第十六章　迁移学习

16.1　迁移学习的概述

16.1 迁移学习的概述

- 迁移学习是一种通过在一个任务上学到的知识来改善在另一个相关任务上的性能的方法, 模型在一个源领域上训练, 然后通过迁移学习的方法适应到一个不同但相关的目标领域上, 常见于数据稀缺或者在新领域中数据收集成本较高的场景. 相较于在线学习和并行计算处理相同的任务, 迁移学习侧重学习知识在不同任务间的应用.

- 迁移学习指的是通过从现有领域中迁移知识, 从而有效且高效地实现目标的过程. 迁移学习的概念最初诞生于心理学和教育学, 心理学家所说的“学习迁移”, 指的是一个学习过程对另一个学习过程的影响. 它也常常出现在我们的日常生活中, 例如, 如果我们会打羽毛球, 那么我们就可以学习打网球,因为打羽毛球和网球有一些相似的策略和技巧; 如果我们会下中国象棋, 那么我们就可以借鉴它与国际象棋之间的相似规则来学习下国际象棋等.

- 首先引入域这一重要概念, 域是执行迁移学习的主体, 它由数据和生成数据的分布两部分组成.我们经常用 $\mathcal{D}$ 来表示一个域, 域的样本可以表示为输入 $\boldsymbol{X}$ 和输出 Y, 其概率分布表示为 $P(\boldsymbol{X}, Y)$, 表明来自 $\mathcal{D}$ 的数据服从分布: $(\boldsymbol{X}, Y) \sim P(\boldsymbol{X}, Y)$. 使用 $\mathcal{X}$ 和 $\mathcal{Y}$ 分别表示特征空间和标签空间,对于任意样本 $(\boldsymbol{X}_i, Y_i)$, 有 $\boldsymbol{X}_i \in \mathcal{X}$, $Y_i \in \mathcal{Y}$, 一个域就可以定义为 $\mathcal{D}=\{\mathcal{X}, \mathcal{Y}, P(\boldsymbol{X}, Y)\}$.

16.1 迁移学习的概述

- 在迁移学习中, 至少有两个域：一个是拥有丰富知识的域, 也就是我们迁移的域, 另一个是我们想要学习的域. 前者称为源域, 后者称为**目标域**, 我们经常用下标 s 和 t 来表示它们, $\mathcal{D}_s$ 是源域, $\mathcal{D}_t$ 是目标域. 当 $\mathcal{D}_s \neq \mathcal{D}_t$ 时,$\mathcal{X}_s \neq \mathcal{X}_t$, $\mathcal{Y}_s \neq \mathcal{Y}_t$ 或 $P_s(\boldsymbol{X}, Y) \neq P_t(\boldsymbol{X}, Y)$.

- **定义 16.1** (迁移学习) 给定一个源域 $\mathcal{D}_s = \{\boldsymbol{X}_i, Y_i\}_{i=1}^{n_s}$ 和一个目标域 $\mathcal{D}_t = \{\boldsymbol{X}_j, Y_j\}_{j=1}^{n_t}$ 目标域用 ℓ 评估, 其中 $\boldsymbol{X}_i \in \mathcal{X}$, $Y_i \in \mathcal{Y}$. 迁移学习的目标是利用源域数据学习一个预测函数 $f: \boldsymbol{X}_t \to Y_t$ 使 f 在目标域上达到最小预测风险:

$$f^* = \underset{f}{\operatorname{argmin}} E_{(\boldsymbol{X},Y)\in\mathcal{D}_t} \ell\left(f\left(\boldsymbol{X}\right), Y\right), \tag{16.1.1}$$

  - 当以下三个条件之一成立时:
  - (1) 不同的特征空间, 即 $\mathcal{X}_s \neq \mathcal{X}_t$;
  - (2) 不同的标签空间, 即 $\mathcal{Y}_s \neq \mathcal{Y}_t$;
  - (3) 具有相同特征和标签空间的不同概率分布, 即 $P_s(\boldsymbol{X}, Y) \neq P_t(\boldsymbol{X}, Y)$.

16.1.1 分布散度的度量

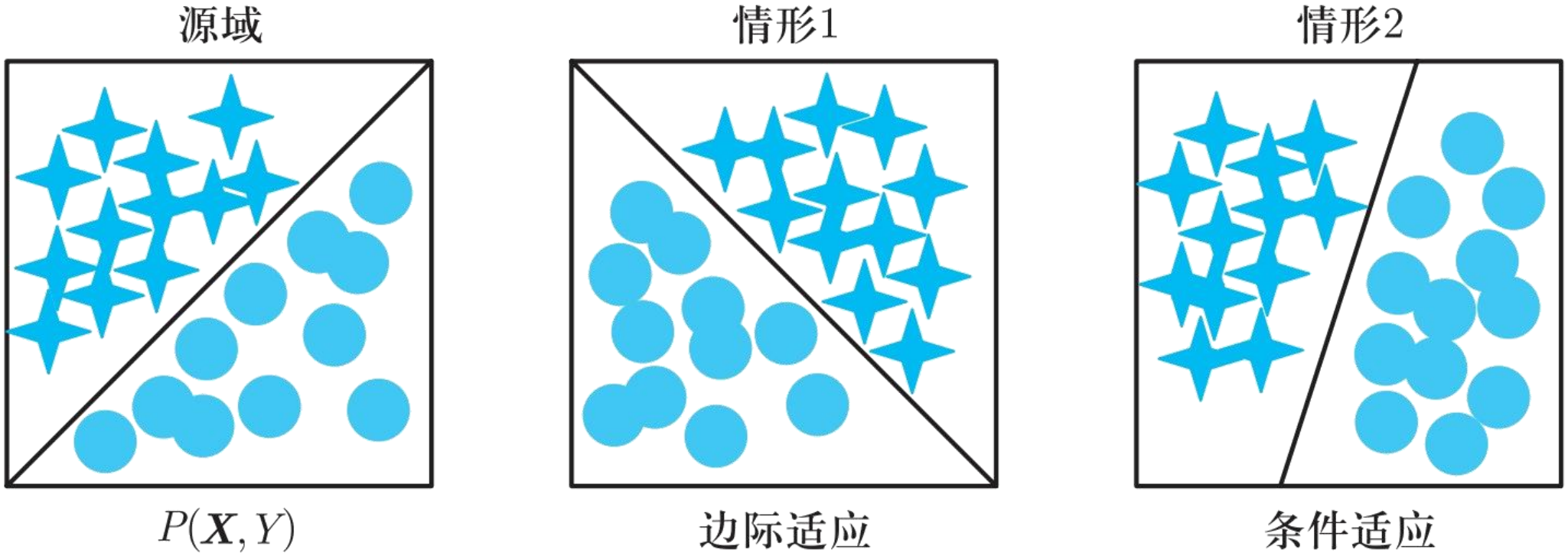
- 迁移学习的目标是开发算法来更好地利用现有知识并促进目标领域的学习，为了利用现有的知识，关键是找到两个领域之间的相似性，而相似性往往通过测量两个域之间的分布散度来计算.
- 根据概率论的基本知识，联合分布、边际分布和条件分布之间存在关系：

$$P(\boldsymbol{X}, Y) = P(\boldsymbol{X}) P(Y|\ \boldsymbol{X}) = P(Y) P(\boldsymbol{X}|\ Y). \tag{16.1.2}$$

- 根据该公式，我们可以通过概率分布匹配的分类法将现有的迁移学习算法分为以下几类：
  - (1) 边际分布适应 (MDA: Marginal Distribution Adaption)
  - (2) 条件分布适应 (CDA: Conditional Distribution Adaption)
  - (3) 联合分布适应 (JDA: Joint Distribution Adaption)
  - (4) 动态分布适应 (DDA: Dynamic Distribution Adaption)

16.1.1 分布散度的度量

- 我们在图 16.1 中进一步说明了什么是边际分布、条件分布和联合分布,以便读者更直观地理解. 显然, 当目标域为图 16.1 中的情形 1 时, 由于边缘分布与全局不同, 我们应该更多地考虑匹配边缘分布; 当目标域是情形 2 时,我们应该更多地考虑条件分布, 因为它们从整体上看是相同的, 只是与每个类的分布不同.



16.1.2 分布散度的统一表示

- 表 16.1 中给出了各种分布匹配方法的假设和问题, 其中函数 $D(\cdot,\cdot)$ 表示分布匹配度量函数, 我们将其作为预定义函数. 从该表中可以清楚地看到,随着假设的变化, 问题的定义也不同. 动态分布适应 (DDA) 是最一般的情况,通过改变平衡因子μ 的值, 可以很容易地将其表述为其他情况:
  - (1) 当 $\mu = 0$ 时, 为边际分布适应;
  - (2) 当 $\mu = 1$ 时, 为条件分布适应;
  - (3) 当 $\mu = 0.5$ 时, 为联合分布自适应.

方法	假设	问题
边际分布适应	$P_s(Y \mid \boldsymbol{X}) = P_t(Y \mid \boldsymbol{X})$	$\min\ D(P_s(\boldsymbol{X}), P_t(\boldsymbol{X}))$
条件分布适应	$P_s(\boldsymbol{X}) = P_t(\boldsymbol{X})$	$\min\ D(P_s(Y \mid \boldsymbol{X}), P_t(Y \mid \boldsymbol{X}))$
联合分布适应	$P_s(\boldsymbol{X}, Y) \neq P_t(\boldsymbol{X}, Y)$	$\min\ D(P_s(\boldsymbol{X}), P_t(\boldsymbol{X})) + D(P_s(Y \mid \boldsymbol{X}), P_t(Y \mid \boldsymbol{X}))$
动态分布适应	$P_s(\boldsymbol{X}, Y) \neq P_t(\boldsymbol{X}, Y)$	$\min\ (1-\mu) D(P_s(\boldsymbol{X}), P_t(\boldsymbol{X})) + \mu D(P_s(Y \mid \boldsymbol{X}), P_t(Y \mid \boldsymbol{X}))$

16.1.2 分布散度的统一表示

- 平衡因子 μ 的估计通常有两种方法: 随机猜测法 (random guess) 和最小最大值平均法 (min-max averaging), 但这两种方法需要烦琐的计算, 也不能从理论上很好地解释. Wang et al.(2018b) 和 Wang et al.(2020) 提出了迁移学习的动态分布适应方法, 并首次提出了 μ 的精确估计. 他们利用分布的全局和局部性质来计算 μ, Ben-David et al.(2007) 采用$\mathcal{A}$-距离 ($\mathcal{A}$-distance)作为基本的分布度量. $\mathcal{A}$-距离可以看作是利用二分类器的误差对两个域进行分类.

- 形式上, 我们将 $\varepsilon(h)$ 表示为分类器 h 对两个域 $\mathcal{D}_s$ 和 $\mathcal{D}_t$ 进行分类的误差, 则可以将 $\mathcal{A}$-距离定义为

$$d_{\mathcal{A}}\left(\mathcal{D}_s, \mathcal{D}_t\right) = 2\left(1 - 2\varepsilon\left(h\right)\right). \tag{16.1.3}$$

- 我们可以直接用方程来计算两个域之间边际分布的 $\mathcal{A}$-距离 d_M, 而对于条件分布的 $\mathcal{A}$-距离, 用 $d_c = d_{\mathcal{A}}\left(\mathcal{D}_s^{(c)}, \mathcal{D}_t^{(c)}\right)$ 表示 c 类上的 $\mathcal{A}$-距离, 其中$\mathcal{D}_s^{(c)}$和 $\mathcal{D}_t^{(c)}$ 是 c 类的样本. 因此, 可以计算 μ 的值为

$$\hat{\mu} = 1 - \frac{d_M}{d_M + \sum_{c=1}^{C} d_c}. \tag{16.1.4}$$

16.1.2 迁移学习的统一框架

- 基于机器学习中结构风险最小化的原理，我们建立了统一的迁移学习框架：
- **定义 16.2** (迁移学习的统一框架) 给定一个有标记的源域 $\mathcal{D}_s = \{X_i, Y_i\}_{i=1}^{n_x}$ 和一个未标记的目标域 $\mathcal{D}_t = \{\boldsymbol{X}_j\}_{j=1}^{n_t}$ 它们的联合分布不同，即 $P_s(\boldsymbol{X}, Y) \neq P_t(\mathbf{X}, Y)$，用 $f \in \mathcal{H}$ 表示目标函数，$\mathcal{H}$ 是它的假设空间. 迁移学习的统一框架为

$$f^* = \arg\min_{f \in \mathcal{H}} \frac{1}{n_s} \sum_{i=1}^{n_s} \ell\left(v_i f\left(\boldsymbol{X}_i\right), Y_i\right) + \lambda R\left(T\left(\mathcal{D}_s\right), T\left(\mathcal{D}_t\right)\right), \tag{16.1.5}$$

  - 其中：(1) $v = (v1, \cdots, v_{ns}) \in \mathbf{R}^{n_s}$ 表示源样本的权重，$v_i \in [0, 1]$，n_s 是源域样本数；
  - (2) $T(\cdot)$ 是两个域上的特征变换函数；
  - (3) 当引入权重 vi 时，可以使用加权平均来代替平均值的计算.
- 我们使用 $R(T(\mathcal{D}_s), T(\mathcal{D}_t))$ 代替结构风险最小化的正则化 $R(f)$，以更好地拟合迁移学习问题. 在这个统一的框架下，迁移学习问题可以概括为寻找最优正则化泛函，这也意味着与传统的机器学习相比，迁移学习更关注源域和目标域之间的关系.

16.1.2 迁移学习的统一框架

- 这个统一的框架总结了大多数迁移学习算法, 具体来说, 我们可以在式(16.1.5) 中赋予 v_i 和 T 不同的值或函数改变它的形式, 可以变换得到三种基本的迁移学习算法:
  - (1) 实例加权方法 (instance weighting methods)：该方法的关键在于学习样本权重 v_i;
  - (2) 特征变换方法 (feature transformation methods)：这种方法适合实际情况 $v_i = 1, \forall i$, 其目标是学习一个特征变换函数 T 以减小正则化损失 $R(\cdot, \cdot)$;
  - (3) 模型预训练方法 (model pre-training methods)：这种方法对应于 $v_i = 1, \forall i, R(T(\mathcal{D}_s), T(\mathcal{D}_t)) = R(\mathcal{D}_t; f_s)$, 其目标是设计策略来正则化源函数 fs并对目标域进行微调.
- 这三种算法可以总结现有文献中大多数流行的迁移学习方法, 我们将在后面的章节中详细介绍它们, 在这里只对它们做一个简短的介绍.

16.2　实例加权方法

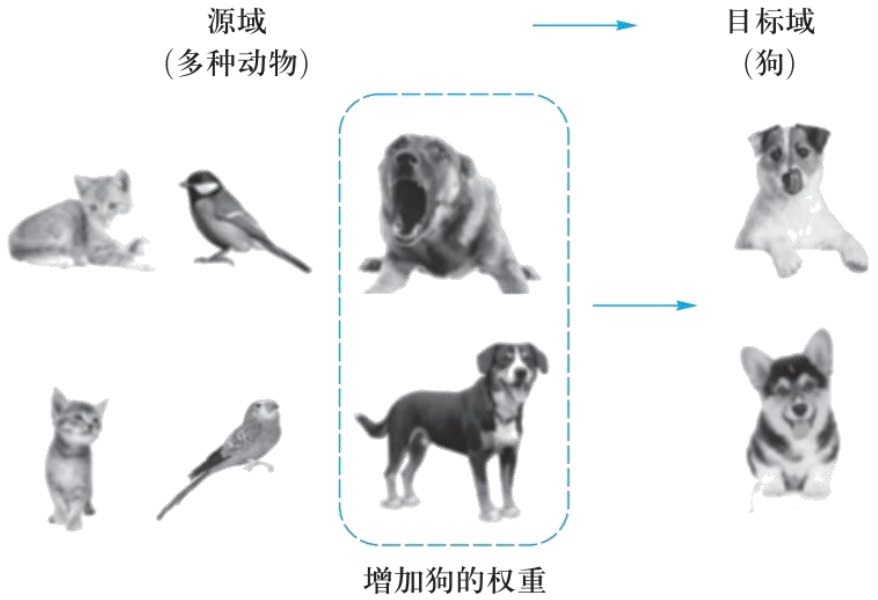
16.2 实例加权方法

- 实例加权方法是迁移学习中最有效的方法之一, 从技术上讲, 任何加权方法都可用来评估每个实例的重要性. 本节中, 我们主要关注两种基本方法：实例选择方法和权重自适应方法. 这两种方法在现有的迁移学习研究中得到了广泛的应用, 也成了更复杂系统的基本模块.

16.2.1 问题定义

- 迁移学习的核心思想是减少源域和目标域之间的分布差异, 那么, 实例加权方法能为实现这个目标做些什么呢? 由于迁移学习中样本的维数和数量通常非常大, 因此不可能直接估计 $P_s(\boldsymbol{X})$ 和 $P_t(\boldsymbol{X})$, 相反, 我们可以从标记的源域中选择一些样本, 构成一个子集, 使该子集的分布与目标域相似, 然后, 使用传统的机器学习方法建立模型, 这种方法的关键是如何设计选择标准. 另一方面, 数据选择可以与如何设计样本加权规则完全相同. (数据选择可以看作是加权的特殊情况. 例如, 我们可以很容易地使用权重 1 和 0 来指示我们是否选择了一个样本.)

- 图 16.2 显示了实例加权方法的基本思想: 有一些动物属于不同的类别,如狗、猫和鸟类, 但在目标域中只有一个主要的类别 (狗). 在迁移学习中, 为了使源域和目标域更相似, 我们可以设计加权策略来增加狗类的权重, 即在训练中赋予它更大的权重.



16.2.1 问题定义

- **定义 16.3** (迁移学习的实例加权) 给定一个有标签的源域 $\mathcal{D}_s = \{\boldsymbol{X}_i, Y_i\}_{i=1}^{n_x}$ 和一个无标签的目标域 $\mathcal{D}_t = \{\boldsymbol{X}_j\}_{j=1}^{n_t}$ 且它们的联合分布不同, 即 $P_s(\boldsymbol{X}, Y) = P_t(\boldsymbol{X}, Y)$, 设向量 $\boldsymbol{v} \in \mathbf{R}^{n_s}$ 表示源域中每个样本的权重, 那么,实例加权方法的目标是学习一个最优的加权向量 $\boldsymbol{v}^*$, 这样分布差异就可以最小化:

$$D\left(P_s\left(X, Y \mid \boldsymbol{v}\right), P_t\left(\boldsymbol{X}, Y\right)\right) < D\left(P_s\left(\boldsymbol{X}, Y\right), P_t\left(\boldsymbol{X}, Y\right)\right).$$

  - 然后, 目标的风险可以最小化:

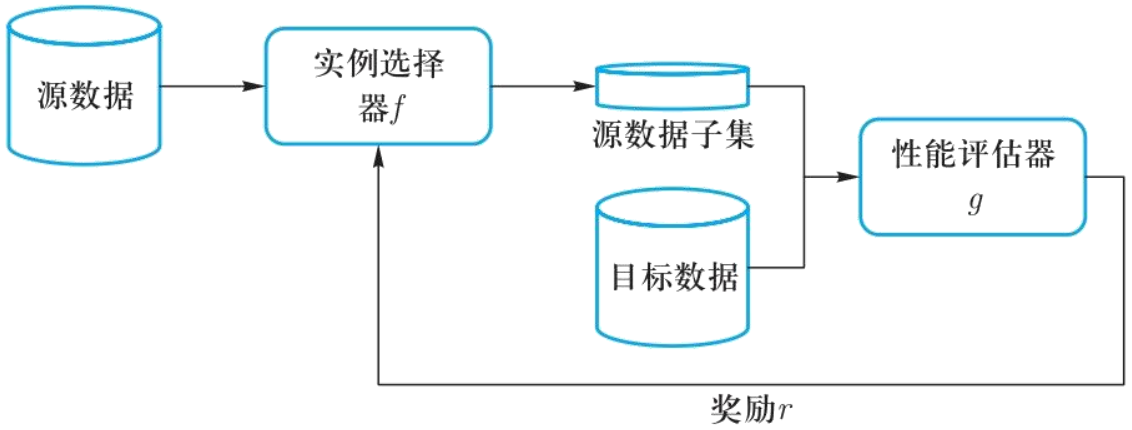
$$f^* = \arg\min_{f \in \mathcal{H}} \frac{1}{n_s} \sum_{i=1}^{n_s} \ell\left(v_i f\left(\boldsymbol{X}_i\right), Y_i\right) + \lambda R\left(\mathcal{D}_s, \mathcal{D}_t\right),$$

  - 其中向量 v 是我们的学习目标.

- 接下来我们将介绍实例选择方法和权重自适应方法, $v_i \in [0, 1]$.

16.2.2 实例选择方法

- 实例选择方法通常假设源域和目标域之间的边际分布是相同的, 即$P_s(\boldsymbol{x}) \approx P_t(\boldsymbol{x})$, 当它们的条件分布不同时, 我们应该设计一些选择策略来选择合适的样本. 事实上, 如果把整个选择过程视为一个决策过程, 那么它可以如图 16.3 所示:



- 本过程主要由以下各个模块组成:
  - (1) 实例选择器 f, 它是从源域中选择与目标域分布相似的样本子集.
  - (2) 性能评估器 g, 即评估所选子集与目标域之间的分布差异.
  - (3) 奖励 r, 即为被选择的子集提供奖励, 然后可以调整其选择过程.

16.2.2 实例选择方法

- 上述过程可以看作是强化学习中的一个马尔可夫决策过程 (MDP)(Sutton 和 Barto, 2018), 因此, 我们有了一个很自然的想法：我们可以通过设计适当的选择器、奖励和性能评估器, 设计强化学习算法来完成这个任务.下面, 根据实例选择方法是否采用强化学习算法, 将其分为两类：基于非强化学习的方法和基于强化学习的方法.
- 第一类是基于非强化学习的方法. 在深度学习之前, 研究人员经常利用传统的学习方法来进行实例选择. 现有的研究工作可以进一步分为三个类：基于距离测量的方法、基于元学习的方法和其他方法.
- 第二类是基于强化学习的方法 (Feng et al., 2018). 该方法将源域划分为几个批次, 然后学习每个批次中每个样本的权重. 值得注意的是, 为了度量域之间的分布差异, 需要首先选择一些有标记的样本作为指导集, 然后, 在批级训练中指导集可以指导权重学习和特征学习过程. 在应用强化学习方法时,定义状态、行动和奖励这三个关键概念是很重要的, 例如, Liu et al. (2019)将这三个概念定义为：
  - (1) 状态由当前批的权向量和特征提取器的参数构成.

16.2.2 实例选择方法

- (2) 行动作为选择过程实现, 因此它是一个二进制向量：0 表示没有选择这个样本, 1 表示选择这个样本.
- (3) 奖励是实现在源域和目标域之间的分布差异.

奖励功能是基于强化学习的方法的关键, 它被实现为

$$r\left(s,a,s'\right)=d\left(\Phi_{B_{j-1}}^{s},\Phi_{t}^{s}\right)-\gamma d\left(\Phi_{B_{j}}^{s'},\Phi_{t}^{s'}\right).$$

- 其中, 下标 t 表示目标域, $d(\cdot,\cdot)$ 是一个分布变量函数, 在实验中可以是最大均值差异 (MMD) 和 Reny 距离, (s, a, s') 表示从状态 s 采取动作 a 成为状态s', B_{j-1} 和 B_j 分别表示第 $j-1$ 次和第 j 次迭代时的批次, Φ 是一个特性.

16.2.3 权重自适应方法

- 与实例选择方法不同, 权重自适应方法假设两个域之间的条件分布相同,而它们的边际分布不同, 即 $P_s(Y|\boldsymbol{X}) \approx P_t(Y|\boldsymbol{X})$ 但 $P_s(\boldsymbol{X}) \neq P_t(\boldsymbol{X})$. 受Jiang 和 Zhai(2007) 经典著作的启发, 我们利用最大似然估计方法解决权重自适应问题. 设 θ 表示模型的可学习参数, 则目标域上的最优超参数可以表示为

$$\theta_t^* = \arg\max_{\theta} \int_X \sum_{Y \in \mathcal{Y}} P_t(\boldsymbol{X}, Y) \log P(Y|\boldsymbol{X}; \theta) \mathrm{d}\boldsymbol{X}.$$

  - 利用贝叶斯定理, 上面的方程可以计算为

$$\theta_t^* = \arg\max_{\theta} \int_X P_t(\boldsymbol{X}) \sum_{Y \in \mathcal{Y}} P_t(Y|\boldsymbol{X}) \log P(Y|\boldsymbol{X}; \theta) \mathrm{d}\boldsymbol{X},$$

  - 其中只有一个未知量 $P_t(Y|\boldsymbol{X})$, 这正是我们的学习目标. 但是, 我们只能利用 $P_s(Y|\boldsymbol{X})$, 通过一些转换, 规避条件分布 $P_t(Y|\boldsymbol{X})$的计算来学习 θ_t^*.构造两个概率之间的关系, 然后利用它们的条件分布几乎相同 $P_s(Y|\boldsymbol{X}) \approx P_t(Y|\boldsymbol{X})$ 这一假设来进行以下转换:

16.2.3 权重自适应方法

$$\theta_t^* \approx \underset{\theta}{\operatorname{argmax}} \int_X \frac{P_t(\boldsymbol{X})}{P_s(\boldsymbol{X})} P_s(\boldsymbol{X}) \sum_{Y \in \mathcal{Y}} P_s(Y|\ \boldsymbol{X}) \log P(Y|\ \boldsymbol{X}; \theta) \mathrm{d}\boldsymbol{X}$$

$$\approx \underset{\theta}{\operatorname{argmax}} \int_X \frac{P_t(\boldsymbol{X})}{P_s(\boldsymbol{X})} \tilde{P}_s(\boldsymbol{X}) \sum_{Y \in \mathcal{Y}} \tilde{P}_s(Y|\ \boldsymbol{X}) \log P(Y|\ \boldsymbol{X}; \theta) \mathrm{d}\boldsymbol{X}$$

$$\approx \underset{\theta}{\operatorname{argmax}} \frac{1}{n_s} \sum_{i=1}^{n_s} \frac{P_t(\boldsymbol{X}_i^s)}{P_s(\boldsymbol{X}_i^s)} \log P(Y_i^s|\ \boldsymbol{X}_i^s; \theta),$$

- 其中 $\frac{P_t(\boldsymbol{X}_i^s)}{P_s(\boldsymbol{X}_i^s)}$ 称为密度比, 用来指导实例加权过程. 我们可以利用密度比来建立源域和目标域之间的关系. 这样, 目标域上的参数就可以表示为

$$\theta_t^* \approx \arg\max_\theta \frac{1}{n_s} \sum_{i=1}^{n_s} \frac{P_t(\boldsymbol{X}_i^s)}{P_s(\boldsymbol{X}_i^s)} \log P(Y_i^s|\ \boldsymbol{X}_i^s; \theta).$$

16.2.3 权重自适应方法

- 从上述分析中知道, 概率密度比可以帮助建立源分布和目标分布之间的关系. 为简单起见, 我们表示密度比为

$$\beta_i = \frac{P_t\left(\boldsymbol{X}_i^s\right)}{P_s\left(\boldsymbol{X}_i^s\right)}.$$

  - 因此, 用向量 $\boldsymbol{\beta}$ 表示概率密度比, 可将目标域内的预测函数重新表述为

$$f^* = \arg\min_{f\in\mathcal{H}} \sum_i^{n_s} \beta_i \ell\left(f\left(\boldsymbol{X}_i\right), Y_i\right) + \lambda R\left(\mathcal{D}_s, \mathcal{D}_t\right).$$

- 上述公式是一种通用的表示算法, 应用于逻辑回归、SVM、特征转换方法(这将在下一节中介绍) 集成等某些特定算法中时可以重新表述为特定公式.比如, 权重自适应方法在与特征转换方法集成时, 如果结合密度比和 MMD的学习, 那么它可以表示为

$$\begin{aligned}\mathrm{MMD}\left(\mathcal{D}_s, \mathcal{D}_t\right) &= \sup_f E_{\mathrm{p}}\left[\frac{1}{n_s}\sum_{i=1}^{n_s}\beta_i f\left(\boldsymbol{X}_i\right) - \frac{1}{n_t}\sum_{j=1}^{n_t} f\left(\boldsymbol{X}_j\right)\right] \\ &= \frac{1}{n_s^2}\boldsymbol{\beta}^{\mathrm{T}} K\boldsymbol{\beta} - \frac{2}{n_t^2}\kappa^{\mathrm{T}}\boldsymbol{\beta} + \text{常数}.\end{aligned}$$

16.2.3 权重自适应方法

- 通过采用核技巧, 上述问题可以表述为

$$\min_{\boldsymbol{\beta}} \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}} K\boldsymbol{\beta} - \kappa^{\mathrm{T}}\boldsymbol{\beta},$$

$$\text{s.t.}\boldsymbol{\beta} \in [0, B], \left| \sum_{i=1}^{n_s} \beta_i - n_s \right| \bullet n_s \varepsilon.$$

- 以上称为核均值匹配算法 (KMM)(Huang et al., 2007), 其中 ε 和 B 是预定义的阈值.
- 有很多的实例权重自适应方法, 值得注意的是, 这类方法可以直接集成到深度学习中, 以学习样本权重.

16.3　统计特征变换方法

16.3 统计特征变换方法

- 在本节中, 我们介绍迁移学习中的统计特征变换方法, 这种方法在近期的深度神经网络研究中非常流行且能够得到很好的结果. 本节首先给出特征变换方法的定义, 然后详细介绍基于最大均值差异 MMD 的特征变换方法和基于度量学习的特征变换方法.

16.3.1 特征变换方法及问题定义

- 我们之前已经接触过许多统计特征, 如均值、方差、假设检验等, 本节着重关注在迁移学习领域广泛应用的几个统计特征.

- **定义 16.4** 给定一个有标签的源域 $\mathcal{D}_s=\{\boldsymbol{X}_i, Y_i\}_{i=1}^{n_x}$ 和一个无标签的目标域 $\mathcal{D}_t=\left\{\boldsymbol{X}_j\right\}_{j=1}^{n_t}$ 且它们的联合分布不同, 即 $P_s(\boldsymbol{X}, Y) \neq P_t(\boldsymbol{X}, Y)$,特征变换的核心是学习特征变换函数 T 从而得到最优预测函数 f:

$$f^*=\arg\min_{f\in\mathcal{H}}\frac{1}{n_s}\sum_{i=1}^{n_s}\ell\left(f\left(\boldsymbol{X}_i\right), Y_i\right)+\lambda R\left(T\left(\mathcal{D}_s\right), T\left(\mathcal{D}_t\right)\right). \tag{16.3.1}$$

- 基于分布散度度量函数的性质, 可以给出如下两类特征变换函数的定义:

- **定义 16.5** 若采用一些预定义的或已有的分布散度度量函数来度量两个分布之间的散度, 并进行特征变换, 称这里的特征变换是显式的特征变换 (explicit feature transformation), 它具有预定义的或已有的分布散度度量$D(\cdot, \cdot)$:

$$f^*=\arg\min_{f\in\mathcal{H}}\frac{1}{n_s}\sum_{i=1}^{n_s}\ell\left(f\left(\boldsymbol{X}_i\right), Y_i\right)+\lambda D\left(T\left(\mathcal{D}_s\right), T\left(\mathcal{D}_t\right)\right). \tag{16.3.2}$$

16.3.1 特征变换方法及问题定义

- 常见的预定义的度量函数有欧几里得距离、余弦相似度、KL 散度和最大均值差异 (maximum mean discrepancy, MMD).

- **定义 16.6** 若分布散度度量函数是由模型经过学习得到的而非预先定义的, 称该特征变换为隐式特征变换 (implicit feature transformation), 它具有可学习的度量函数 Metric $(\cdot, \cdot)$,

$$f^* = \arg\min_{f \in \mathcal{H}} \frac{1}{n_s} \sum_{i=1}^{n_s} \ell\left(f\left(\boldsymbol{X}_i\right), Y_i\right) + \lambda \text{Metric}\left(T\left(\mathcal{D}_s\right), T\left(\mathcal{D}_t\right)\right). \tag{16.3.3}$$

- 上述的隐式特征变换包括度量学习、几何特征对齐和对抗学习.

16.3.2 基于最大均值差异的方法

- 最大均值差异是使用最多的统计距离度量函数之一, 它最初是一种应用于假设检验中有效的双样本检测手段. 本小节中我们主要介绍 MMD 的理论并叙述其在迁移学习中的使用.
- 下面介绍基于 MMD 的迁移学习. 引用式 (16.3.1), 建立 MMD 与特征变换函数 T 之间的关系, 分布散度的一般形式可表示为

$$D\left(\mathcal{D}_s, \mathcal{D}_t\right) \approx (1-\mu) D\left(P_s(\boldsymbol{X}), P_t(\boldsymbol{X})\right) + \mu D\left(P_s(Y|\ \boldsymbol{X}), P_t(Y|\ \boldsymbol{X})\right).$$

- 可以看出, MMD 可以直接用于计算边际分布散度 $D\left(P_s(\boldsymbol{X}), P_t(\boldsymbol{X})\right)$, 这与经典的迁移学习方法迁移成分分析 (TCA) 相对应.
- 我们使用半定矩阵 $\boldsymbol{A}$ 来表示使用 MMD 计算的特征变换矩阵, 那么矩阵 $\boldsymbol{A}$ 即为我们在基于 MMD 的迁移学习方法中的学习目标, 两个边际分布之间的 MMD 可以表示为

$$\mathrm{MMD}\left(P_s(\boldsymbol{X}), P_t(\boldsymbol{X})\right) = \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{X}_i - \frac{1}{n_t} \sum_{j=1}^{n_t} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{X}_j \right\|_{\mathcal{H}_t}^2 . \tag{16.3.4}$$

16.3.2 基于最大均值差异的方法

- 下面首先介绍充分统计量 (sufficient statistics) 的概念：在样本量充分大时, 若存在许多未知的统计量, 则可以选择一些已知的统计量来近似我们的目标量. 我们利用这个概念近似计算目标域上的分布 $P_t(Y \mid \boldsymbol{X})$. 根据贝叶斯公式 $P_t(Y \mid \boldsymbol{X}) = P_t(Y)P_t(\boldsymbol{X} \mid Y)$, 忽略 $P_t(Y)$ 项, 就可以用类条件概率$P_t(\boldsymbol{X} \mid Y)$来近似 $P_t(Y \mid \boldsymbol{X})$, 而目标域是无标签的, 通常采用迭代式的训练策略：首先, 使用 $(\boldsymbol{X}_s, Y_s)$ 来训练一个分类器 (例如 KNN 和逻辑回归等), 从而得到未标记目标域的伪标签 $\hat{Y}_t$, 这个伪标签可以用于迁移学习, 经过特征变换后, 伪标签就能够在以后的迭代中进行更新. 使用这种方法可以计算得到 $P_t(\boldsymbol{X} \mid Y)$ 从而得到两条件分布间的 MMD 如下：

$$\mathrm{MMD}\left(P_s(Y|\ \boldsymbol{X}), P_t(Y|\ \boldsymbol{X})\right) = \sum_{c=1}^{C} \left\| \frac{1}{n_s^{(c)}} \sum_{\boldsymbol{X}_i \in \mathcal{D}_s^{(c)}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{X}_i - \frac{1}{n_t^{(c)}} \sum_{\boldsymbol{X}_j \in \mathcal{D}_i^{(c)}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{X}_j \right\|_{\mathcal{H}}^2,$$

  - 其中 C 表示所有分类的总数, $n_s^{(c)}$ 和 $n_t^{(c)}$分别表示源域和目标域中第 c 类样本集 $\mathcal{D}_s^{(c)}$和 $\mathcal{D}_t^{(c)}$ 中的样本个数. 经求解得到基于 MMD 的迁移学习方法形式为

$$\min \mathrm{tr}\left(A^{\mathrm{T}} XMX^{\mathrm{T}} A\right), \tag{16.3.5}$$

16.3.2 基于最大均值差异的方法

- 其中 tr(·) 表示矩阵的迹, $\boldsymbol{X}$ 是由源域和目标域中的特征所构成的矩阵, $\boldsymbol{M}$是 MMD 矩阵, 计算公式为

$$M = (1-\mu)M_0 + \mu\sum_{c=1}^{C} M_c,$$

- 其中边际 MMD 矩阵与条件 MMD 矩阵计算公式分别为

$$(\boldsymbol{M}_0)_{ij} = \begin{cases} \dfrac{1}{n_s^2}, & \boldsymbol{X}_i, \boldsymbol{X}_j \in \mathcal{D}_s, \\ \dfrac{1}{n_t^2}, & \boldsymbol{X}_i, \boldsymbol{X}_j \in \mathcal{D}_t, \\ [2ex] - \dfrac{1}{n_s n_t}, & \text{其他}. \end{cases} \quad (\boldsymbol{M}_c)_{ij} = \begin{cases} \dfrac{1}{\left(n_s^{(c)}\right)^2}, & \boldsymbol{X}_i, \boldsymbol{X}_j \in \mathcal{D}_s^{(c)}, \\ \dfrac{1}{\left(n_t^{(c)}\right)^2}, & \boldsymbol{X}_i, \boldsymbol{X}_j \in \mathcal{D}_t^{(c)}, \\ -\dfrac{1}{n_s^{(c)} n_t^{(c)}}, & \begin{cases} \boldsymbol{X}_i \in \mathcal{D}_s^{(c)}, \boldsymbol{X}_j \in \mathcal{D}_t^{(c)}, \\ \boldsymbol{X}_i \in \mathcal{D}_t^{(c)}, \boldsymbol{X}_j \in \mathcal{D}_s^{(c)}, \end{cases} \\ 0, & \text{其他}. \end{cases}$$

- 详细的求解步骤可参考相关文献.

16.3.2 基于最大均值差异的方法

- 最小化式 (16.3.5) 的同时需要考虑到其约束条件, 在这里我们考虑特征变换前后的协方差矩阵. 给定样本 $\boldsymbol{X}$, 它的协方差矩阵 $\boldsymbol{S}$ 可表示为

$$\boldsymbol{S}=\sum_{j=1}^{n}\left(X_j-\bar{\boldsymbol{X}}\right)\left(\boldsymbol{X}_j-\bar{\boldsymbol{X}}\right)^{\mathrm{T}}=\sum_{j=1}^{n}\left(X_j-\bar{X}\right)\otimes\left(\boldsymbol{X}_j-\bar{\boldsymbol{X}}\right)=\left(\sum_{j=1}^{n}\boldsymbol{X}_j\boldsymbol{X}_j^{\mathrm{T}}\right)-n\overline{\boldsymbol{X}}\,\overline{\boldsymbol{X}}^{\mathrm{T}},$$

  - 其中 $\bar{\boldsymbol{X}}=\frac{1}{n}\sum_{j=1}^{n}\boldsymbol{X}_j$ 表示样本均值, $\otimes$ 表示外积. 用 $\boldsymbol{H}=\boldsymbol{I}-(1/n)1$ 表示中心矩阵, $\boldsymbol{I}\in\mathbf{R}^{(n+m)\times(n+\mathrm{m})}$ 表示单位矩阵, 则协方差矩阵可表示为

$$\boldsymbol{S}=\boldsymbol{X}\boldsymbol{H}\boldsymbol{X}^{\mathrm{T}}.$$

  - 将 $\boldsymbol{A}$ 代入上式得到方差的最大化表示:

$$\max\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\right)H\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\right)^{\mathrm{T}}.$$

  - 结合 (16.3.5) 式, 可以得到基于 MMD 的迁移学习方法的最终形式为

$$\min\operatorname{tr}\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{M}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\right)+\lambda\|\boldsymbol{A}\|_F^2,\quad\text{s.t.}\quad\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{M}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{I}.\tag{16.3.6}$$

16.3.2 基于最大均值差异的方法

- 其中正则项 $\lambda\|A\|_F^2$ 的引入可以保证问题的良定性, $\lambda > 0$ 是一个超参数.

- 通常使用拉格朗日方法求解该最优化问题, 首先构造拉格朗日函数:

$$L = \mathrm{tr}\left(\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{A}\boldsymbol{X}^{\mathrm{T}} + \lambda\boldsymbol{I}\right)\boldsymbol{A}\right) + \mathrm{tr}\left(\left(\boldsymbol{I} - \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{H}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\right)\Phi\right).$$

  - 令 $\partial \boldsymbol{L}/\partial \boldsymbol{A} = 0$, 得到

$$\left(\boldsymbol{X}\boldsymbol{M}\boldsymbol{X}^{\mathrm{T}} + \lambda\boldsymbol{I}\right)A = \boldsymbol{X}\boldsymbol{H}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\Phi,$$

  - 其中 Φ 为拉格朗日乘子. 利用 Python 或者 MATLAB 容易求解上式得到变换矩阵 $\boldsymbol{A}$.

- 我们使用前文叙述的多次迭代的方法可以使目标域的伪标签更准确, 从而改进最终结果. 同时, 如果将 μ 取为不同的值, 就可以得到相应的边际、条件和联合分布方法的最终形式, 这一部分留给读者完成

- 基于 MMD 的迁移学习的完整步骤总结如下:

  - (1) 输入两个特征矩阵, 使用一个简单的分类器 (如 KNN) 来计算目标域的伪标签;
  - (2) 计算矩阵 $\boldsymbol{M}$ 和 $\boldsymbol{H}$;

16.3.2 基于最大均值差异的方法

- (3) 选择一些常见的核函数 (如线性核、多项核、高斯核等) 来计算核;
- (4) 求解等式 (16.3.6) 得到源域和目标域的变换特征. 取它的前 m 个特征, 即为矩阵 $\boldsymbol{A}$;
- (5) 可以进一步使用多次迭代的方法使伪标签更加准确.

16.3.3 基于度量学习的方法

- 在本节中, 我们介绍如何使用度量学习得到特征变换 T, 也即式 (16.3.3)中的 Metric$(\cdot,\cdot)$.
- 度量学习 (metric learning) 是机器学习中的一个十分重要的研究方向.实际上, 度量两个样本之间的距离涉及分类、回归和聚类等重要问题, 选择一个好的度量可以用来构造好的特征表示从而建立一个更好的模型. 度量学习在计算机视觉、文本挖掘和生物信息学等领域有广泛的应用. 可以说, 如果没有适当的度量, 在机器学习中就没有好的模型. 欧几里得距离、马氏距离、余弦相似度和 MMD 都是度量的例子, 它们都是预定义的距离. 而在某些特定的应用场景中, 这些度量并不能保证我们的模型总是获得最好的表现, 因此可求助于度量学习.
- 度量学习的基本过程是为给定的样本集计算一个更好的距离度量, 使该度量能够反映数据集的重要性质. 这些样本通常包含一些先验知识, 度量学习算法可以基于这些先验知识建立目标函数从而得出一个针对这些样本的更好的度量. 从这个角度来看, 度量学习可以看作是在一定条件下的最优化问题.

16.3.3 基于度量学习的方法

- 度量学习的核心是聚类假设: 属于同一聚类的数据很有可能属于同一个类. 因此, 度量学习重点关注成对的距离, 同时考虑类内距离和类间距离的作用. 为了评估样本之间的相似性, 度量学习采用线性判别分析(LDA) 中的方法来计算类内和类间的距离, 目标是使得类间距离较大, 类内距离较小, 分别用 $S_c^{(M)}$ 和 $S_b^{(M)}$ 分别表示类内和类间的距离, 计算方法如下:

$$S_c^{(M)} = \frac{1}{nk_1}\sum_{i=1}^{n}\sum_{j=1}^{n} P_{ij} d^2\left(\boldsymbol{X}_i, \boldsymbol{X}_j\right),$$

$$S_b^{(M)} = \frac{1}{nk_2}\sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij} d^2\left(\boldsymbol{X}_i, \boldsymbol{X}_j\right),$$

  - 其中 P_{ij} 和 Q_{ij} 分别表示类内和类间距离, 若 $\boldsymbol{X}_i$ 为 $\boldsymbol{X}_j$ 的 k_1 个邻居之一,则 $P_{ij} = 1$, 否则为 0 ; 类似地, 若 $\boldsymbol{X}_i$ 为 $\boldsymbol{X}$ 的 k_2 个邻居之一, 则 $Q_{ij} = 1$,否则为 0 . $d(\cdot, \cdot)$ 的定义将在下文中介绍.

16.3.3 基于度量学习的方法

- 现有的迁移学习方法大多基于一个预定义的距离函数, 比如上一小节介绍的 MMD, 而在一般场合下, 这种度量并不能很好地推广. 通过在迁移学习中使用度量学习的方法, 能够得到更好的距离度量函数. 我们将 MMD 集成到度量学习中, 从而得到以下优化目标函数:

$$J = S_c^{(M)} - \alpha S_b^{(M)} + \beta D_{\mathrm{MMD}}\left(\boldsymbol{X}_s, \boldsymbol{X}_t\right),$$

  - 其中 β 为表示权重的超参数. 令 $M \in \mathbf{R}^{d\times d}$ 为一个半定矩阵, 则样本 $\boldsymbol{X}_i$ 与 $\boldsymbol{X}_j$ 之间的马氏距离可表示为

$$d_{ij} = \sqrt{\left(\boldsymbol{X}_i - \boldsymbol{X}_j\right)^{\mathrm{T}} M \left(\boldsymbol{X}_i - \boldsymbol{X}_j\right)}.$$

- 由于 $\boldsymbol{M}$ 为半定的, 由半定矩阵的性质可分解为 $\boldsymbol{M} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$, 其中 $\boldsymbol{A} \in \mathbf{R}^{d\times d}$, 则上式可化为

$$d_{ij} = \sqrt{\left(\boldsymbol{X}_i - \boldsymbol{X}_j\right)^{\mathrm{T}} M \left(\boldsymbol{X}_i - \boldsymbol{X}_j\right)} = \sqrt{\left(\boldsymbol{A}\boldsymbol{X}_i - \boldsymbol{A}\boldsymbol{X}_j\right)^{\mathrm{T}} \left(\boldsymbol{A}\boldsymbol{X}_i - \boldsymbol{A}\boldsymbol{X}_j\right)}.$$

- 以上结果表明, 要得到马氏距离矩阵 $\boldsymbol{M}$ 等同于找到源域和目标域之间的线性特征变换 $\boldsymbol{A}$. 因此, 我们不需要直接计算 $\boldsymbol{M}$, 而是通过 16.3.2 节的方法计算线性变换矩阵 $\boldsymbol{A}$ (或使用核方法进行非线性变换) 来计算 d_{ij} , 故最优化过程也可仿照 16.3.2 节进行.

16.4　几何特征变换方法

16.4 几何特征变换方法

- 在本节介绍用于迁移学习的几何特征变换方法, 这与 16.3 节中的统计特征变换有所不同. 几何特征变换可以利用潜在的几何特征来获得简洁有效的表示, 并具有显著的性能. 与统计特征相似, 也有许多几何特征. 我们主要介绍三种类型的几何特征变换方法：子空间学习、流形学习和最优传输方法. 这些方法在方法论上有所不同, 但在迁移学习中都很重要.

16.4.1 子空间学习方法

- 在本节介绍用于迁移学习的几何特征变换方法, 这与 16.3 节中的统计特征变换有所不同. 几何特征变换可以利用潜在的几何特征来获得简洁有效的表示, 并具有显著的性能. 与统计特征相似, 也有许多几何特征. 我们主要介绍三种类型的几何特征变换方法：子空间学习、流形学习和最优传输方法. 这些方法在方法论上有所不同, 但在迁移学习中都很重要.
- 几何特征变换方法属于隐含的特征变换方法. 因此, 尽管我们不能直接测量分布散度, 但可以通过应用几何特征变换来降低
- 子空间学习经常假设经过特征变换后源数据和目标数据在子空间中有相似的分布. 在子空间中, 我们可以实行分布对齐和使用传统机器学习方法来建立模型. 对齐的概念有着直观的几何信息：如果来自两个领域的数据是对齐的, 那么认为它们之间的分布差距是最小的. 因此, 子空间学习可用于分布对齐. 子空间对齐 (SA) 是一种经典的子空间学习方法.SA 的目标是找到一种可以进行域对齐的线性特征变换 $\boldsymbol{M}$. 让 $\boldsymbol{X}_s$ 和 $\boldsymbol{X}_t$ 分别表示对源特征 $\boldsymbol{S}$和目标特征 $\boldsymbol{T}$ 进行主成分分析 (PCA) 变换的 d 维特征矩阵, 称为子空间.

16.4.1 子空间学习方法

- 然后, SA 的目标可以表述为

$$F(\boldsymbol{M}) = \| \boldsymbol{X}_s \boldsymbol{M} - \boldsymbol{X}_t \|_F^2 .$$

- 特征变换矩阵 $\boldsymbol{M}$ 的最优解 $\boldsymbol{M}^*$ 可以表示为

$$\boldsymbol{M}^* = \arg\min_{\boldsymbol{M}} F(\boldsymbol{M}).$$

- 因此, 由于子空间学习的正交性, 即 $\boldsymbol{X}_s^T \boldsymbol{X}_s = \boldsymbol{I}$, 可以直接得到上述问题的封闭解

$$F(\boldsymbol{M}) = \| \boldsymbol{X}_s^{\mathrm{T}} \boldsymbol{X}_s \boldsymbol{M} - \boldsymbol{X}_s^{\mathrm{T}} \boldsymbol{X}_t \|_F^2 = \| \boldsymbol{M} - \boldsymbol{X}_s^{\mathrm{T}} \boldsymbol{X}_t \|_F^2 .$$

- 因此, 特征变换矩阵 $\boldsymbol{M}$ 的最优解计算为 $\boldsymbol{M}^* = \boldsymbol{X}_s^T \boldsymbol{X}_t$. 这意味着源域和目标域相同时 (即 $\boldsymbol{X}_s = \boldsymbol{X}_t$), $\boldsymbol{M}^*$ 应为单位矩阵. 我们称 $\boldsymbol{X}_a = \boldsymbol{X}_s \boldsymbol{X}_s^T \boldsymbol{X}_t$ 为目标对齐源坐标系, 它通过下式将源域转换为一个新的子空间,

$$\boldsymbol{S}_a = \boldsymbol{S}\boldsymbol{X}_a .$$

16.4.1 子空间学习方法

- 相似地, 目标域转换为 $\boldsymbol{T}_t = \boldsymbol{T}\boldsymbol{X}_t$. 最终我们使用 $\boldsymbol{S}_a$ 和 $\boldsymbol{T}_t$ 建立机器学习模型, 而不是原始的特征 $\boldsymbol{S}$ 和 $\boldsymbol{T}$. 这使得 SA 在实践中实现起来非常简单.基于 SA, Sun 和 Saenko(2015) 提出了子空间分布对齐 (SDA), 将概率分布适应加入到子空间学习中. 具体来说, SDA 认为除了子空间学习矩阵 $\boldsymbol{G}$, 我们也应该加入一个分布变换矩阵 $\boldsymbol{A}$. SDA 的优化目标被表述为

$$\boldsymbol{M} = \boldsymbol{X}_s \boldsymbol{G} \boldsymbol{A} \boldsymbol{X}_t^G.$$

- 我们得到两个域的变换特征, 然后按照 SA 中类似的步骤建立模型.
- 不同于 SA 和 SDA 只进行一阶对齐, Sun et al. 提出了相关性对齐 (CORAL)来进行二阶对齐. 假设 $\boldsymbol{C}_s$ 和 $\boldsymbol{C}_t$ 分别是源域和目标域的协方差矩阵, 那么CORAL 学习了一个二阶特征变换矩阵 $\boldsymbol{A}$ 来降低它们的分布差异:

$$\min_{\boldsymbol{A}} \| \boldsymbol{A}^{\mathrm{T}} \boldsymbol{C}_s \boldsymbol{A} - \boldsymbol{C}_t \|_F^2,$$

16.4.1 子空间学习方法

- 这样, 源特征和目标特征可以通过下式进行变换

$$z^r = \begin{cases} \boldsymbol{X}^r(\boldsymbol{C}_s + \boldsymbol{E}_s)^{-\frac{1}{2}}(\boldsymbol{C}_t + \boldsymbol{E}_t)^{\frac{1}{2}}, & r = s, \\ [2ex]\boldsymbol{X}^r, & r = t. \end{cases} \tag{16.4.1}$$

- 其中 $\boldsymbol{E}_s$ 和 $\boldsymbol{E}_t$ 分别是与源域和目标域相同大小的单位矩阵. 我们把这一步骤看作每个子空间重新着色的过程, 其中方程 (16.4.1) 通过重新着色去噪后的源特征, 使其与目标分布的协方差对齐.

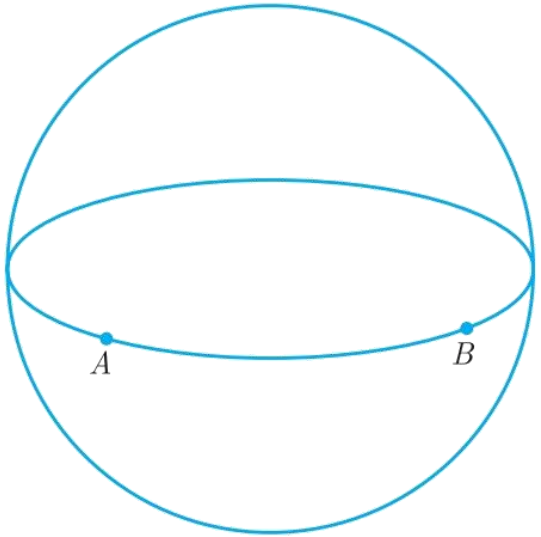
- CORAL 随后被扩展到深度神经网络中, 被称为 **DCORAL(深度CORAL)**(Sun 和 Saenko, 2016). 在 DCORAL 中, CORAL 用于构建网络中的一个适应性损失, 可以替换现有的 MMD 损失. 深度学习中的 CORAL损失定义为

$$\ell_{\text{CORAL}} = \frac{1}{4d^2}\| \boldsymbol{C}_s - \boldsymbol{C}_t\|_F^2,$$

- 其中 d 是特征维度的数量. CORAL 的计算也非常简单, 不需要调整任何超参数. 此外, CORAL 在领域自适应和领域泛化方面也取得了优异的性能. 在本章的实践部分, 我们将展示 CORAL 在比其他方法更简便的情况下, 也能取得优异的性能.

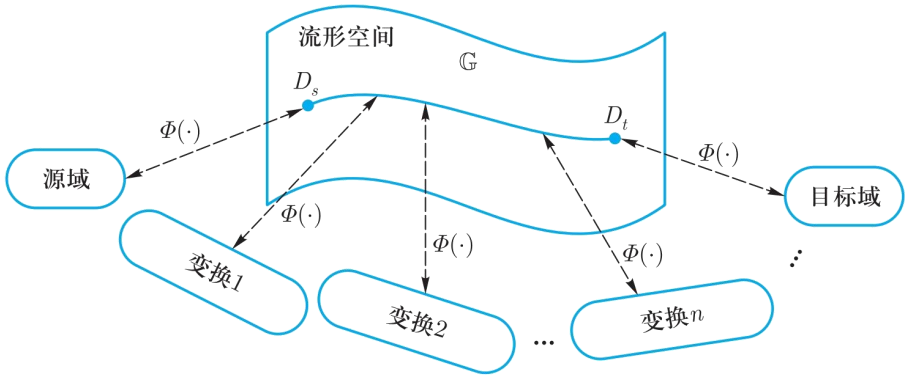
16.4.2 流形学习方法

- 自 2000 年在《科学》杂志首次提出以来 (Seung 和 Lee, 2000), 流形学习已成为机器学习和数据挖掘领域的热门研究话题. 流形学习通常假设当前数据是从高维空间中采样的, 并且具有低维流形结构. 流形是一类几何对象. 一般来说, 我们无法直接从原始数据中观察到隐藏结构, 但可以想象数据位于高维空间中, 其中具有某种可观察的形状. 一个很好的例子是天空中星座的形状. 为了描述所有星座, 我们可以想象它们在天空中具有某种形状, 从而产生了许多受欢迎的星座, 如天琴座和猎户座. 经典流形学习方法包括等距映射、局部线性嵌入和拉普拉斯特征映射等 (Zhou, 2016; Bishop, 2006). 流形学习的核心是利用几何结构简化问题. 距离测量在流形学习中也很重要, 因为我们可以利用几何结构获得更好的距离. 那么, 在流形学习中, 两点之间的最短路径是什么? 在二维空间中, 两点之间的最短距离是线段. 但在三维、四维或 n 维空间 ($n > 4$) 中呢? 实际上, 当我们展开地球空间时, 地球上两点之间的最短路径是直线. 这条直线实际上是一条曲线, 被称为测地线. 一般来说, 测地线距离是任何空间中任意两点之间的最短路径. 例如, 图 16.4 显示了在三维空间中, 球体上两点 A 和 B 之间的最短路径是曲线.



16.4.2 流形学习方法

- 我们熟悉的欧几里得空间也是一种流形结构. 事实上, 惠特尼嵌入定理(Greene 和 Jacobowitz, 1971) 表明, 任何流形都可以嵌入到高维欧几里得空间中, 这使得通过流形进行计算成为可能. 流形学习方法通常采用流形假设(Belkin et al., 2006), 即数据在其流形嵌入空间中通常具有与邻居相似的几何性质.
- 由于流形空间中的数据通常具有良好的几何性质, 可以克服特征失真, 因此可以采用流形学习进行迁移学习. 在许多现有的流形中, Grassmann 流形$\mathbb{G}(d)$ 将原始 d 维空间视为其元素, 从而便于分类器学习. 在 Grassmann 流形中, 特征变换和分布适应通常具有有效的数值形式, 可以容易地解决 (Hamm和 Lee, 2008). 因此, 我们可以使用 Grassmann 流形进行迁移学习 (Gopalanet al., 2011; Baktashmotlagh et al., 2014).
- 基于流形学习的迁移学习受到增量学习的启发：如果一个人想要从当前点移动到另一个点, 他需要一步一步地走. 如果将源域和目标域视为高维空间中的两个点, 那么可以模拟人类的行走过程, 逐步进行特征变换, 直到到达目标域. 图 16.5 简要展示了这一过程. 在该图中, 源域通过特征变换函数 $\Phi(\cdot)$从起始点移动到终点, 以完成流形学习.



16.4.2 流形学习方法

- 早期的方法将这个问题优化为在流形空间中采样 d 个点, 然后构建一个测地线流. 我们只需要找到每个点的变换. 这种类型的方法被称为采样测地线流 (SGF)(Gopalan et al., 2011), 这是第一个提出这种想法的工作. 然而,SGF 有几个局限性：我们应该找到多少个中间点, 以及如何做高效计算？之后, Gong et al. 提出了测地流核 (GFK) 来解决这个问题. 简单来说, 为确定中间点, GFK 采用一种核学习方法来利用测地线上无限个点.
- 我们用 $\mathcal{S}_s$ 和 $\mathcal{S}_t$ 表示主成分分析变换后的子空间. $\boldsymbol{G}$ 看作是所有 d 维空间的集合, 每个 d 维子空间看作是 $\boldsymbol{G}$ 上的一个点. 因此, 两点之间的测地线$\boldsymbol{\Phi}(t) : 0 \leqslant t \leqslant 1$ 是这两点之间的最短路径.
- 若令 $\mathcal{S}_s = \boldsymbol{\Phi}(0)$, $\mathcal{S}_t = \boldsymbol{\Phi}(1)$, 则寻找从 $\boldsymbol{\Phi}(0)$ 到 $\boldsymbol{\Phi}(1)$ 的测地路径等同于将原始特征转换为无限维空间, 并且最终减少域偏移问题. 特别地, 流形空间中的特征表示为 $\boldsymbol{Z} = \boldsymbol{\Phi}(t)^{\mathrm{T}}\boldsymbol{X}$, 经过变换的特征 $\boldsymbol{Z}_i$ 和 $\boldsymbol{Z}_j$ 定义了一个半正定测地线流核：

$$\langle \boldsymbol{Z}_i, \boldsymbol{Z}_j \rangle = \int_0^1 (\boldsymbol{\Phi}(t)^{\mathrm{T}} \boldsymbol{X}_i)^{\mathrm{T}} (\boldsymbol{\Phi}(t)^{\mathrm{T}} \boldsymbol{X}_j) \, \mathrm{d}t = \boldsymbol{X}_i^{\mathrm{T}} \boldsymbol{G} \boldsymbol{X}_j .$$

16.4.2 流形学习方法

- 测地线流核计算为

$$\boldsymbol{\Phi}(t)=\boldsymbol{P}_s\boldsymbol{U}_1\boldsymbol{\Gamma}(t)-\boldsymbol{R}_s\boldsymbol{U}_2\boldsymbol{\Sigma}(t)=\begin{pmatrix}\boldsymbol{P}_s, & \boldsymbol{R}_s\end{pmatrix}\begin{pmatrix}\boldsymbol{U}_1 & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{U}_2\end{pmatrix}\begin{pmatrix}\boldsymbol{\Gamma}(t)\\ \boldsymbol{\Sigma}(t)\end{pmatrix},$$

  - 其中 $\mathbf{R}_s \in \mathbf{R}^{D\times d}$ 是 $\boldsymbol{P}_s$ 的补元素. $\boldsymbol{U}_1 \in \mathbf{R}^{D\times d}$ 和 $\boldsymbol{U}_2 \in \mathbf{R}^{D\times d}$ 是两个正交矩阵:

$$\boldsymbol{P}_s^{\mathrm{T}}\boldsymbol{P}_t=\boldsymbol{U}_1\boldsymbol{V}^{\mathrm{T}},\boldsymbol{R}_s^{\mathrm{T}}\boldsymbol{P}_t=-\boldsymbol{U}_2\boldsymbol{V}^{\mathrm{T}}.$$

- 核 $\boldsymbol{G}$ 可以计算为

$$\boldsymbol{G}=\begin{pmatrix}\boldsymbol{P}_s\boldsymbol{U}_1, & \boldsymbol{R}_s\boldsymbol{U}_2\end{pmatrix}\begin{pmatrix}\boldsymbol{\Lambda}_1 & \boldsymbol{\Lambda}_2\\ \boldsymbol{\Lambda}_3 & \boldsymbol{\Lambda}_4\end{pmatrix}\begin{pmatrix}\boldsymbol{U}_1^{\mathrm{T}}\boldsymbol{P}_s^{\mathrm{T}}\\ \boldsymbol{U}_2^{\mathrm{T}}\boldsymbol{R}_s^{\mathrm{T}}\end{pmatrix},$$

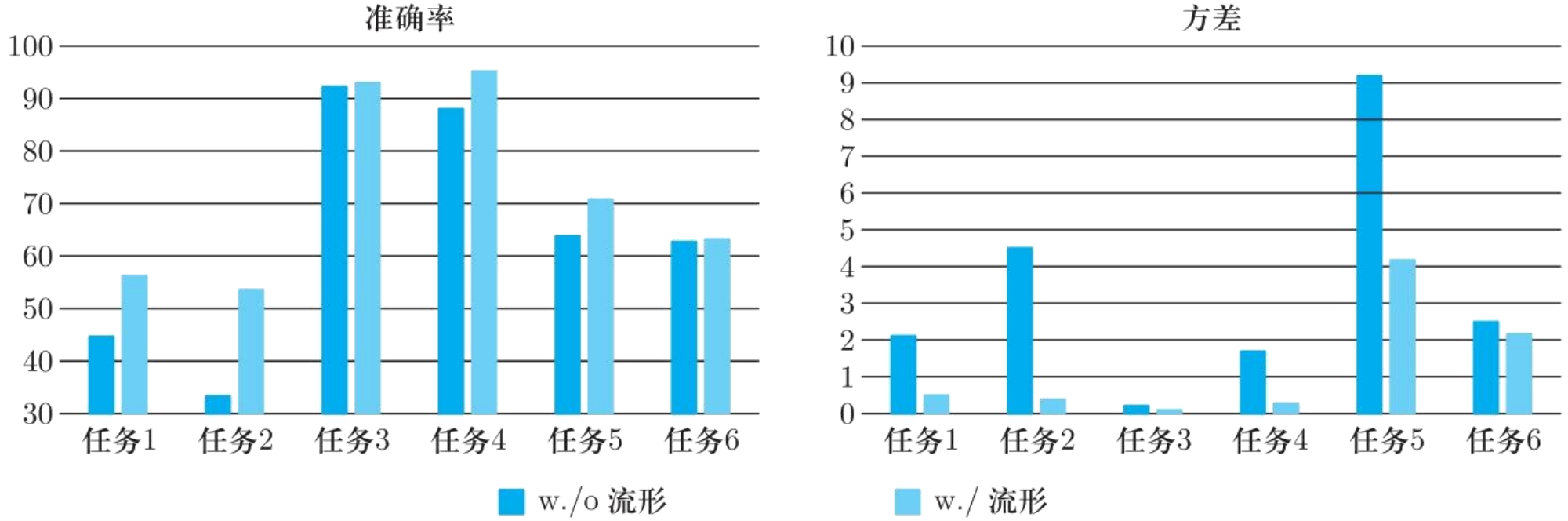
  - 其中, $\boldsymbol{\Lambda}_1,\boldsymbol{\Lambda}_2,\boldsymbol{\Lambda}_3$ 是三个对角矩阵, 它们的角度 θ_i 由奇异值分解 (SVD) 计算得出:

$$\begin{aligned}\lambda_{1i}&=\int_0^1\cos^2(t\theta_i)\,\mathrm{d}t=1+\frac{\sin(2\theta_i)}{2\theta_i},\\ \lambda_{2i}&=-\int_0^1\cos(t\theta_i)\sin(t\theta_i)\,\mathrm{d}t=\frac{\cos(2\theta_i)-1}{2\theta_i},\end{aligned}\tag{16.4.2}$$

16.4.2 流形学习方法

$$\lambda_{3i}=\int_0^1 \sin^2(t\theta_i)\,\mathrm{d}t=1-\frac{\sin(2\theta_i)}{2\theta_i}.$$

- 然后, 原始空间中的特征可以通过 $\boldsymbol{Z}=\sqrt{\boldsymbol{G}}\boldsymbol{X}$ 映射到 Grassmann 流形.核 $\boldsymbol{G}$ 可以使用 SVD 轻松计算. GFK 的实现相当简单, 并且可以作为许多方法的特征处理步骤. 例如, 在流形嵌入分布对齐 (MEDA)(Wang et al., 2018)中, 作者建议在应用分布对齐之前使用 GFK 进行特征提取. 如图 16.6所示,GFK 的集成提高了算法的性能和稳健性.



16.4.2 流形学习方法

- 在后续研究中, Qin et al. (2019) 提出了时变 GFK, 通过添加时间变量来扩展原始 GFK, 以解决跨领域活动识别问题. 他们声称 GFK 的“行走”过程是一个马尔可夫过程：当前变换仅依赖于前一个时间步. 当 GFK 逐渐将源域转换为目标域的子空间时, 后者的变换具有更好的影响. 对于 $t_1 \leqslant t_2$, t_2时刻的变换对最终结果的影响应该比 t1 时刻更大. 时变广义卡尔曼滤波器(temporally adaptive GFK) 将时间变量添加到方程 (16.4.2) 中：

$$\lambda_{1i} = \int_0^1 t\cos^2(t\theta_i)\,\mathrm{d}t = \frac{1}{4} - \frac{1}{4\theta_i^2}\sin^2\theta_i + \frac{1}{4\theta_i}\sin 2\theta_i,$$

$$\lambda_{2i} = -\int_0^1 t\cos(t\theta_i)\sin(t\theta_i)\,\mathrm{d}t = \frac{\cos(2\theta_i)}{4\theta_i} - \frac{\sin 2\theta_i}{8\theta_i^2},$$

$$\lambda_{3i} = \int_0^1 t\sin^2(t\theta_i)\,\mathrm{d}t = \frac{1}{4} + \frac{1}{4\theta_i^2}\sin^2\theta_i - \frac{1}{4\theta_i}\sin 2\theta_i.$$

- 时间自适应 GFK 的性能甚至优于原始 GFK. 此外, 还有许多其他流形迁移学习方法. 例如, Baktashmotlagh et al. (2014) 提出了使用黎曼流形中的赫林格 (Hellinger) 距离来计算从源域到目标域的变换. Guerrero et al. (2014)还提出了一种联合流形适应方法.

16.4.3 最优传输方法

- 基于最优传输的迁移学习方法提供了几何特征变换的另一个视角. 最优传输是一个经典的研究领域, 它具有优美的理论基础, 因此为数学、计算机科学和经济学中许多应用的独特研究提供了独特的研究意义. 最优传输 (OT)最初由法国数学家 Gaspard Monge 于 18 世纪提出. 第二次世界大战期间, 苏联数学家和经济学家 Kantorovich 对其产生了极大关注, 为线性规划奠定了基础. 1975 年, Kantorovich 因其在最优资源分配方面的贡献获得了诺贝尔经济学奖. 我们通常将经典的最优运输问题称为 Monge 问题.

- 最优传输具有可靠的实践环境. 我们以一个例子来说明这一点. 杰克和罗斯一起成长. 他们的家人靠经营仓库为生. 有一天, 罗斯的房子着火了, 她急需一些应急用品. 现在, 杰克必须挺身而出帮助她！我们假设杰克有 n 个不同的仓库. 每个仓库都有一定数量的套件, 表示为 $\{G_i\}_{i=1}^n$ 其中 G_i 是第i 个仓库的套件数量. n 个仓库的位置表示为$\{\boldsymbol{X}_i\}_{i=1}^n$ 同样, 罗斯有 m 个不同的仓库, 其位置表示为 $\{Y_i\}_{i=1}^n$. 每个仓库需要 $\{H_i\}_{i=1}^n$ 个套件. 我们用 $\{c(\boldsymbol{X}_i, \boldsymbol{X}_j)\}_{i,j=1}^{m,n}$ 表示杰克的仓库 i 到罗斯的仓库 j 之间的距离. 已知运输成本会随着距离的增加而增加. 那么, 问题是：杰克如何才能以最低成本帮助罗斯？我们使用一个矩阵 $\boldsymbol{T} \in \mathbf{R}^{n \times m}$ 来表示运输关系, 其中每个元素 T_{ij} 表示从杰克的仓库 i 到罗斯的仓库 j 的套件数量. 这个问题可以表示为

16.4.3 最优传输方法

$$\min \sum_{i,j=1}^{n,m} T_{ij} c\left(\boldsymbol{X}_i, Y_j\right),$$

$$\text{s.t.} \quad \sum_j T_{ij} = G_i, \sum_j T_{ij} = H_i.$$

- 这是一个最优传输的应用. 我们将仓库和套件分别看作概率分布和随机变量. 然后, 最优运输的形成被定义为确定将分布 $P(\boldsymbol{X})$ 转换为 $Q(Y)$ 的最小成本, 其公式为

$$L = \arg\min_{\pi} \iint_{X,Y} \pi(\boldsymbol{X}, Y) c(\boldsymbol{X}, Y) \mathrm{d}\boldsymbol{X} \mathrm{d}Y. \tag{16.4.3}$$

- 给定以下约束条件 ($\pi(\boldsymbol{X}, Y)$ 是它们的联合分布):

$$\int_Y \pi(\boldsymbol{X}, Y) \mathrm{d}Y = P(\boldsymbol{X}),$$

$$\int_X \pi(\boldsymbol{X}, Y) \mathrm{d}\boldsymbol{X} = Q(Y).$$

16.4.3 最优传输方法

- 上述方程表明, 最优传输是关于分布的连接, 这显然与迁移学习有关.
- 为了在迁移学习中使用最优传输, 我们修改方程 (16.4.3) 以得到由最优传输定义的分布散度:

$$D(P,Q)=\inf_{\pi}\iint_{\boldsymbol{X}\times Y}\pi(\boldsymbol{X},Y)c(\boldsymbol{X},Y)\mathrm{d}\boldsymbol{X}\mathrm{d}Y. \tag{16.4.4}$$

- 我们常用L_2 距离来计算, 即

$$c(\boldsymbol{X},Y)=\|\boldsymbol{X}-Y\|_2^2,$$

  - 通过采用 L_2 距离, 方程 (16.4.4) 变成二阶 Wasserstein 距离

$$W_2^2(P,Q)=\inf_{\pi}\int_{\boldsymbol{X}\times Y}\pi(\boldsymbol{X},Y)\|\boldsymbol{X}-Y\|_2^2\,\mathrm{d}\boldsymbol{X}\mathrm{d}Y.$$

- 不同于传统的特征变换方法,最优传输研究点之间的耦合矩阵$\boldsymbol{T}$. 然后,在$\boldsymbol{T}$的映射之后, 源分布可以以最小成本映射到目标域.对于一个数据分布μ,经过重力映射和耦合矩阵$\boldsymbol{T}$之后, 可以得到分布μ.它的新特征向量是

$$\widehat{\boldsymbol{X}}_i=\arg\min_{\boldsymbol{X}\in\boldsymbol{R}^d}\sum_j\boldsymbol{T}(i,j)c(\boldsymbol{X},\boldsymbol{X}_j).$$

16.4.3 最优传输方法

- 那么, 如何确定这个耦合矩阵 $\boldsymbol{T}$ 呢? 这通常与成本有关. 在最优传输中,通常使用变换成本来评估成本, 用 $C(\boldsymbol{T})$ 表示. 带有概率度量 μ 的 $\boldsymbol{T}$ 的成本定义为

$$C(\boldsymbol{T}) = \int_{\Omega_s} c(\boldsymbol{X}, \boldsymbol{T}(\boldsymbol{X}))\mathrm{d}\mu(\boldsymbol{X}).$$

  - 其中 $c(\boldsymbol{X}, \boldsymbol{T}(\boldsymbol{X}))$ 是成本函数, 也可以理解为距离函数. 我们使用以下变换将源分布转换为目标分布:

$$\gamma_0 = \arg\min_{\gamma \in \Pi} \int_{\Omega_s \times \Omega_t} c(\boldsymbol{X}^s, \boldsymbol{X}^t)\mathrm{d}\gamma(\boldsymbol{X}^s, \boldsymbol{X}^t).$$

- 对于分布适应, 我们需要进行边缘适应、条件适应和动态适应. Courtyet al. (2014, 2016) 提出了使用最优传输来学习特征变换 T, 以减少边际分布距离. 然后, 作者提出了联合分布最优传输 (JDOT)(Courty et al., 2017) 以加入条件分布适应. JDOT 的核心公式化为

$$\gamma_0 = \arg\min_{\gamma \in \Pi(\psi_s, \psi_t)} \int_{\Omega \times C}^{2} \mathcal{D}(\boldsymbol{X}_1, Y_1; \boldsymbol{X}_2, Y_2)\mathrm{d}\gamma(\boldsymbol{X}_1, Y_1; \boldsymbol{X}_2, Y_2).$$

16.4.3 最优传输方法

- 其成本函数被表示为边际分布差异和条件分布差异的加权和：

$$\mathcal{D}=\alpha d\left(\boldsymbol{X}_i^s,\boldsymbol{X}_j^t\right)+\mathcal{L}\left(Y_i^s,f\left(\boldsymbol{X}_j^t\right)\right).$$

- 最优运输问题可以使用一些流行的工具来解决, 例如 PythonOT. 最优运输也可以应用于深度学习, 例如 Xu et al. (2020b), Xu et al. (2020a), BhushanDamodaran et al. (2018) 和 Li et al. (2019). 最近, Lu et al. (2021) 提出了将基于最优传输的域自适应应用于跨域人类活动识别, 并取得了很好的性能.他们的方法被称为子结构最优传输 (SOT). 他们认为, 域级和类级的最优传输过于粗略, 可能导致欠适应, 而样本级的匹配可能受到噪声的严重影响, 最终导致过适应. SOT 通过用聚类方法获取的活动的子结构来利用活动的局部信息, 并寻求不同领域之间加权子结构的耦合. 因此, 它被视为更细粒度的最优传输, 并取得了比传统域级最优传输更好的效果.

16.5　并行计算实践

实践代码