第 12 章 迁移学习

目录

12.1 迁移学习问题

《论语·为政》中有这样一句家喻户晓的关于学习的句子：

温故而知新，可以为师矣。

这句话的意思是说，我们在学习新知识之前，如果能先对旧知识加以温习，则可以获得新的理解与体会。而凭借这个能力，就可以成为老师了。更进一步，这句话说明人们的新知识、新能力往往都是由过去所学的旧知识发展变化而来的。在学习新知识时，如果能从旧知识中寻找到与新知识的连接点和相似之处，就可以事半功倍。

《庄子·天运》中有这样一则故事：

故西施病心而颦其里，其里之丑人见而美之，归亦捧心而颦其里。其里之富人见之，坚闭门而不出；贫人见之，挈妻子而去之走。彼知颦美，而不知颦之所以美。

这就是我们非常熟悉的“东施效颦”的故事。西施由于有胸口痛的病，所以她捂着心口皱着眉头走在村子里。同村的东施是一个长得很丑的人，看见西施皱着眉头的样子后，觉得她这样做很漂亮，回家后便也捂着自己的心口在村里行走。村里的富人见了她以后紧闭大门；穷人见了东施则带着妻儿躲开她。东施只知道皱着眉头会很美，却不知道皱眉头为什么会美。

旧的知识可以提炼升华，迁移到新的知识的学习上来，这是一个正向的故事。同样的道理，那为什么东施效颦以失败告终？

其实这中间的关键问题，在于两者之间的相似性。为什么旧知识的温习可以帮助新知识的学习？因为旧知识和新知识之间存在某种关联，正是这种关联给二者建立了桥梁。而东施本来就很丑，与四大美女之一的西施之间根本没有可比性。故她虽然模仿其皱眉，最终也只能贻笑大方。

那么，如何有效地利用事物之间的相关性，来帮助我们解决新问题、学习新能力呢？

这就引出了本书的主题：迁移学习。

12.1.1 迁移学习的概念

迁移学习，顾名思义，就是要进行迁移。放到人工智能和机器学习的学科里讲，迁移学习是一种学习的思想和模式。机器学习是人工智能的一大类重要方法，也是目前发展最迅速、效果最显著的方法，其解决的是让机器自主地从数据中获取知识，从而应用于新的问题中。迁移学习作为机器学习的一个重要分支，侧重于将已经学习过的知识迁移应用于新的问题中。迁移学习的核心问题是，找到新

问题和原问题之间的相似性，才可以顺利地实现知识的迁移。比如，我们如果已经会打乒乓球，就可以类比着学习打网球。再比如，我们如果已经会下中国象棋，就可以类比着下国际象棋。因为这些活动之间，往往有着极高的相似性。生活中常用的“举一反三”、“照猫画虎”就很好地体现了迁移学习的思想。

本书用更加学术更加机器学习的语言来对迁移学习下一个定义。迁移学习，是指利用数据、任务、或模型之间的相似性，将在旧领域学习过的模型，应用于新领域的一种学习过程。

12.1.2 为什么需要迁移学习

1.大数据与少标注之间的矛盾。

现在的人正处在一个大数据时代，每天每时，社交网络、智能交通、视频监控、行业物流等，都产生着海量的图像、文本、语音等各类数据。数据的增多，使得机器学习和深度学习模型可以依赖于如此海量的数据，持续不断地训练和更新相应的模型，使得模型的性能越来越好，越来越适合特定场景的应用。然而，这些大数据带来了严重的问题：总是缺乏完善的数据标注。

众所周知，机器学习模型的训练和更新，均依赖于数据的标注，因为有监督的机器学习需要有标注的数据来作为先验经验。然而，尽管可以获取到海量的数据，这些数据往往是很初级的原始形态，很少有数据被加以正确的人工标注。数据的标注是一个耗时且昂贵的操作，目前为止，尚未有行之有效的方式来解决这一问题。这给机器学习和深度学习的模型训练和更新带来了挑战。反过来说，特定的领域，因为没有足够的标定数据用来学习，使得这些领域一直不能很好的发展。

2.大数据与弱计算之间的矛盾。

大数据，就需要大设备、强计算能力的设备来进行存储和计算。然而，大数据的大计算能力，是”有钱人”才能玩得起的游戏。比如 Google，Facebook，Microsoft，这些巨无霸公司有着雄厚的计算能力去利用这些数据训练模型。例如，使用深度残差网络 ResNet 去训练 ImageNet 数据集就相当耗时间。GoogleTPU 这样计算神经网络的专用芯片也都是有钱人才可以用得起的。

绝大多数普通用户是不可能具有这些强计算能力的。这就引发了大数据和弱计算之间的矛盾。在这种情况下，普通人想要利用这些海量的大数据去训练模型完成自己的任务，基本上不太可能。那么如何让普通人也能利用这些数据和模型？

3.普适化模型与个性化需求之间的矛盾。

机器学习的目标是构建一个尽可能通用的模型，使得这个模型对于不同用户、不同设备、不同环境、不同需求，都可以很好地进行满足。这是一个美好愿景，也就是要尽可能地提高机器学习模型的泛化能力，使之适应不同的数据情形。基于这样的愿望，研究者们构建了多种多样的普适化模型，来服务于现实应用。然而，这只能是研究者们竭尽全力想要做的，目前却始终无法彻底解决的问题。人们的个性化需求五花八门，短期内根本无法用一个通用的模型去满足。比如导航模型，可以定位及导航所有的路线。但是不同的人有不同的需求。比如有的人喜欢走高速，有的人喜欢走偏僻小路，这就是个性化需求。并且，不同的用户，通常都有不同的隐私需求。这也是构建应用需要着重考虑的。

所以目前的情况是，研究者们对于每一个通用的任务都构建了一个通用的模型。这个模型可以解决绝大多数的公共问题。但是具体到每个个体、每个需求，都存在其唯一性和特异性，一个普适化的通用模型根本无法满足。那么，能否将这个通用的模型加以改造和适配，使其更好地服务于人们的个性化需求？

4.特定应用的需求。

机器学习已经被广泛应用于现实生活中。在这些应用中，也存在着一些特定的应用，它们面临着一些现实存在的问题。比如推荐系统的冷启动问题。具体而言，即一个新的推荐系统，没有足够的用户数据，如何进行精准的推荐?一个崭新的图片标注系统，没有足够的标签，如何进行精准的服务？现实世界中的应用驱动着我们去开发更加便捷更加高效的机器学习方法来加以解决。

12.1.3 迁移学习的基本问题

根据杨强教授《迁移学习》专著及综述文献的描述，迁移学习主要研究以下三个基本问题：

1.何时迁移

何时迁移，对应于迁移学习的可能性和使用迁移学习的原因。值得注意的是，此步骤应该发生在迁移学习过程的第一步。给定待学习的目标，首先要做的便是判断当时的任务是否适合进行迁移学习。

2.何处迁移

判断当时的任务适合迁移学习之后，第二步要解决的便是从何处进行迁移。这里的何处，本章用 What 和 Where 来表达可能更好理解。What，指的是要迁移什么知识，这些知识可以是神经网络权值、特征变换矩阵、某些参数等；而 Where 指的是要从哪个地方进行迁移，这些地方可以是某个源域、某个神经元、某个随

机森林里的树等。

3.如何迁移

这一步是绝大多数方法的着力点。给定学习的源域和目标域，这一步则是要学习最优的迁移学习方法以达到最好的性能。

这三个基本问题贯穿整个迁移学习的生命周期。从目前的研究现状来看，何时迁移对应于一些理论、边界条件的证明，大多是一种理论上的保证。它使得迁移学习在具体实现上能够做到有章可循。何处迁移则强调一个动态的迁移过程。在大数据时代，迁移学习需要动态地从数据中学习出更适合迁移的领域、网络、分布等。最后，如何迁移，则旨在建立最优的迁移方法以顺利完成迁移。

另外，这三个基本问题并不是完全对立的，而是在一定条件下可以互相转化。例如，何处迁移往往是随着数据表征动态变化的，而数据表征又与如何迁移有着紧密联系。在特定的数据表征下，这三个问题可以互相辅助，相辅相成。

12.1.4 完整的迁移学习过程

对迁移学习的基本问题有了大致了解后，一个完整的迁移学习过程可以概括为下图所示的步骤。获取所需的数据后，需要对数据进行可迁移性分析，也就对应着基本问题中的何时迁移与何处迁移两个基本问题。接下来便是迁移过程，此部分将在本书的“方法与技术”部分中重点介绍。与机器学习流程类似，一个迁移学习过程结束后，需要按照特定的模型选择方法对迁移学习模型和参数进行选择。可迁移性分析、迁移过程、模型选择这三大基本过程并不是序列式的，而是可以互为反馈、相辅相成的。选择出最好的模型后便是模型的部属和评估。

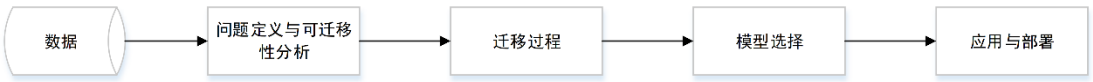


图 1 一个完整的迁移学习过程

12.2 域适应

12.2.1 域适应的定义

为了解释域适应，本书首先需要介绍下迁移学习的基本概念，并对迁移学习进行形式化。

在迁移学习中，有两个基本的概念：领域(Domain)和任务(Task)。它们是最基础的概念。定义如下：

领域(Domain):是进行学习的主体。领域主要由两部分构成：数据和生成这些数据的概率分布。通常用 D 来表示一个 domain，用 P 来表示一个概率分布。特别地，因为涉及到迁移，所以对应于两个基本的领域：源领域(Source Domain)和目标领域(Target Domain)。这两个概念很好理解。源领域就是有知识、有大量数据标注的领域，是要迁移的对象；目标领域就是最终要赋予知识、赋予标注的对象。知识从源领域传递到目标领域，就完成了迁移。领域上的数据，通常用小写粗体 $\mathbf{x}$ 来表示，它也是向量的表示形式。例如，x_i 就表示第 i 个样本或特征。用大写的黑体 $\mathbf{X}$ 表示一个领域的数据，这是一种矩阵形式。通常用大写花体 $\mathcal{X}$ 来表示数据的特征空间，用小写下标 s 和 t 来分别指代两个领域。结合领域的表示方式，则：D_s 表示源领域，D_t 表示目标领域。值得注意的是，概率分布 P 通常只是一个逻辑上的概念，即一般认为不同领域有不同的概率分布，却不给出（也难以给出）P 的具体形式。

任务(Task):是学习的目标。任务主要由两部分组成：标签和标签对应的函数。通常用花体 $\mathcal{Y}$ 来表示一个标签空间，用 $f(\cdot)$ 来表示一个学习函数。相应地，源领域和目标领域的类别空间就可以分别表示为 $\mathcal{Y}_\mathrm{s}$ 和 $\mathcal{Y}_\mathrm{t}$。用小写 y_s 和 y_t 分别表示源领域和目标领域的实际类别。

有了上面领域和任务的定义，迁移学习的形式化就可以进行给出。迁移学习(Transfer Learning):给定一个有标记的源域 $D_s=\{\mathrm{x}_i,y_i\}_{i=1}^{n}$ 和一个无标记的目标域 $D_t=\{\mathrm{x}_j\}_{j=n+1}^{n+m}$。这两个领域的数据分布 $\mathrm{P}(\mathrm{x}_\mathrm{s})$ 和 $\mathrm{P}(\mathrm{x}_t)$ 不同，即 $\mathrm{P}(\mathrm{x}_\mathrm{s})\neq\mathrm{P}(\mathrm{x}_\mathrm{t})$。迁移学习的目的就是要借助 D_s 的知识，来学习目标域 D_t 的知识(标签)。更进一步，结合上文提出的迁移学习研究领域，迁移学习的定义需要进行如下的考虑：

(1)特征空间的异同，即 $\mathcal{X}_\mathrm{s}$ 和 $\mathcal{X}_t$ 是否相等。

(2)类别空间的异同：即 $\mathcal{Y}_\mathrm{s}$ 和 $\mathcal{Y}_\mathrm{t}$ 是否相等。

(3)条件概率分布的异同：即 $Q_\mathrm{s}\left(\mathcal{Y}_\mathrm{s}|\mathcal{X}_s\right)$ 和 $Q_\mathrm{t}\left(\mathcal{Y}_\mathrm{t}|\mathcal{X}_t\right)$ 是否相等。

结合上述形式化，给出领域自适应(Domain Adaptation)这一热门研究方向的定义：领域自适应(Domain Adaptation):即给定一个有标记的源域 $D_s=\{\mathrm{x}_i,y_i\}_{i=1}^{n}$ 和一个无标记的目标域 $D_t=\{\mathrm{x}_j\}_{j=n+1}^{n+m}$，假定它们的特征空间相同，

即 $X_s = X_t$，并且它们的类别空间也相同，即 $\gamma_s = \gamma_t$ 以及条件概率分布也相同，即 $Q_s(\gamma_s | X_s) = Q_t(\gamma_t | X_t)$。但是这两个域的边缘分布不同，即 $P(x_s) \neq P(x_t)$。迁移学习的目标就是，利用有标记的数据 D_s 去学习一个分类器 $f: x_t \to y_t$ 来预测目标域 D_t 的标签 $y_t \in \gamma_t$ 。

例如，要实现一个自动区分垃圾邮件的算法。源数据是个人的邮箱中的数据。现在要用这个算法对朋友邮箱中的邮件进行分类。那么这就是两个任务相同，边缘分布不同的任务了。这样的特殊的迁移学习任务就叫做领域自适应，简称域适应。在实际的研究和应用中，读者可以针对自己的不同任务，结合上述表述，灵活地给出相关的形式化定义。

12.2.2 域适应的分类

根据目标域数据是否有标签，域适应可以被分为以下三种情形：

1. 监督领域自适应(Supervised Domain Adaptation, SDA)，即目标域数据全部有标签的情形($D_t = \{x_j, y_j\}_{j=1}^{N_t}$) ；
2. 半监督领域自适应(Semi-supervised Domain Adaptation, SSDA)，即目标域数据有部分标签的情形($D_t = \{x_j, y_j\}_{j=1}^{N_{tu}} \cup \{x_j, y_j\}_{j=N_{tu}+1}^{N_{tl}}$，其中 N_{tu} 和 N_{tl} 分别为无标签和有标签的目标域数据个数)；
3. 无监督领域自适应(Unsupervised Domain Adaptation, UDA)，即目标域数据完全没有标签的情形($D_t = \{x_j, ?\}_{j=1}^{N_t}$)

12.2.3 域适应的技术

常见的域适应技术有以下几种：

1. 基于特征的域适应(Feature Adaption)是将源域样本和目标域样本用一个映射 Φ 调整到同一个特征空间，这样在这个特征空间样本能够“对齐”，这也是最常用的方法：

$$\min \frac{1}{n} \sum_{i=1}^{n} L(\Phi(x_i^s), y_i^s, \theta) \tag{1}$$

2. 基于实例的域适应(Instance Adaption)是考虑到源域中总有一些样本和目标域样本很相似，那么就将源域的所有样本的损失值 Loss 在训练时都乘以一

个权重 w_i （即表示“看重”的程度），和目标域越相似的样本，这个权重就越大：

$$\min \frac{1}{n} \sum_{i=1}^{n} w_i L(\Phi(x_i^s), y_i^s, \theta) \tag{2}$$

3. 基于模型参数的域适应(Model Adaption)是找到新的参数 θ'，通过参数的迁移使得模型能更好的在目标域上工作：

$$\min \frac{1}{n} \sum_{i=1}^{n} L(\Phi(x_i^s), y_i^s, \theta') \tag{3}$$

12.3 最大均值差异法

12.3.1 最大均值差异法的基本概念

迁移学习的核心就是找到源领域和目标领域之间的相似性，并加以合理利用。而当我们找到这种相似性后，下一步需要做的工作就是度量和利用相似性。那么如何度量这种相似性呢？

一般而言，可以使用某种度量原则来计算出源领域和目标领域之间的相似性。在机器学习中，计算两个向量（点、矩阵）的距离和相似度是许多机器学习算法中都会使用到的步骤。常见的相似度和距离度量手段有很多，而在迁移学习中，最大均值差异是使用频率最高的度量。

最大均值差异度量(MMD Maximum Mean Discrepancy)，它度量在再生希尔伯特空间中两个分布的距离，是一种核学习方法。记任意两个随机变量为 A 和 B，对应的第 i，j 次采样取值分别为 a_i，b_j，则这两个随机变量的 MMD 平方距离为

$$\mathrm{MMD}^2(A,B) = \left\| \sum_{i=1}^{n1} \phi(a_i) - \sum_{j=1}^{n2} \phi(b_j) \right\|_H^2, \tag{4}$$

其中，$\phi(\cdot)$ 是映射，用于把原变量映射到再生核希尔伯特空间 (Reproducing Kernel Hilbert Space, RKHS) 中。什么是 RKHS？形式化定义太复杂，简单来说希尔伯特空间是对于函数的内积完备的，而再生核希尔伯特空间是具有再生性 $\langle K(a,\cdot), K(b,\cdot) \rangle_H = K(a,b)$ 的希尔伯特空间。将平方展开后，RKHS 空间中的内积就可以转换成核函数，所以最终 MMD 可以直接通过核函数进行计算。简单理解，MMD 就是求两堆数据在 RKHS 中的均值的距离。

而如果把现在的核 k 视为一组不同的核函数的组合，然后用一定的优化方法求得这个组合后的最优的结果，岂不是可以解决问题？这就催生了多核的 MMD，简称 MK-MMD。现有的 MMD 方法是基于单一核变换的，多核的 MMD 假设最优的核可以由多个核线性组合得到。多核 MMD 的提出和计算方法在文献 [Gretton et al., 2012] 中形式化给出。MK-MMD 在许多后来的方法中被大量使用，最著名的方法是 DAN [Long et al., 2015a]，即深度适配网络。

12.3.2 基于最大均值差异的迁移方法

本节介绍基于最大均值差异 MMD 的迁移学习方法。回顾迁移学习的统一表征公式，特征变换法的迁移学习优化目标如下：

$$f^* = \arg\min_{f \in \mathrm{H}} \frac{1}{N_s} \sum_{i=1}^{N_s} \ell(f(x_i), y_i) + \lambda R(T(D_s), T(D_t)) , \quad (5)$$

其中 T 为需要求的特征变换。

那么，MMD 距离与特征变换函数 T 有什么关系呢？回顾分布差异度量的一般表达形式：

$$D(D_s, D_t) \approx (1-\mu) D(P_s(x), P_t(x)) + \mu D(P_s(y|x), P_t(y|x)) , \quad (6)$$

（6）式中可以明显直接用 MMD 距离去计算边缘分布差异 $D(P_s(x), P_t(x))$，事实上这就是经典迁移学习方法迁移成分分析(Transfer Component Analysis TCA)的核心思想。问题在于:目标域样本没有标签,即无法求得$P_t(y|x)$，因此条件分布差异 $D(P_s(y|x), P_t(y|x))$在这里无解。

这条路看来是走不通了,也就是说,直接建模$P_t(y|x)$不行。那么,能不能有别的办法可以逼近这个条件概率？可以换个角度,利用类条件概率 $P_t(x|y)$。根据贝叶斯公式 $P_t(y|x) = P_t(y) P_t(x|y)$ ，如果忽略 $P_t(y)$，那么就可以用 $P_t(x|y)$ 来近似 $P_t(y|x)$吗？

实际怎么做呢？在依然没有 y_t 的情况下，一般采用的方法是，用 (x_s, y_s) 来训练一个简单的分类器(比如 KNN 、逻辑斯特回归)到 x_t 上直接进行预测，这样总能够得到一些伪标签 $\hat{y}_t$ 。然后根据伪标签来计算，这个问题就可解了。

边缘分布的 MMD 距离可以被表示为

$$MMD(P_s(x),P_t(x))=\left\|\frac{1}{N_s}\sum_{i=1}^{N_s}A^Tx_i-\frac{1}{N_t}\sum_{j=1}^{N_t}A^Tx_j\right\|_{\mathrm{H}}^2 \quad , \tag{7}$$

边缘分布自适应方法由杨强教授及其团队在 2009 年提出，称为迁移成分分析(Transfer Component Analysis (TCA))。该方法是领域的经典方法。类似地，条件分布的 MMD 距离可以被近似表示为

$$MMD(P_s(y|x),P_t(y|x))=\sum_{c=1}^{C}\left\|\frac{1}{N_s^{(C)}}\sum_{x_i\in D_s^{(c)}}A^Tx_i-\frac{1}{N_t^{(C)}}\sum_{x_i\in D_t^{(c)}}A^Tx_j\right\|_{\mathrm{H}}^2, \tag{8}$$

其中，$N_s^{(C)}$，$N_t^{(C)}$分别表示源域和目标域中来自第 c 类的样本个数，C 为类别个数。$D_s^{(C)}$，$D_{\mathrm{t}}^{(C)}$分别表示源域和目标域中来自第 c 类的样本。

这里先给出上述式子的最终表达形式，然后再以公式（7）为例，详解如何进行数学变换。用 MMD 进行迁移学习的最终表达形式为

$$tr(A^TXMX^TA) \quad , \tag{9}$$

其中，tr(·)表示矩阵的迹，A 矩阵是要求的特征变换函数 T 的对应矩阵，X 是由源域和目标源域样本拼接成的矩阵。怎么样，这是不是就通过 MMD 巧妙地达到了特征变换和分布距离的统一？

上式中的 M 是 MMD 矩阵，其可以被计算为

$$M=(1-\mu)M_0+\mu\sum_{c=1}^{C}M_c \quad , \tag{10}$$

其中的边缘和条件 MMD 矩阵可以按如下方式计算：

$$(M_0)_{ij}=\begin{cases}\frac{1}{N_s^2},x_i,x_j\in D_s\\ \frac{1}{N_t^2},x_i,x_j\in D_t\\ -\frac{1}{N_sN_t},otherwise\end{cases} \quad , \tag{11}$$

$$(M_c)_{ij}=\begin{cases}\frac{1}{(N_s^{(c)})^2},x_i,x_j\in D_s^{(c)}\\ \frac{1}{(N_t^{(c)})^2},x_i,x_j\in D_t^{(c)}\\ -\frac{1}{N_s^{(c)}N_t^{(c)}},\begin{cases}x_i\in D_s^{(c)},x_j\in D_t^{(c)}\\ x_i\in D_t^{(c)},x_j\in D_s^{(c)}\end{cases}\\ 0,otherwise\end{cases} \quad , \tag{12}$$

特别地，当设定自适应因子 μ = 0.5 时，上式对应于联合分布自适应(Joint Distribution Adaptation, JDA)方法，而更为一般的形式则是动态分布自适应方法。

下面以边缘分布的 MMD 距离为例，详细介绍如何进行这样的概率化简，最终整理成核的形式。

$$
\begin{aligned}
&\left\| \frac{1}{N_s}\sum_{i=1}^{N_s} A^T x_i - \frac{1}{N_t}\sum_{j=1}^{N_t} A^T x_j \right\|^2 \\
&= \left\| \frac{1}{N_s} A^T [x_1 x_2 ... x_{N_s}]_{1\times N_s} \begin{bmatrix} 1 \\ 1 \\ . \\ . \\ . \\ 1 \end{bmatrix}_{N_s\times 1} - \frac{1}{N_t} A^T [x_1 x_2 ... x_{N_t}]_{1\times N_s} \begin{bmatrix} 1 \\ 1 \\ . \\ . \\ . \\ 1 \end{bmatrix}_{N_t\times 1} \right\|^2 \\
&= tr(\frac{1}{N_s^2} A^T X_s 1 (A^T X_s 1)^T + \frac{1}{N_t^2} A^T X_t 1 (A^T X_s 1)^T - \frac{1}{N_s N_t} A^T X_s 1 (A^T X_t 1)^T - \frac{1}{N_s N_t} A^T X_t 1 (A^T X_s 1)^T) \\
&= tr(\frac{1}{N_s^2} A^T X_s 1 1^T X_s^T A^T + \frac{1}{N_s^2} A^T X_t 1 1^T X_t^T A^T - \frac{1}{N_s N_t} A^T X_s 1 1^T X_t^T - \frac{1}{N_s N_t} A^T X_t 1 1^T X_s^T) \\
&= tr[A^T (\frac{1}{N_s^2} A^T X_s 1 1^T X_s^T A^T + \frac{1}{N_s^2} A^T X_t 1 1^T X_t^T A^T - \frac{1}{N_s N_t} A^T X_s 1 1^T X_t^T - \frac{1}{N_s N_t} A^T X_t 1 1^T X_s^T) A] \\
&= tr(A^T [X_s X_t] \begin{bmatrix} \frac{1}{N_s^2} 1 1^T & \frac{-1}{N_s N_t} 1 1^T \\ \frac{-1}{N_s N_t} 1 1^T & \frac{1}{N_t^2} 1 1^T \end{bmatrix} \begin{bmatrix} X_s \\ X_t \end{bmatrix} A) \\
&= tr(A^T X M X^T A)
\end{aligned}
\tag{13}
$$

上述推导用到了矩阵两个非常重要的性质：

1. $\|A\|^2 = tr(AA^T)$ ，在第二步中使用。

2. $tr(AB) = tr(BA)$ ，在第四步中使用。

条件分布的 MMD 距离变化同理。

12.3.3 求解与计算

好了，现在推导出了问题的最终表达形式公式，只需要将其最小化就可以了。事实真的如此吗？任何最优目标都是有约束的，否则，最小化公式极其简单：只

要令特征变换矩阵 A 中的所有元素都为 0 便可以了。可是这样有什么意义呢？

事实上，还需要考虑的约束条件是特征变换前后样本的散度问题，也可以理解成是数据的方差。那么怎么求数据的方差呢？

可以用散度（Scatter）对方差进行近似。给定样本集 x，其散度矩阵 S 可以表示为

$$\begin{aligned} S &= \sum_{j=1}^{n}(x_j-\bar{x})(x_j-\bar{x})^T \\ &= \sum_{j=1}^{n}(x_j-\bar{x})\otimes(x_j-\bar{x}) \\ &= (\sum_{j=1}^{n}(x_j x_j^T))-n(\overline{XX})^T , \end{aligned} \tag{14}$$

其中 $\bar{x}=\frac{1}{n}\sum_{j=1}^{n}x_j$ 表示样本均值。用 $H=I-(1/n)1$ 表示中心矩阵，$I\in\mathbb{R}^{(n+m)\times(n+m)}$ 表示单位矩阵，则样本的散度矩阵（即方差）可以被表示为

$$S=XHX^T , \tag{15}$$

将 A 代入，则方差最大化可以被形式化表示为

$$\max(A^TX)H(A^TX)^T , \tag{16}$$

将公式（9）的数据均值之差最小化和公式（16）中的散度最大化结合起来，最终优化目标表示为

$$\min\frac{tr(A^TXMX^TA)}{tr(A^TXHX^TA)} , \tag{17}$$

对公式（17）而言，我们要求其分子最小化，分母最大化。这给求解带来了困难。注意到迁移变换矩阵 A 是一个 Hermitan 矩阵，即满足 $A^H=A$,H 表示矩阵的共轭转置。在此条件下，公式（17）可以根据瑞利商（Rayleigh Quotient）进行变换求解。将公式（17）变换为

$$\begin{aligned} &\min tr(A^TXMX^TA)+\lambda\|A\|_F^2 , \\ &s.t.A^TXHX^TA=I. \end{aligned} \quad , \tag{18}$$

其中，正则项 $\lambda\|A\|_F^2$ 用来保证此问题是良好定义的，λ 为正则项系数。公式（18）即为基于 MMD 进行迁移学习的最终学习目标。

上式如何求解？通常我们用拉格朗日法进行。上式的拉格朗日函数表示为

$$L=tr((A^TXAX^T+\lambda I)A)+tr((I-A^TXHX^TA)\Phi) , \tag{19}$$

令上式的导数$\partial L / \partial A = 0$ ，得

$$(XMX^T + \lambda I)A = XHX^T A\Phi \quad , \qquad (20)$$

其中的Φ是拉格朗日乘子。别看这个式子复杂，既有要求解的 A、又有一个新加入的Φ，但是它在 Matlab 和 Python 等编程语言中是可以直接求解的（例如，Matlab 中用 eigs 函数即可）。这样我们就得到了变换 A，问题解决了。

可是伪标签终究是伪标签啊，肯定精度不高，怎么办？有个非常流行的机制叫做迭代：做一次不行，就再做一次。做后一次迭代时，我们用上一轮得到的标签来做这一次的伪标签。这样的目的是得到越来越好的伪标签，而参与迁移的数据是不会变的，如此往返多次，结果就自然而然好了。

12.3.4 应用与扩展

基于 MMD 进行迁移学习方法的步骤如下:输入两个特征矩阵,首先用一个初始的简单分类器(如 KNN)计算目标域的伪标签。随后计算 M 和 H 矩阵,然后选择一些常用的核函数进行映射(比如线性核、高斯核)计算 K,接着求解公式（17）中的 A，取其前 m 个特征值。之后，得到的就是源域和目标域的降维后的数据。由于在计算条件分布差异时，伪标签并不准确，因此使用多次迭代使结果更好。

基于 MMD 的迁移方法得到了广泛的扩展和应用。对于更一般的动态分布适配方法而言,研究者提出基于流形学习的动态迁移方法 MEDA。最近也有研究者提出一个动态对坑适配网络 DAAN（Dynamic Adversarial Adaptation Network）来解决对抗网络中的动态适配问题。

12.4 深度学习中的迁移学习

随着深度学习方法的大行其道，越来越多的研究人员使用深度神经网络进行迁移学习。对比传统的非深度迁移学习方法，深度迁移学习直接提升了在不同任务上的学习效果。并且，由于深度学习直接对原始数据进行学习，所以其对比非深度方法还有两个优势：自动化地提取更具表现力的特征，以及满足了实际应用中的端到端 (End-to-End) 需求。

近年来，以生成对抗网络 (Generative Adversarial Nets, GAN) [Goodfellow et al., 2014] 为代表的对抗学习也吸引了很多研究者的目光。基于 GAN 的各种变体网络不断涌现。对抗学习网络对比传统的深度神经网络，极大地提升了学习效果。因此，基于对抗网络的迁移学习，也是一个热门的研究点。

本部分重点介绍迁移学习在深度学习中的基本思路，具体而言，如何在深度神经网络中采用迁移学习？首先本书回答一个最基本的问题：为什么深度网络是可迁移的？然后，介绍最简单的深度网络迁移形式：finetune。接着分别介绍使用深度网络和深度对抗网络进行迁移学习的基本思路和核心方法。值得注意的是，由于深度迁移学习方面的研究工作层出不穷，本书不可能覆盖到所有最新的方法。但是基本上，这些方法的原理都大同小异。因此，本书的介绍是具有普适性的。

12.4.1 深度网络的可迁移性

随着 AlexNet [Krizhevsky et al., 2012] 在 2012 年的 ImageNet 大赛上获得冠军，深度学习开始在机器学习的研究和应用领域大放异彩。尽管取得了很好的结果，但是神经网络 本身就像一个黑箱子，看得见，摸不着，解释性不好。由于神经网络具有良好的层次结构，很自然地就有人开始关注，能否通过这些层次结构来很好地解释网络？于是，有了大家熟知的例子：假设一个网络要识别一只猫，那么一开始它只能检测到一些边边角角的东西，和猫根本没有关系；然后可能会检测到一些线条和圆形；慢慢地，可以检测到有猫的区域；接着是猫腿、猫脸等等。

这表达了一个什么事实呢？概括来说就是：前面几层都学习到的是通用的特征（general feature）；随着网络层次的加深，后面的网络更偏重于学习任务特定的特征（specific feature）。 这非常好理解，大家也都很好接受。那么问题来了：如何得知哪些层能够学习到 general feature，哪些层能够学习到 specific feature。更进一步：如果应用于迁移学习，如何决定该迁移哪些层、固定哪些层？

这个问题对于理解神经网络以及深度迁移学习都有着非常重要的意义。

来自康奈尔大学的 Jason Yosinski 等人 [Yosinski et al., 2014] 率先进行了深度神经网络可迁移性的研究，将成果发表在 2014 年机器学习领域顶级会议 NIPS 上并做了口头汇报。该论文是一篇实验性质的作者（通篇没有一个公式）。其目的就是要探究上文提到的几个关键性问题。因此，作者的全部贡献都来自于实验及其结果。

在 ImageNet 的 1000 类上，作者把 1000 类分成两份（A 和 B），每份 500 个类别。然后，分别对 A 和 B 基于 Caffe （一个深度学习框架）训练了一个 AlexNet 网络。一个 AlexNet 网络一共有 8 层， 除去第 8 层是类别相关的网络无法迁移以外，作者在 1 到 7 这 7 层上逐层进行 finetune 实验，探索网络的可迁移性。

实验结果表明：（1）神经网络的前 3 层基本都是 general feature，进行

迁移的效果会比较好;(2)深度迁移网络中加入 fine-tune,效果会提升比较大,可能会比原网络效果还好;(3)Finetune 可以比较好地克服数据之间的差异性;

12.4.2 最简单的深度迁移：finetune

这里对 finetune 进行正式介绍。 Finetune，也叫微调、finetuning，是深度学习中的一个重要概念。简而言之，finetune 就是利用别人已经训练好的网络，针对自己的任务再进行调整。从这个意思上看，不难理解 finetune 是迁移学习的一部分。

为什么需要已经训练好的网络？因为在实际的应用中，算法科学家们通常不会针对一个新任务，就去从头开始训练一个神经网络。这样的操作显然是非常耗时的。尤其是，训练数据一般不可能像 ImageNet 那么大，可以训练出泛化能力足够强的深度神经网络。即使有如此之多的训练数据，如果从头开始训练，其代价也是不可承受的。那么怎么办呢？从迁移学习可以得知，利用之前已经训练好的模型，将它很好地迁移到自己的任务上即可。

为什么需要 finetune？ 因为别人训练好的模型，可能并不是完全适用于自己的任务。可能别人的训练数据和自己的数据之间不服从同一个分布；可能别人的网络能做比自己的任务更多的事情；可能别人的网络比较复杂，而自身的任务比较简单。举一个例子来说，假如想训练一个猫狗图像二分类的神经网络，那么很有参考价值的就是在 CIFAR-100 数据集上训练好的神经网络。但是 CIFAR-100 有 100 个类别，我们只需要 2 个类别。此时，就需要针对我们自己的任务，固定原始网络的相关层，修改网络的输出层，以使结果更符合自身的需要。

Finetune 的优势是显然的，包括：

- 不需要针对新任务从头开始训练网络，节省了时间成本；
- 预训练好的模型通常都是在大数据集上进行的，无形中扩充了自身的训练数据，使得模型更鲁棒、泛化能力更好；
- Finetune 实现简单，使得每个人只关注自己的任务即可。

在实际应用中，通常几乎没有人会针对自己的新任务从头开始训练一个神经网络。Finetune 是一个理想的选择。Finetune 并不只是针对深度神经网络有促进作用，对传统的非深度学习也有很好的效果。例如，finetune 对传统的人工提取特征方法就进行了很好的替代。我们可以使用深度网络对原始数据进行训练，依赖网络提取出更丰富更有表现力的特征。然后，将这些特征作为 传统机器学习方法的输入。这样的好处是显然的：既避免了繁复的手工特征提取，又能自动地提取出更有表现力的特征。

12.4.3 深度网络自适应

深度网络的 finetune 可以帮助模型节省训练时间，提高学习精度。但是 finetune 有它的先天不足：它无法处理训练数据和测试数据分布不同的情况。因为 finetune 的基本假设也是训练数据和测试数据服从相同的数据分布。这在迁移学习中也是不成立的。因此，更重要的是针对深度网络开发出更好的方法使之更好地完成迁移学习任务。

这里本书首先介绍第一种深度迁移学习方法，数据分布自适应方法。针对数据分布自适应，许多深度学习方法都开发出了自适应层 (Adaptation Layer) 来完成源域和目标域数据的自适应。自适应能够使得源域和目标域的数据分布更加接近，从而使得网络的效果更好。

因此，深度网络的自适应主要完成两部分的工作：一是哪些层可以自适应，这决定了网络的学习程度； 二是采用什么样的自适应方法 (度量准则)，这决定了网络的泛化能力。

深度网络中最重要的是网络损失的定义。绝大多数深度迁移学习方法都采用了以下的损失定义方式：

$$\ell = \ell_c(D_s, y_s) + \lambda \ell_A(D_s, D_t) \tag{21}$$

其中，ℓ 表示网络的最终损失，$\ell_c(D_s, y_s)$ 表示网络在有标注的数据 (大部分是源域) 上的常规分类损失 (这与普通的深度网络完全一致)，$\ell_A(D_s, D_t)$ 表示网络的自适应损失。最后一部分是传统的深度网络所不具有的、迁移学习所独有的。此部分的表达与本书先前讨论过的源域和目标域的分布差异，在道理上是相同的。式中的 λ 是权衡两部分的权重参数。

上述的分析指导我们设计深度迁移网络的基本准则：决定自适应层，然后在这些层加入自适应度量，最后对网络进行 finetune。

12.4.4 深度对抗网络迁移

我们继续介绍第二种深度迁移学习方法：生成对抗网络 GAN。受到自博弈论中的二人零和博弈 (two-player game) 思想的启发而提出。它一共包括两个部分：一部分为生成网络 (Generative Network)，此部分负责生成尽可能地以假乱真的样本，这部分被成为生成器 (Generator)；另一部分为判别网络 (Discriminative Network)， 此部分负责判断样本是真实的，还是由生成器生

成的，这部分被成为判别器 (Discriminator)。 生成器和判别器的互相博弈，就完成了对抗训练。

GAN 的目标很明确：生成训练样本。这似乎与迁移学习的大目标有些许出入。然而，由于在迁移学习中，天然地存在一个源领域，一个目标领域，因此，我们可以免去生成样本的过程，而直接将其中一个领域的数据 (通常是目标域) 当作是生成的样本。此时，生成器的职能发生变化，不再生成新样本，而是扮演了特征提取的功能：不断学习领域数据的特征，使得判别器无法对两个领域进行分辨。这样，原来的生成器也可以称为特征提取器 (Feature Extractor)。

通常用G_f来表示特征提取器，用G_d来表示判别器。正是基于这样的领域对抗的思想，深度对抗网络可以被很好地运用于迁移学习问题中。与深度网络自适应迁移方法类似，深度对抗网络的损失也由两部分构成：网络训练的损失ℓ_c和领域判别损失ℓ_d：

$$\ell=\ell_c(D_s,y_s)+\lambda\ell_d(D_s,D_t) \tag{22}$$

12.5 迁移学习实例

为了让读者加深对迁移学习的了解，提供了三个迁移学习的实例以供参考。

12.5.1 基于深度迁移学习模型的花卉种类识别

花卉种类识别，一直是植物识别领域的热门话题之一。在国内，诸多学者在花卉种类识别的研究中，取得了一些进展。这里，作者使用深度迁移学习方法对花卉种类进行识别。具体而言，基于 Inception_v3 模型来做迁移学习。核心点在于利用预训练的源模型，此方法的优点在于不需要模型源数据与具体图像数据一致，只是将 Inception_v3 模型的自动提取特征的能力移植过来，保留低层的结构参数，并对相关参数进行微调，从而构建花卉识别的神经网络。

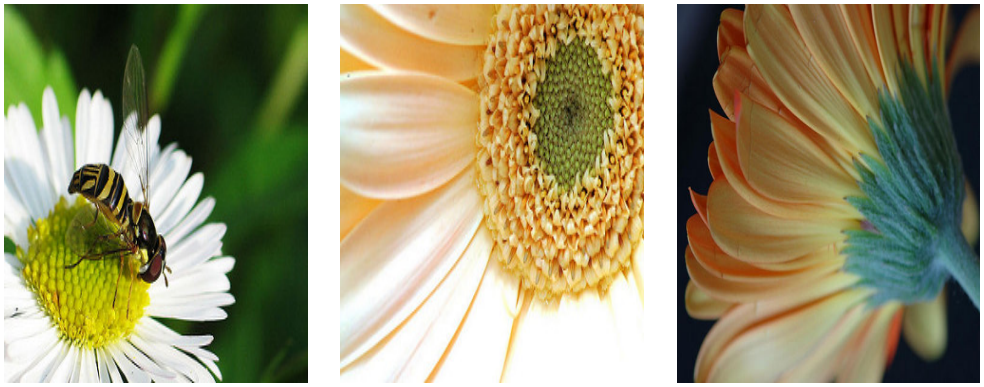
（1）数据集

该研究所用到的图像来自于中国植物主题数据库，主要选取了雏菊、蒲公英、玫瑰花、向日葵和郁金香 5 种常见花卉作为识别对象。5 类花卉试验样本图像共采集 3670 张，其中雏菊 633 张，蒲公英 898 张，玫瑰花 641 张，向日葵 699 张，郁金香 799 张。为避免训练过拟合，对原始数据集进行扩增。通过 Python 语言，将图片进行水平翻转、旋转 90°，对样本量进行扩增。此时，共得到 11010 张，其中雏菊 1899 张，蒲公英 2694 张，玫瑰花 1923 张，向日葵 2097 张，郁金

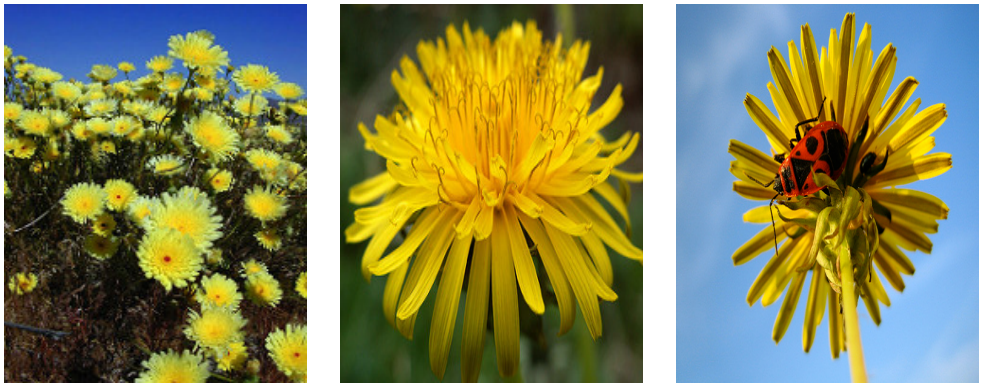
香 2397 张。每类花卉随机选取 200 张图片作为验证集，剩余 10010 张作为训练集。表 1 列出了具体的数据集数量。每类花卉示例见图 2。

表 1（数据来源：刘嘉政，江苏农业科学，2019，47(20) ：231－236）

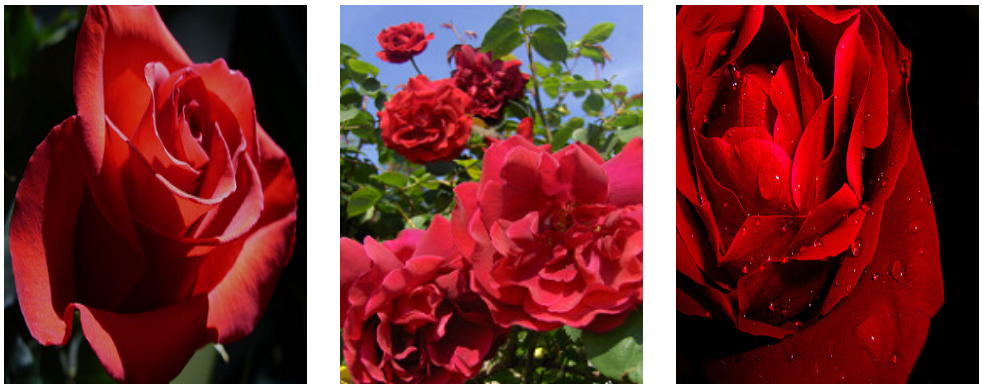
花卉名称	训练集（张）	验证集（张）
雏菊	1699	200
蒲公英	2494	200
玫瑰花	1723	200
向日葵	1897	200
郁金香	2197	200



雏菊

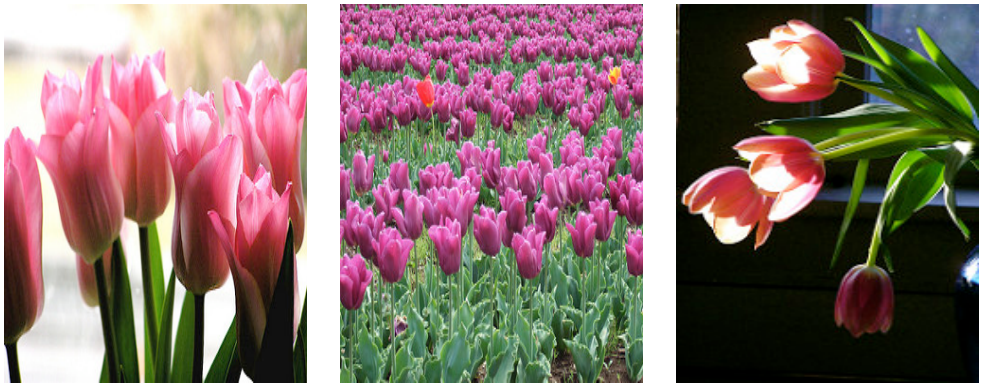


蒲公英



玫瑰

向日葵



郁金香

图 2 花卉数据集示例（图片来源：中国植物主题数据库）

（2）模型构建

图 3 为花卉识别深度迁移学习模型的构建流程图。利用 GoogleNet 上预先训练好的 Inception_v3 模型，将具有自动提取图像特征能力的卷积、池化层迁移到本研究的模型中。当输入 1 张花卉图像时，对该图像特征进行提取，并用 2048 维张量表示，并依次存入缓存文件中。作者还对 Inception_v3 的全连接层、Dropout 层进行了重新训练，对结构参数进行微调，以适应实验要求。

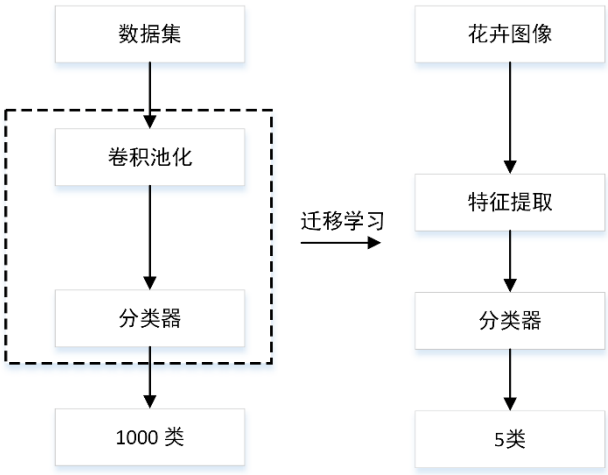


图 3 花卉识别深度迁移学习模型的构建流程

深度迁移学习过程分为 3 个阶段:

①初始化设置: 对随机变换的图像与原始图像相结合,构成用于本研究训练的花卉图像数据库。基于预训练的 Inception_v3 模型自带的结构参数，将花卉数据库作为下一阶段模型训练的数据源;

②除了预训练模型和最后 1 层全连接层以外，要重新训练完整的全连接层，并将最后的全连接层的结构参数作为下一阶段的特征提取输入源;

③最后 1 个阶段是对整个模型结构层数进行微调，以适应该研究要求。最后采用 Softmax 分类器，作为最后的分类输出层。

（3）结果分析

经过数据初始化、参数调整、训练模型，得到 accuracy 和 loss 迭代收敛结果(图 4、图 5) 。当迭代次数达到 4500 次 时，准确率和损失值都接近收敛，数值不再有大的变化。

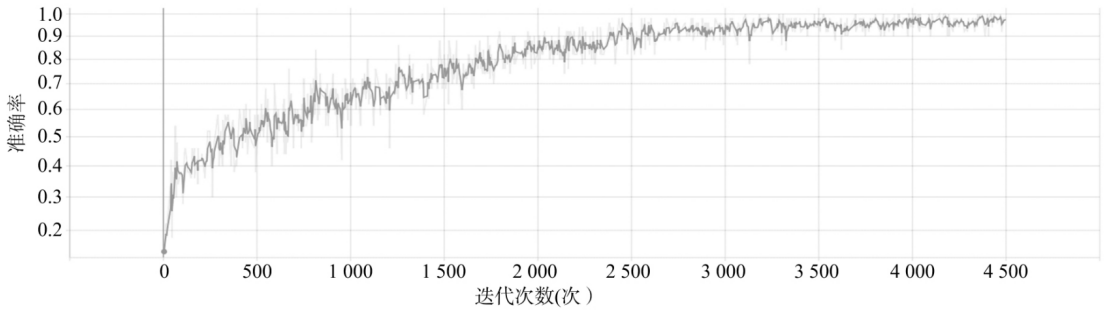


图 4 accuracy 迭代收敛结果（图片来源：刘嘉政，江苏农业科学，2019，47(20) : 231－236）

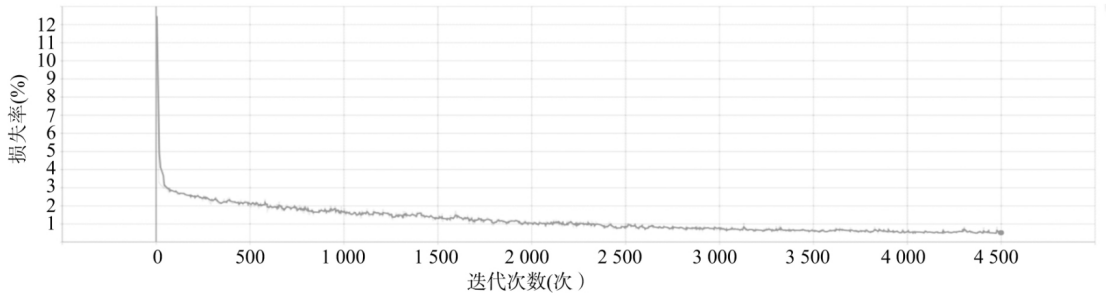


图 5 loss 迭代收敛结果（图片来源：刘嘉政，江苏农业科学，2019，47(20) : 231－236）

（4）方法对比

为验证迁移学习方法的可行性，与传统的支持向量机（ SVM) 、卷积神经网络(CNN) 进行比较。对于 SVM 模型，通过提取全局特征和局部特征，对花卉图像进行识别，平均识别率最低; 对于 CNN 来说，平均识别率较高，达到 80% 以上; 而深度迁移学习方法，对 5 种花卉图像的平均识别率达到 93.73%，比较结果见表 2。

表 2

方法	分类识别率(%)				
	雏菊	蒲公英	玫瑰花	向日葵	郁金香
SVM	50.61	47.58	45.69	54.29	57.27
CNN	82.52	81.25	84.19	80.11	81.24
深度迁移学习	94.25	95.14	92.21	93.17	93.89

12.5.2 基于迁移学习的对抗生成网络实现医学图像分割

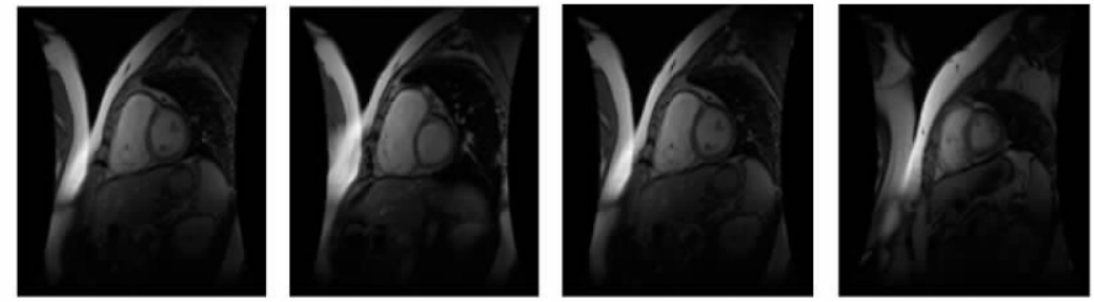
随着人工智能技术的飞速发展，医学图像处理在临床诊断中的作用日益凸显，其中医学图像分割是医学图像处理的重要环节。其目的是将医学图像中有特殊意义的部分在整张图像中分割出来，并将其相关特征进行提取，为医生的诊断以及病理学研究等提供可靠的辅助依据。

为了实现医学图像分割，作者用基于迁移学习的分割网络 （ transfer learning based segmentation network，TLBSN) 构建对抗生成网络（GAN）提取特征，解决样本数据不足、网络难以获得足够的特征信息的问题，并通过构建多尺度的判别器以获得局部和全局特征信息，提升分割的准确性。

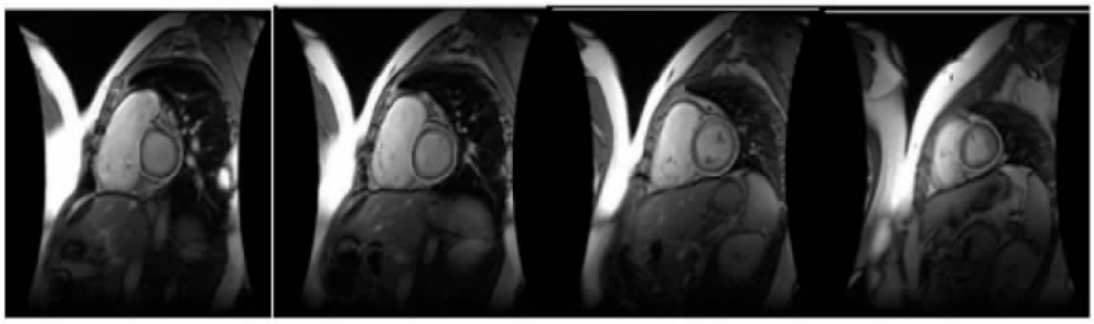
（1） 数据集

作者的实验数据来源于多伦多儿童病医院影像科，该数据集由 33 名受试

者的短轴心脏的核磁共振图像序列组成， 每位受试者都有 8 到 15 个序列，每个序列是一个 20 帧的心动周期，其中含有手动分割标签 5 011 张图像， 使用其中 4 009 张图像作为实验的训练集，剩下的 1 002 张图像作为测试集。原始图像大小为 256×256，通过感兴趣区域(Region Of Interest)提取 128 ×128 的大小以在保存左心室轮廓的情况下减少无用信息，如图 6 所示。



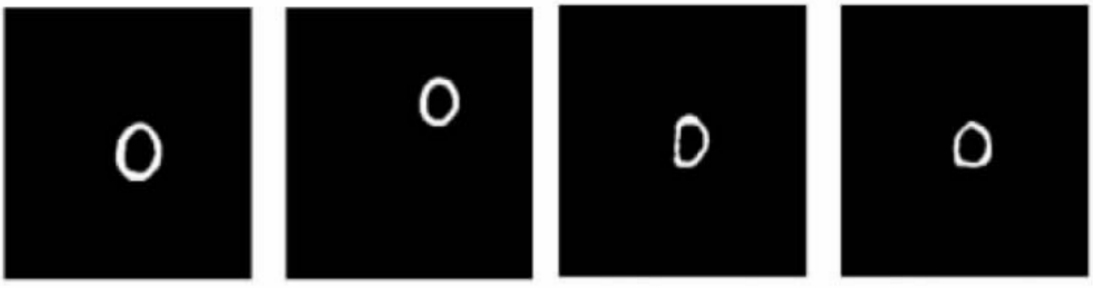
（a）原始图像



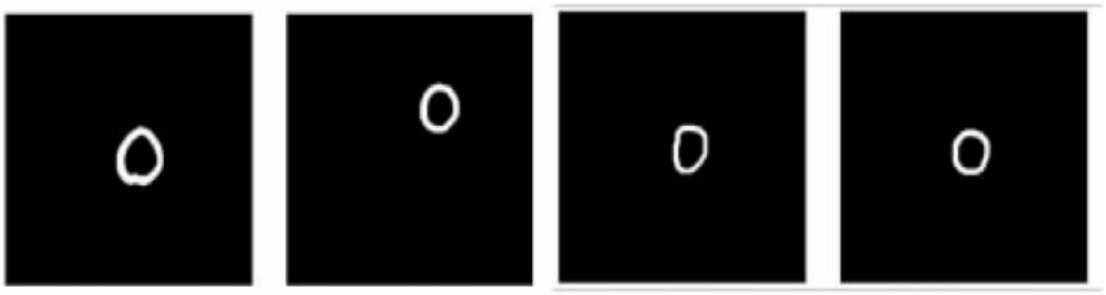
（b）标注图像

图 6 Cardiac MRI dataset 数据集的图像（图片来源：苑金辉等，计算机技术与发展，2021，31(06):35-39）

图 7 为将专家手动分割轮廓和文中算法分割轮廓进行二值化后得到的结果。



（a）专家手动分割



（b）使用算法分割

图 7 不同左心室心肌分割结果二值图（图片来源：苑金辉等，计算机技术与发展，2021，31(06):35-39）

（2） 模型构建

如图 8 所示，生成对抗网络由生成器(G) 和判别器(D) 两部分组成。

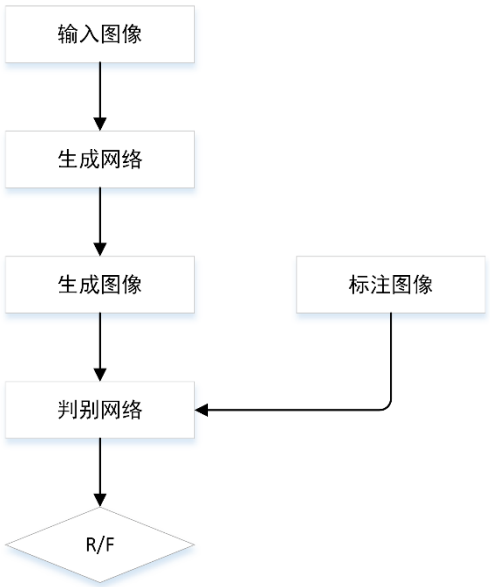


图 8 经典的生成对抗网络结构

用于分割任务的生成网络通过学习图像与标注数据之间的映射关系尽可能生成能够欺骗判别网络的分割图像，而判别器接收到分割图像后对图像进行判定，通过判定结果调整生成网络的参数。通过大量的迭代过程使生成网络和判别网络达到一个平衡，获得最优分割结果。生成器(G)和判别器(D)使用下述价值函数进行极大极小化博弈:

$$\min_{G}\max_{D}V(G,D)=E_{x\sim P_{data}(x)}[\log D(x)]+E_{z\sim P_{z}(z)}[\log(1-D(G(z)))] \quad (22)$$

其中，x 是来自未知分布 P_{data} 的真实图像，z 是来自概率分布(如高斯分布) P_z 的生成器的随机输入，G 和 D 分别表示 GAN 中生成网络和判别网络的参数。

然后作者对 GAN 进行改进，用于分割图像，网络整体结构如下：

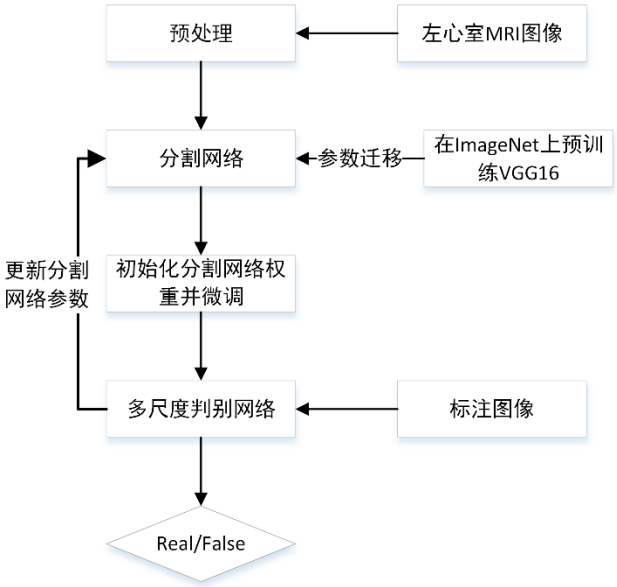


图 9 改进的生成对抗网络

在医学图像分割的研究中，数据集往往是限制研究的一个重要问题，由于标注数据难以获得，因此很多 数据集往往只有几十张或几百张图片，远远无法满足研究需求。而大规模的数据集往往可以使神经网络学习到更多的特征信息，获得更好的泛化能力。所以可以在较大的数据集上预训练，将得到的参数迁移到目标网络，用来自源网络的权重初始化目标网络，然后目标网络其他层将被随机初始化，再通过逐步微调在目标网络上训练目标数据集，提升分割效果。

作者使用全卷积神经网络做分割网络。分割网络由编码模块和解码模块两部分组成，编码模块采用 VGG16 的结构，解码模块通过卷积层和上采样层逐层 对齐编码模块。为了获得精确的分割结果，使用跳跃连接(skip－connection) 学习目标的多尺度信息。

另外，该作者采用图 10 所示的迁移学习方法实现左心室分割的分割网络。首先尝试将从 ImageNet 数据集学习到的自然图像的特征信息转移到目标数据集(即左心室 MRI 数据集) 的左心室分割目标任务上，但是由于自然图像与医学图像相似度较低，分割网络很难通过先验知识学习到足够的医学图像特征，所以需要将预训练网络在目标数据集上进行二次训练，根据源域和目标域的大小和相似程度选择对预训练网络进行逐层微调使网络自适应地调整网络参数，避免直接微调整个网络导致产生过大的计算量。通过迁移学习可以在小目标数据集上训练较大的网络模型而不会出现严重的过拟合。

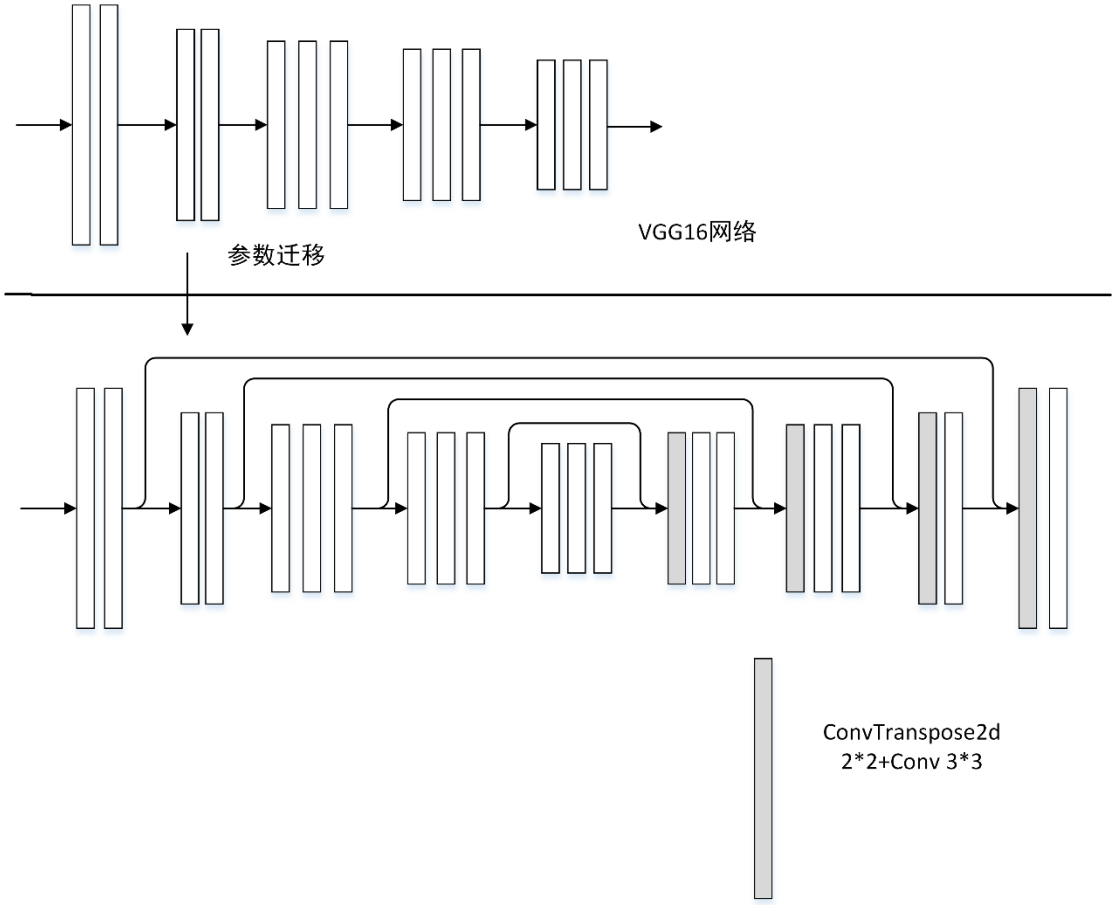


图 10 基于迁移学习的分割网络

在经过参数迁移后，作者再通过逐层微调方法对目标网络进行自适应参数调整，以提高网络学习特征的能力。如图 11 所示，通过逐层冻结与释放卷积层，微调释放卷积层的参数，在微调过程中通过释放各层后的分割效果确定微调的有效层数，选取最佳分割网络。

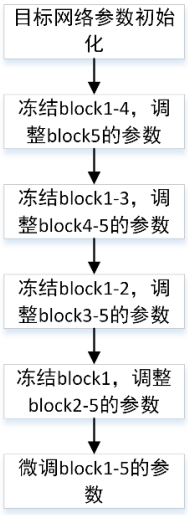


图 11 逐层微调流程

（3） 损失函数

实验表明 L1 均方误差损失函数与生成对抗网络的目标函数相结合能够得到更好的分割结果，L1 均方误差损失函数如下：

$$L_{mae}(F(x),y)=\frac{1}{N}\sum_{n=1}^{N}\| F(x_i)-Y_i \| \tag{23}$$

最终，总的损失函数如下：

$$\min_{G}\max_{D}V(G,D)+\beta L_{mae}(F(x),y) \tag{24}$$

（4） 结果分析与对比

作者使用Dice 系数、Jaccard 系数和灵敏度(Sensitivity) 三种评价指标对分割准确度进行度量。

Dice 系数表达式如下式所示，Dice 系数值越大，代表分割后的结果与专家手动分割图像越相似，准确率越高。

$$\mathrm{Dice}=2\left|X\cap Y\right|\left|X\right|+\left|Y\right|$$

Jaccard 系数表达式如下式所示，该项指标代表 专家手工标注图像与分割得到的左心室轮廓区域的交 集区域面积与并集区域面积的占比。

$$\mathrm{J}=\left|X\cap Y\right|\left|X\cup Y\right|$$

灵敏度(Sensitivity) 表达式如下所示，灵敏度代表所有正样本划分正确的比例。

$$Sensitivity=\left|X\cap Y\right|\left|X\right|$$

其中，Y 为专家手动分割左心室的目标和背景的像素集，X 为实验分割左心室的目标和背景像素集。

整体代码由 Pytorch 实现，Batch Size 大小设置为 10，激活函数选择 ReLU，由于使用迁移学习，所以学习率要设置的比较小，设为 0. 000 2。分割网络与判别网络均使用 Adam 优化算法进行优化。

表 3 列出了 TLMDB GAN、TLBSN、U－net 网络和研究左心室分割的其他文献的方法在文中数据集上得到的实验结果。由表 1 结果能够看出 TLMDB GAN 分割内膜和外膜的 Dice 系数分别为 0.9399 和 0. 9697，相比其他方法高出 0.01; 在 Jaccard 系数和 Sensitivity 系数上均超过了其他方法，达到了当前最高精度，比其他方法高出 0.02。

表 3 Cardiac MRI 数据集的实验结果 （数据来源：苑金辉等，计算机技术与发展，2021，31(06):35-39）

方法	Dice		Jaccard		Sensitivity	
	内膜	外膜	内膜	外膜	内膜	外膜
Nasr(2018)	~	0.8724	~	~	~	0.8769
Nasr(2018)	0.9258	0.9606	0.8692	0.9037	0.9224	0.9583
U-net	0.8829	0.9292	0.8536	0.8843	0.8910	0.9314
TLBSN	0.9307	0.9623	0.8764	0.9281	0.9232	0.9503
迁移学习对抗网络	0.9399	0.9697	0.8968	0.9415	0.9363	0.9686

习题

为了让读者更加熟悉迁移学习的基准模型，我们给读者提供了一个小 demo 来进行上手实践。

我们使用 flower17 数据集，它是一个包含 17 种花卉类别的数据集，每个类别有 80 张图像。收集的花都是英国一些常见的花，这些图像具有大比例、不同姿态和光线变化等性质。下载链接为：https://github.com/ck196/tensorflow-alexnet/blob/master/17flowers.tar.gz。也可从数据集的官方网站进行下载，链接为：https://www.robots.ox.ac.uk/~vgg/data/flowers/17/。

下载并解压数据后，可以看到内含 17 个子文件夹，对应 17 中不同的花卉类别。为加快实验速度，我们可以任选两类进行训练。注意到每个子文件夹下含有 80 张图像，可按照 7:1 的比例将其划分为训练集和验证集。以选择前两类花卉数据为例，按照以下目录结构放置文件：

```
当前目录
|--data
    |-- train
        |--0
```

```
        |--image_0001.jpg
        |--image_0002.jpg
        |--...
        |--image_0070.jpg
    |--1
        |--image_0081.jpg
        |--image_0082.jpg
        |--...
        |--image_0150.jpg
|--validation
    |--0
        |--image_0071.jpg
        |--image_0072.jpg
        |--...
        |--image_0080.jpg
    |--1
        |--image_0151.jpg
        |--image_0152.jpg
        |--...
        |--image_0160.jpg
```

由于样本量较小，我们在预训练的 VGG16 网络之上构建分类器，在训练中冻结 VGG16 网络的前几层，对后面几层进行 finetuning。以下介绍每一环节的具体代码。

为了实现，首先导入所有必需的库，包括应用程序、预处理、模型检查点以及相关对象，cv2 库和 NumPy 库用于图像处理和数值的基本操作。

```
from keras import applications
from keras.preprocessing.image import ImageDataGenerator
from keras import optimizers
from keras.models import Model
from keras.layers import Dropout, Flatten, Dense
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.applications.vgg16 import preprocess_input
import cv2
```

```
import numpy as np
```

定义输入、数据源及与训练参数相关的所有变量。

```
img_width, img_height = 224, 224
train_data_dir = "data/train"
validation_data_dir = "data/validation"
nb_train_samples = 300
nb_validation_samples = 100
batch_size = 16
epochs = 100
```

调用 VGG16 预训练模型，其中不包括顶部的平整化层。冻结不参与训练的层，这里我们冻结前五层，然后添加自定义层，从而创建最终的模型。注意以下代码会自动从官方网址下载预训练的 VGG16 模型，若自动下载失败，则需复制下载链接进行手动下载，并将下载的模型文件放置于对应路径下。Windows 环境需放置于 C:\Users\user_name\.keras\models 目录，Linux 和 Mac 环境一般放置于~/.keras/models/目录。

```
model = applications.VGG16(weights="imagenet", include_top=False, input
_shape=(img_width, img_height, 3))
for layer in model.layers[:5]:
    layer.trainable = False
x = model.output
x = Flatten()(x)
x = Dense(1024, activation="relu")(x)
x = Dropout(0.5)(x)
x = Dense(1024, activation="relu")(x)
predictions = Dense(2, activation="softmax")(x)
model_final = Model(inputs=model.input, output=predictions)
```

接着开始编译模型，并为训练、测试数据集创建图像数据增强生成器。

```
model_final.compile(loss="categorical_crossentropy", optimizer=optimizers
.SGD(lr=0.0001, momentum=0.9), metrics=["accuracy"])
train_datagen = ImageDataGenerator(rescale=1. / 255, horizontal_flip=T
```

```
rue, fill_mode="nearest", zoom_range=0.3, width_shift_range=0.3, height
_shift_range=0.3, rotation_range=30)
test_datagen = ImageDataGenerator(rescale=1. / 255, horizontal_flip=Tr
ue, fill_mode="nearest", zoom_range=0.3, width_shift_range=0.3, height_
shift_range=0.3, rotation_range=30)
```

生成增强后新的数据，根据情况保存模型。

```
train_generator = train_datagen.flow_from_directory(train_data_dir, tar
get_size=(img_height, img_width), batch_size=batch_size, class_mode="ca
tegorical")
validation_generator = test_datagen.flow_from_directory(validation_data_
dir, target_size=(img_height, img_width), class_mode="categorical")
checkpoint = ModelCheckpoint("vgg16_1.h5", monitor='val_acc', verbose=1
, save_best_only=True, save_weights_only=False, mode='auto', period=1)

early = EarlyStopping(monitor='val_acc', min_delta=0, patience=10, ver
bose=1, mode='auto')
```

开始对模型中新的网络层进行拟合。

```
model_final.fit_generator(train_generator, samples_per_epoch=nb_train_sam
ples, nb_epoch=epochs, validation_data=validation_generator, nb_val_sam
ples=nb_validation_samples, callbacks=[checkpoint, early])
```

训练完成后用水仙花图像测试这个新模型，输出的正确值应该为接近[1.,0.]的数组。

```
im = cv2.resize(cv2.imread('data/validation/image_0071.jpg'), (img_widt
h, img_height))
im = np.expand_dims(im, axis=0).astype(np.float32)
im = preprocess_input(im)
out = model_final.predict(im)
print(out)
print(np.argmax(out))
```

请读者自行尝试运行代码，测试运行效果。

阅读材料

迁移学习最权威的综述作者是香港科技大学杨强教授团队的 A survey on transfer learning [Pan and Yang, 2010]。这篇综述性作者是一篇迁移学习入门者的必读作者，分门别类介绍了迁移学习和其他几个相关领域的区别，并对迁移学习研究现状做了总结，非常具有权威性和阅读价值。

本文所介绍的迁移学习三个实例来源在此，读者可自行查找阅读。

1. 基于深度迁移学习模型的花卉种类识别[Liu, 2019]
2. 基于最优传输特征选择的医学图像分割迁移学习[Wang, 2021]
3. 融合迁移学习的土壤湿度预测时空模型[Wang, 2021]