示样本空间；$\mathcal{D}_i$ 表示第 i 方拥有的数据集。

根据数据端的状态不同：

- 跨孤岛的联邦学习
- 跨设备的联邦学习