Transfer learning

handbook

目录

1. 迁移学习的介绍

1.1 迁移学习的定义

1.1.1 数学符号表示

在这里介绍本文中将要用到的一些数学符号表示以及定义。首先，给出域（domain）和任务（task）的定义。

（1）域（domain）

一个域 D 由两部分组成：特征空间 X 和边际概率分布 P（X），其中 $X = \{x_1,...,x_n\} \in \chi$。比如，一个学习任务是文本分类，每个部分都看成一个二元特征，那么 X 是所有向量空间，x_i是文本对应的第 i 个向量，X是学习样本。一般地，如果两个域是不同的，那么它们要么特征空间不同，要么边际概率分布不同。

（2）任务（task）

给定域 D=｛X，P(X)｝后，一个任务由两部分组成：标签空间 Y 和观测不到但是可以从训练数据中学习得到的目标预测函数 f（·），任务 T=｛Y，f（·）｝，其中训练数据由｛x_i,y_i｝组成，$x_i \in X$,$y_i \in Y$。

本文中，只考虑一个源域(source domain)D_S和一个目标域(target domain)D_T的情况，将 source domain 表示为$D_S = \{(x_{S_1},y_{S_1}),...,(x_{S_{n_S}},y_{S_{n_S}})\}$，其中$x_{S_i} \in \chi_S$是数据样本，$y_{S_i} \in \mathrm{Y}_S$是对应的类别标签。

1.1.2 迁移学习的定义

给定源域D_S、学习任务T_S、目标域D_T和学习任务T_T，迁移学习的目标是利用D_S和T_S的信息（$D_S \neq D_T$或$T_S \neq \mathrm{T}_T$），来提升D_T的目标预测函数$f_T(\cdot)$的学习效果。

1.2 迁移学习的种类

迁移学习分为三个子类：归纳式迁移学习(Inductive Transfer learning)、直推式迁移学习(Transductive Transfer Learning)、无监督迁移学习(Unsupervised Transfer Learning)。Table 1 中给出了三个子类的条件的比较。

Transfer Learning Settings	Related Areas	Source Domain Labels	Target Domain Labels	Tasks
Inductive Transfer Llearning	Multi-task Learning	Available	Available	Regression, Classification
	Self-taught Learning	Unavailable	Available	Regression, Classification
Transductive Transfer Llearning	Domain Adaption, Sample Selection Bias, Co-variate Shift	Available	Unavailable	Regression, Classification
Unsupervised Transfer Learning		Unavailable	Unavailable	Clustering, Dimensionality Reduction

Table 1 Transfer Learning 三种子类的不同条件

1.2.1 归纳式迁移学习(Inductive Transfer Learning)

归纳式迁移学习(Inductive Transfer learning)：目标任务和源任务不同，无论源域和目标域是否相同。目标域的标签数据要被归纳为目标域中的目标预测模型$f_T(\cdot)$。归纳式迁移学习分为两种情况：

（1）源域中有大量有标签的数据。

（2）源域中没有被标记的数据。

1.2.2 直推式迁移学习(Transductive Transfer Learning)

直推式迁移学习(Transductive Transfer Learning)：源任务和目标任务相同，源域和目标域不同。

此时，当源域中有很多有标签数据时，目标域中的数据是无标签的。由于源域和目标域的不同，我们可以将直推式迁移学习分为两类：

（1）源域和目标域的特征空间不同，即$X_S \neq X_T$。在自然语言处理的背景下，这通常被称为跨语言适应（cross-lingual adaptation）。

（2）源域和目标域的特征空间相同（$X_S = X_T$），但源域和目标域 的边际概率分布不同，即 $P(X_S) \neq P(X_T)$。这个情景通常被称为域适应（domain adaptation）。

1.2.3 无监督迁移学习(Unsupervised Transfer Learning)

无监督迁移学习(Unsupervised Transfer Learning)的条件与归纳式迁移学习的条件相似，目标任务与源任务不同但是相关。但是无监督迁移学习着重于解决目

标域中的无监督学习任务，比如聚类、降维和密度估计等。此时，源域和目标域都没有被标记的数据。

1.3 迁移学习的方法

迁移学习中，对于“迁移什么”有四种方法：（1）基于样本的迁移学习、（2）基于特征的迁移学习、（3）基于参数的迁移学习、（4）基于关系的迁移学习。Table 2 给出了这四种方法的简述。

Transfer Learning Approaches	Brief Description
Instance-Transfer	在源域中找到与目标域相似的数据，把这个数据的权值进行调整，使得新的数据与目标域的数据进行匹配。
Feature-representation-transfer	假设源域和目标域含有一些共同的交叉特征，通过特征变换，将源域和目标域的特征变换到相同空间，使得该空间中源域数据与目标域数据具有相同分布的数据分布，然后进行传统的机器学习。
Parameter-transfer	假设源域和目标域共享模型参数，是指将之前在源域中通过大量数据训练好的模型应用到目标域上进行预测。
Relational-knowledge-transfer	假设两个域是相似的，那么它们之间会共享某种相似关系，将源域中的 关系应用到目标域上来进行迁移。

Table 2 迁移学习四种方法的简述

这四种方法在迁移学习的三个子类中有不同的应用，下面将详述这四种方法在归纳式迁移学习、直推式迁移学习和无监督迁移学习中的具体算法。

1.3.1 归纳式迁移学习中的方法介绍

1.3.1.1 基于样本的迁移学习（Transferring Knowledge of Instances）

归纳式迁移学习中的样本迁移方法表明，尽管源域不能直接被再使用，但是目标域中的一些有标签的样本可以用来训练数据。

Dai *et al.*[2]提出了一种在 Adaboost 算法上延伸的 Tradaboost 算法，这种算法的提出是为了解决归纳式迁移学习中的问题。Tradaboost 假定源域和目标域使用相同的的特征集合和标签集合，但是这两个域的分布不同。也由此，Tradaboost 还假定源域中的一些数据对目标域的预测可能起负作用。Tradaboost 通过对源域迭代再赋权的方式来减小“坏”的源数据的影响，让“好”的源数据对目标域的贡献在迭代中不断增加。在每轮迭代中，Tradaboost 在赋权的源数据和目标域数据上训练基分类器，只计算在目标域数据上的误差。Tradaboost 还借鉴了 Adaboost 对目标域中错分类的样本进行更新的思想，在源域中更新错分类样本。

关于样本迁移，还有其他学者做了其他改进研究，在此不做深入介绍。

1.3.1.2 基于特征的迁移学习（Transferring Knowledge of Feature Representations）

基于特征表示的迁移方法旨在寻找“好”的特征来最小化域间差异和分类回归误差。不同类型的源域数据有不同的方法来寻找“好”特征。下面介绍有监督和无监督的两种特征构建方法。

（1） 有监督的特征构建

如果源域中有很多有标签的数据的话，构建特征表示可以使用有监督的学习方法。在多任务学习中，有监督的特征构建和一般的特征学习相似。基本的思想是学习相关任务中共有的低维表示，而且习得的新的表示可以减小分类或回归模型的误差。

Argyriou et al.[3]提出了一种针对多任务学习的稀疏特征学习方法。在归纳式迁移学习的条件下，一般特征可以通过解决下列优化问题来习得：

$$\arg\min_{A,U} \sum_{t\in\{T,S\}} \sum_{i=1}^{n_t} L\left(y_{t_i}, < a_t, U^T x_{t_i} >\right) + \gamma \| A \|_{2,1}^2 \qquad (1.1)$$

s.t. $U \in O^d$

在这个表达式中，S 和 T 表示源域和目标域的任务，$A = [a_S, a_T] \in R^{d\times 2}$是参数矩阵，U 是一个 $d \times d$ 的正交矩阵（映射函数）把正交高维数据映射为低维表达。A 的(r,p)范数定义为$\| A \|_{r,p} := (\sum_{i=1}^{d} \| a^i \|_r^p)^{\frac{1}{p}}$。对于优化问题（1.1），要同时估计的是低维表示$U^T X_T, U^T X_S$和参数 A。（1.1）式可以转化为一个等价的凸优化公式来有效解决。

（2）无监督的特征构建

在文献[4]中，Raina et al.提出，针对迁移学习中的高维特征，可使用一种无监督的特征构建方法——稀疏编码（sparse coding）。这个方法的基本思想由两个步骤组成。

Step 1 ：高维基向量 $b = \{b_1, b_2, ..., b_s\}$是通过求解优化问题（1.2）得到的：

$$\min_{a,b} \sum_i \| x_{S_i} - \sum_j a_{S_i}^j b_j \|_2^2 + \beta \| a_{S_i} \|_1 \qquad (1.2)$$

$$s.t. \qquad \| b_j \|_2 \le 1, \forall j \in 1, ..., s$$

在这个表达式中，$a_{S_i}^j$是在输入x_{S_i}后，基向量b_j的新的表示；β是平衡特征部分和正则化部分的系数。

Step 2 ：得到b_j后，为得到目标域数据上的高维特征，求解优化算法（1.3）：

$$a_{T_i}^* = \arg\min_{a_{T_i}} \| x_{T_i} - \sum_j a_{T_i}^j b_j \|_2^2 + \beta \| a_{T_i} \|_1 \quad (1.3)$$

最后，对$\{a_{T_i}^*\}$及对应的标签应用判别算法来训练目标域的分类或回归模型。

1.3.1.3 基于参数的迁移学习（Transferring Knowledge of Parameters）

大部分参数迁移方法都假定相关任务的单独的模型应该有共同的参数或者超参数的先验分布。本文中介绍的方法都包含正则化和分级贝叶斯结构来适应多任务学习。多任务学习同时学习源任务和目标任务，但迁移学习仅仅使用源域舒俱来提升目标域的表现，所以在多任务学习中，源域和目标域的损失函数的权重是相同的，但迁移学习中的不同域的损失函数的权重不同。通常，令目标域的损失函数权重更大，以达到目标域表现更好的目的。

Evgenious 和 Pontil[5]借鉴了多任务学习中 SVMs 的分级贝叶斯框架(hierarchical Bayesian framework , HB)的思想。他提出的方法里假定了参数 w 在 SVMs 中的每个任务中都能分成两个部分：一个是所有任务的一般部分，另一个是每个任务特定的的部分，表达式如下：

$$w_S = w_0 + v_S \,,\ w_T = w_0 + v_T$$

其中，w_S和w_T分别是 SVMs 中源任务和目标任务的参数，w_0是一般参数，v_S和v_T分别是源任务和目标任务的特定参数。假设$f_t = w_t \cdot x$ 是任务 t 的超平面，多任务学习情境中 SVMs 的一个扩展可以写成如下形式：

$$\min_{w_0, v_t, \xi_{t_i}} \quad J(w_0, v_t, \xi_{t_i}) \quad (1.4)$$

$$= \sum_{t\in\{S,T\}} \sum_{i=1}^{n_t} \xi_{t_i} + \frac{\lambda_1}{2} \sum_{t\in\{S,T\}} \| v_t \|^2 + \lambda_2 \| w_0 \|^2$$

$$s.t.\ \ y_{t_i}(w_0 + v_t) \cdot x_{t_i} \geq 1 - \xi_{t_i},$$

$$\xi_{t_i} \geq 0 \,,\ i \in \{1,2,\ldots,n_t\} \text{ 且 } t \in \{S,T\}$$

通过求解（1.4），可以同时得到w_0, v_S, v_T。

1.3.1.4 基于关系的迁移学习（Transferring Relational Knowledge）

关系迁移处理相关的域间的迁移学习问题，其中数据不是独立同分布的，而且可以表示为多种关系，比如网络数据和社交网络数据。这个方法和传统方法不同域，它不假设每个域中的数据是独立同分布的，它从源域迁移相关关系到目标域。本文中，将介绍解决这些问题的统计相关性学习技术。

Mihalkova *et al.*[6]提出了 *TAMAR* 算法，通过马尔科夫逻辑网络——Markov

Logic Networks（MLNs）来迁移相关的信息。MLNs[7]是很有效的形式，在统计的相关性学习中，它把一阶逻辑的简洁表达和概率的灵活性结合起来。在 MLNs 中，相关的域中的个体被谓词（predicates）表示，它们的关系表示为一阶逻辑（first-order logic）。*TAMAR* 中，若两个域是相关的，那么从源域到目标域可能存在联系实体和关系的映射。比如，一个教授既可以属于学术领域，也可以属于工业管理的领域。而且教授和学生的关系也可以近似为管理者和工作者的关系。因此，存在一个教授到管理者的映射，也存在一个教授——学生关系到管理者——工作者关系的映射。

TAMAR 是一个两阶段算法，具体步骤如下：

Step 1 ： 基于加权伪似然估计（weighted pseudo loglikelihood measure,WPLL）构建从源 MLN 到目标域的映射；

Step 2 ： 通过 *FORTE* 算法对目标域中的映射结构进行修正。*FORTE* 算法[8]是修正一阶理论的归纳式逻辑程序（inductive logic programming，ILP）算法。这种修正的 MLN 可以用于关系模型在目标域上的推断或推理。

Mihalkova 后来将 *TAMAR* 延伸到单实体中心（single-entity-centered），即目标域中只有一个实体，此处不作详述。

1.3.2 直推式迁移学习中的方法介绍

在直推式迁移学习中，要明确一个条件：源域和目标域的任务是相同的，而且目标域中有无标签的数据。直推式迁移学习中，由于源域和目标域的不同分成了两个情况：（1）特征空间不同；（2）数据的边际概率分布不同，这两种情况也对应了预适应（Domain Adaptation）和样本选择偏差（Sample Selection Bias）。下述的方法主要针对情景（2），也就是特征空间相同但边际概率分布不同的情景。

1.3.2.1 基于样本的迁移学习（Transferring Knowledge of Instances）

大多数直推式迁移学习中的样本迁移都是基于重要度抽样的。为了探究重要度抽样起的作用，先回顾经验风险最小化（Empirical Risk Minimization，ERM）[9]的问题。一般地，通过最小化期望风险来得到模型的最优参数，

$$\theta^* = arg\min_{\theta\in\Theta} \mathbb{E}_{(x,y)\in P}[l(x,y,\theta)]$$

其中 $l(x,y,\theta)$是依赖参数θ的损失函数。

因为很难估计概率分布 P，所以转化为最小化 ERM 来计算：

$$\theta^* = arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n}[l(x_i,y_i,\theta)]$$ ，其中 n 是训练样本容量。

在直推式迁移学习的设定中，我们想通过最小化期望风险得到目标域的最优模型：

$$\theta^* = arg\min_{\theta\in\Theta} \sum_{(x,y)\in D_T} P(D_T) l(x,y,\theta)$$

因为训练集中目标域的数据没有标签，所以要从源域数据中得到模型。若$P(D_S) = P(D_T)$，那么可以通过求解下述目标域中的优化问题来得到模型：

$$\theta^* = arg\min_{\theta\in\Theta} \sum_{(x,y)\in D_S} P(D_S) l(x,y,\theta)$$

另外，当$P(D_S) \neq P(D_T)$时，我们需要修改上式来得到有高度泛化能力的模型，如下：

$$\theta^* = arg\min_{\theta\in\Theta} \sum_{(x,y)\in D_S} \frac{P(D_T)}{P(D_S)} P(D_S) l(x,y,\theta)$$

$$\approx arg\min_{\theta\in\Theta} \sum_{i=1}^{n_S} \frac{P_T(x_{T_i},y_{T_i})}{P_S(x_{S_i},y_{S_i})} l(x_{S_i},y_{S_i},\theta) \quad (1.5)$$

因此，对每个样本(x_{S_i},y_{S_i})加入不同的惩罚值，每个样本对应的权重为$\frac{P_T(x_{T_i},y_{T_i})}{P_S(x_{S_i},y_{S_i})}$，可以得到目标域的精确模型。由于$P(Y_T|X_T) = P(Y_S|X_S)$，因此$P(D_S)$和$P(D_T)$间的差异由$P(X_T)$、$P(X_T)$和$\frac{P_T(x_{T_i},y_{T_i})}{P_S(x_{S_i},y_{S_i})}$引起。若估计出了$\frac{P(x_{S_i})}{P(x_{T_i})}$，就可以解决直推式迁移学习的问题。

估计$\frac{P(x_{S_i})}{P(x_{T_i})}$的方法有很多，Huang *et al.*[10]提出核均值匹配(Kernel-mean Matching，KMM)算法，通过匹配源域和目标域在再生核希尔伯特空间(reproducing-kernel Hilbert space，RKHS)上的均值来直接得到$\frac{P(x_{S_i})}{P(x_{T_i})}$。KMM 可以写为下面的二次程序(quadratic programming，QP)优化问题：

$$\min_{\beta} \frac{1}{2}\beta^T K\beta - \mathcal{K}^T\beta \quad (1.6)$$

$$s.t.\ \beta_i \in [0,B] \text{ 且 } |\sum_{i=1}^{n_S} \beta_i - n_S| \leq n_S\epsilon$$

其中$K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix}$，$K_{ij} = k(x_i,x_j)$。$K_{S,S}$和$K_{T,T}$是源域和目标域的核矩阵。当$x_{T_j} \in X_T$时，$\mathcal{K}_i = \frac{n_S}{n_T}\sum_{j=1}^{n_T} k(x_i,x_{T_j})$，其中$x_i \in X_S \cup X_T$。

可证明$\beta_i = \frac{P(x_{S_i})}{P(x_{T_i})}$[10]。使用KMM的优点在于它可以避免表达$P(x_{S_i})$和$P(x_{T_i})$的密度估计，当数据集很小时这比较困难。

1.3.2.2 基于特征的迁移学习（Transferring Knowledge of Feature Representations）

大多数直推式迁移学习的特征迁移方法是在无监督学习框架下的。Blitzer *et al.*[11]提出了一种结构一致学习(structural correspondence learning,SCL)算法，它使用目标域中的无标签数据来提炼能减小域间差异的相关特征。SCL 分为以下几个步骤：

Step 1 ：在两个域的无标签数据上定义轴特征(pivot feature)的数据集(轴特征的数量为 m)。

Step 2 ：把轴特征移出数据，把每个轴特征当作一个新的标签向量，建立 m 个分类问题。假定每个问题都通过线性分类器求解，如下式：

$$f_l(x) = sgn(w_l{}^T \cdot x)，\ l = 1,...,m$$

SCL 可以得到一个参数矩阵 $W = [w_1 w_2 ... w_m]$。

Step 3 ：对参数矩阵 W 进行奇异值分解（SVD）。令 $W = UDV^T$，那么$\theta = U^T_{[1:h,:]}$（h 是公共特征的数量）的行是 W 的上左奇异向量(top left singular vectors)。

Step 4 ：对扩张的特征向量使用标准判别算法来建立模型。扩张的特征向量包含所有的原始特征x_i加上新的公共特征θ_{x_i}。文献[11]中提到，若轴特征设定的较好，那么得到的映射编码了不同域里特征的一致性。尽管 SCL 可以减小域间差异，但是怎样选择轴特征是困难而且依域而定。

1.3.3 无监督迁移学习中的方法介绍

无监督迁移学习在训练时，源域和目标域中的数据都没有标签。该领域的研究成果有 Self-taught clustering(STC)算法和迁移判别分析算法(transferred discriminative analysis,TDA)，它们是为了解决迁移聚类和迁移降维问题而提出的。

1.3.3.1 基于特征的迁移学习（Transferring Knowledge of Feature Representations）

Dai *et al.*[12]提出了自学习聚类(STC)，它属于无监督迁移学习，这种方法旨在通过源域中大量无标签数据来将目标域中的一小部分无标签数据聚类。STC 试图得到域间的公共特征空间，以便在目标域中聚类。STC 的目标函数如下：

$$J(\widetilde{X_T},\widetilde{X_S},\widetilde{Z}) = I(X_T,Z) - I(\widetilde{X_T},\widetilde{Z}) + \lambda[I(X_S,Z) - I(\widetilde{X_S},\widetilde{Z})] \quad (1.7)$$

其中，X_S和X_T是源域和目标域的数据。Z 是X_S和X_T的公共特征空间，$I(\cdot,\cdot)$是两个随机变量的共同信息。假设存在三个聚类函数C_{X_T}：$X_T \to \widetilde{X_T}$，C_{X_S}：$X_S \to \widetilde{X_S}$，C_Z：$Z \to \tilde{Z}$，其中$\widetilde{X_T}$、$\widetilde{X_S}$、$\tilde{Z}$是X_T、X_S、Z 的对应聚类。STC 的目标是通过求解式（1.7）得到$\widetilde{X_T}$：

$$\arg\min_{\widetilde{X_T},\widetilde{X_S},\tilde{Z}} J(\widetilde{X_T},\widetilde{X_S},\tilde{Z}) \tag{1.8}$$

文献[12]中给出了求解式（1.8）的迭代算法。

文献[13]提出了迁移判别分析算法（TDA）来解决迁移降维问题。TDA 首先运用聚类方法来形成目标域中无标签数据的伪标签，然后对目标域和有标签的源域使用降维方法。迭代这两个步骤来得到目标域的最优子空间。

2. 归纳式迁移学习方法详述

2.1 样本迁移——Tradaboost

在 Tradaboost 中，建立分类模型时，使用和测试数据相同分布的有标签数据作为训练数据的一部分，称这样的小部分训练数据为同分布训练数据（same-distribution training data）。同分布训练数据的数量通常不足以训练一个好的分类模型。

训练数据中还有一部分过时的数据，它们的分布和测试数据不同，称为异分布训练数据（diff-distribution training data）。假设这些数据是充分的，但是这些数据由于和测试数据分布不同，同样不能训练处一个好的分类器。

2.1.1 数学符号表示

令X_s为同分布样本空间，X_d为异分布样本空间，$Y = \{0,1\}$为标签类别集合。布尔函数 c 是 X 到 Y 的映射，其中 $X = X_s \cup X_d$。训练数据表示为 $S = \{(x_i^t)\}$，其中$x_i^t \in X_s\ (i = 1,...,k)$。$k$是没有标签的测试集$S$的容量。训练数据及$T \subseteq \{X \times Y\}$分成$T_d$和$T_s$两部分：$T_d = \{(x_i^d,c(x_i^d)\}$代表异分布训练数据，其中$x_i^d \in X_d(i = 1,...,n)$；$T_s = \{(x_j^s,c(x_j^s)\}$代表同分布训练数据，其中$x_j^s \in X_s(j = 1,...,n)$。$n$ 和 m 分别是T_d和T_s的样本容量，*c(x)*返回样本 x 的标签。

组合训练集 $T = \{(x_i,c(x_i))\}$的定义如下：

$$x_i = \begin{cases} x_i^d, & i = 1,...,n; \\ x_i^s, & i = n+1,...,n+m. \end{cases}$$

在这里，T_d对应着旧域中的一些有标签数据，在训练中要再利用这些数据，

但这些数据中的有效部分是未知的。现在要做的是从新域T_s中标记一小部分数据，然后使用这些数据去发现T_d中的有效部分。

我们要解决的问题是：给定同分布数据T_s中的少量数据、异分布数据T_d中的很多数据和未标记的测试数据 S，目标是训练一个分类器$\hat{c}: X \to Y$ 来最小化未标记数据 S 的预测误差。

2.1.2 算法描述

现在给出 Tradaboost 算法的结构。Adaboost 算法是通过调整样本权重、组合弱学习器来得到相应的分类器的，但 Adaboost 假设训练集和测试集的分布相同。在 Tradaboost 中，同分布训练数据中仍使用 Adaboost 来建模，但对于异分布训练数据的样本，当他们在模型中由于分布改变而被误判时，这些样本就是和同分布样本极不相似的数据。因此，Tradaboost 加入了一个机制：降低这些错判样本的权重来减弱它们的作用。

Tradaboost 的算法步骤如下：

输入 两个有标签的数据T_d、T_s，
无标签数据 S,
学习算法 **Learner**
最大迭代次数 N

初始化 初始权重向量：$w^1 = (w_1^1,\ldots,w_{n+m}^1)$
初始权重可以自定义

For $t=1,\cdots N$

1. 令$p^t = w^t/(\sum_{i=1}^{n+m} w_i^t)$
2. 训练分类器，
使用组合训练数据 T和未标记的数据集 S，其中 T的分布为p^t
之后回到假设$h_t: X \to Y$（或置信区间为[0,1]）
3. 计算h_t在T_s上的误差：

$$\epsilon_t = \sum_{i=n+1}^{n+m} \frac{w_i^t \cdot |h_t(x_i) - c(x_i)|}{\sum_{i=n+1}^{n+m} w_i^t}$$

4. 令$\beta_t = \epsilon_t/(1-\epsilon_t)$，$\beta = 1/(1+\sqrt{2\ln n/N})$
注意$\epsilon_t < 1/2$
5. 更新权重向量

$$w_i^{t+1} = \begin{cases} w_i^t \beta^{|h_t(x_i) - c(x_i)|}, & 1 \le i \le n \\ w_i^t \beta_t^{-|h_t(x_i) - c(x_i)|}, & n+1 \le i \le n+m \end{cases}$$

Output 假设

$$h_f(x)=\begin{cases}1, \prod_{t=[N/2]}^{N}\beta_t^{-h_t(x)} \geq \prod_{t=[N/2]}^{N}\beta_t^{-1/2} \\ 0, \quad 其他\end{cases}$$

关于 Tradaboost 的理论分析和性质，详见文献[2]。

2.2 特征迁移——有监督与无监督算法

2.2.1 有监督特征构建——Multi-task Feature Learning

多任务在很多形式上有相关性，比如，假定所有函数在范数形式上是相近的，或者多任务中共享一般的特征表示。在有监督特征迁移中，考虑后一种情况，即得到多个相关任务的共同的低维表示。

该部分研究了学习数据特征的问题，通常这个问题是关于监督性学习任务的。尽管流行的正则化问题是非凸的，但是本文中，第一个关键结果是这个非凸问题等同于另外形式的凸优化问题。为了解决后者（凸优化问题），本文使用了一种迭代算法。这个算法通过两个交互步骤来同时学习特征和任务函数。第一步：独立地学习任务回归或分类函数的参数；第二步：通过非监督学习的方式，学习任务参数的低维表示。

2.2.1.1 学习稀疏多任务的表示(Representations)

（1） 数学符号表示

令 $\mathfrak{R}$ 为实数集，$\mathfrak{R}_+(\mathfrak{R}_{++})$为非负（正）子集。令 T 为任务（tasks）数量，定义$\mathbb{N}_T := \{1,2,...,T\}$。对每个任务 $t \in \mathbb{N}_T$，给定 m 个样本$(x_{t1},y_{t1}),...,(x_{tm},y_{tm}) \in \mathfrak{R}^d \times \mathfrak{R}$。基于这个数据，要估计 T 个函数$f_t: \mathfrak{R}^d \to \mathfrak{R}, t \in \mathbb{N}_T$，函数$f_t$较好地估计数据和进行统计预测。

若 $w,u \in \mathfrak{R}^d$，定义$\langle w,u\rangle := \sum_{i=1}^{d} w_i u_i$，这是$\mathfrak{R}^d$上的标准内积。对于每个 $p \geq 1$，定义向量 w 的 p-范数为$\|w\|_p := (\sum_{i=1}^{d} |w_i|^p)^{\frac{1}{p}}$。特殊情况：$\|w\|_2 := \sqrt{\langle w,w\rangle}$。若 A 是 $d \times T$ 维的矩阵，那么定义$a^i \in \mathfrak{R}^T$是矩阵 A 的第 i 行，$a_t \in \mathfrak{R}^d$是矩阵 A 的第 t 列。对于每个 $r,p \geq 1$，定义矩阵 A 的$(r,p)-\text{norm}$ 为$\|A\|_{r,p} := (\sum_{i=1}^{d} \|a^i\|_r^p)^{\frac{1}{p}}$。

定义S^d为 $d \times d$ 维的实对称矩阵，定义$S_+^d(S_{++}^d)$为半正定矩阵的子集，S_-^d是半负定矩阵的子集。若 D 是 $d \times d$ 矩阵，定义 $trace(D) := \sum_{i=1}^{d} D_{ii}$。若 $w \in \mathfrak{R}^d$，定义 $\text{Diag}(w)$或 $\text{Diag}(w_i)_{i=1}^d$是 w 的对角矩阵。若 X 是 $n \times q$ 的实矩阵，那么定义 range(X)为集合$\{x \in \mathfrak{R}^n: x = Xz, \text{for some } z \in \mathfrak{R}^q\}$。令$O^d$为 $d \times d$ 维的正交矩阵集。最后，若 D 是 $d \times d$ 维的矩阵，定义D^+是它的伪逆(pseudoinverse)。特殊情况：

若 $a \in R$，那么当 $a \neq 0$ 时，$a^{+} = \frac{1}{a}$。

（2）问题描述

假设函数f_t是相关的，那么它们会共享一小部分的特征。函数f_t可以表达为：

$$f_t(x) = \sum_{i=1}^{d} a_{it} h_i(x),\ t \in \mathbb{N}_T \qquad (2.1)$$

其中，h_i:$\mathfrak{R}^d \to \mathfrak{R}$，$i \in \mathbb{N}_T$是特征，$a_{it} \in \mathfrak{R}$ 是回归参数。这个表达的主要假设为：在所有任务中，大部分特征的系数为 0。

为了简化问题，在这里只考虑线性特征，即$h_i(\mathrm{x}) = \langle u_i,x \rangle$，其中$u_i \in \mathfrak{R}^d$且向量$u_i$是正交的。因此，若 U 表示 $d \times d$ 维的矩阵，且列向量为u_i，那么 $U \in O^d$。函数f_t是线性的，即$f_t(x) = \langle w_i,x \rangle$，其中$w_t = \sum_i a_{it} u_i$。

我们的目标是学习数据的特征h_i，参数a_{it}和特征数 N。为了简化，首先考虑特征是线性同质(*linear homogeneous*)函数，即：它们的形式都是$h_i(\mathrm{x}) = \langle u_i,x \rangle$，其中$u_i \in \mathfrak{R}^d$。在第五部分，将延伸这个公式到一种情况：$h_i$是再生核希尔伯特空间的成分，因此$h_i$一般是非线性的。非线性的情况可以使用核函数，详细推导见文献[]。

定义 W 为 $d \times T$ 维矩阵，W 的列向量为w_t，A 为 $d \times T$ 维矩阵，A 的列向量为a_{it}，令 $W = UA$。假设所有任务共享一小部分的特征，这意味着矩阵 A 中很多行都为零，因此对应的特征（矩阵 U 的列）不用来代表任务参数（矩阵 W 的列）。换句话说，矩阵 W 是低秩矩阵。

下面介绍计算特征向量u_i和参数a_{it}的方法。首先考虑单任务 t 和固定特征u_i的情况：在学习数据$\{(x_{ti},y_{ti})\}_{i=1}^{m}$的参数向量$a_t \in \mathfrak{R}^d$时，在$a_t$的非零部分的数量上界的约束下，来最小化经验误差$\sum_{i=1}^{m} L(y_{ti},\langle a_t,U^T x_{ti} \rangle)$，其中 L: $R \times R \to \mathfrak{R}_+$是给定的损失函数，$L$ 在第二部分上是凸的。这个较难处理的问题通常设置a_t的 1-范数的上界来缓解，即，考虑问题 $\min\{\sum_{i=1}^{m} L(y_{ti},\langle a_t,U^T x_{ti} \rangle) + \gamma \|a_t\|_1^2 : a_t \in \alpha\}$，或者等价于无约束问题

$$\min\{\sum_{i=1}^{m} L(y_{ti},\langle a_t,U^T x_{ti} \rangle) + \gamma \|a_t\|_1^2 : a_t \in \mathfrak{R}^d\} \qquad (2.2)$$

其中，$\gamma > 0$ 是正则化参数。使用 1-范数来得到稀疏解，即向量a_t的很多部分为零。而且，问题（2.2）的解的非零部分的数量是典型的γ的非增函数。

将问题（2.2）推广到多任务的情况，那么介绍正则化误差函数

$$\mathcal{E}(A,U) = \sum_{t=1}^{T} \sum_{i=1}^{m} L(y_{ti},\langle a_t,U^T x_{ti} \rangle) + \gamma \|A\|_{2,1}^2 \qquad (2.3)$$

式（2.3）的第一项是各个任务的平均经验误差；第二项是正则化参数，作为矩阵 A 的(2,1)-范数的判罚，先计算行向量a^i的 2-范数，再计算向量 $b(A) = (\|a^1\|_2,\|a^2\|_2,\ldots,\|a^d\|_2)$的 1-范数就可以得到矩阵 A 的(2,1)-范数。这个范数把任务都结合起来，保证了任务间的共同特征能被筛选出来。

事实上，若特征 U 提前给定，且$\widehat{A}$是最小化了函数 $\mathcal{E}$ 后所得 A 的估计值，那么向量 $b(\widehat{A})$的非零部分的个数是γ的非增函数，像单任务正则化的 1-范数一样。

而且，向量 $b(\widehat{A})$的每个元素体现了每个特征的重要度。

为了既筛选特征又学习这些特征，接下来最小化关于 U 的函数 $\mathcal{E}$，即考虑下列优化问题：

$$\min\{\mathrm{E}(A,U):U\in O^d,A\in\Re^{d\times T}\} \tag{2.4}$$

这个方法可以学到各个任务中的低维表示，在单任务情况中，特征个数是正则化参数的非增函数。当矩阵 U 没有被学习到时，设定 $U=I_{d\times d}$，那么式（2.4）可以转化为下列凸优化问题：

$$\min\{\sum_{t=1}^{T}\sum_{i=1}^{d}L(y_{ti},\langle a_t,x_{ti}\rangle)+\gamma\|A\|_{2,1}^2:A\in\Re^{d\times T}\} \tag{2.5}$$

2.2.1.2 等价凸优化问题

求解式（2.4）需要克服两个问题：（1）这是一个非凸问题，尽管在变量 A 和 U 上分别都是凸的；（2）范数$\|A\|_{2,1}$是非光滑的以致于难以优化。

式（2.4）可以转化为等价的凸问题：对每个 $W\in\Re^{d\times T}$，其中列为w_t，且 $D\in S_+^d$，定义函数：

$$\mathcal{R}(W,D)=\sum_{t=1}^{T}\sum_{i=1}^{m}L(y_{ti},\langle w_t,x_{ti}\rangle)+\gamma\sum_{t=1}^{T}\langle w_t,D^+w_t\rangle \tag{2.6}$$

定理 2.2.1.1 式（2.4）等价于下列问题（range 表示值域）：

$$\min\{\mathcal{R}(W,D):W\in\Re^{d\times T},D\in S_+^d,trace(D)\le 1,range(W)\subseteq range(D)\} \tag{2.7}$$

即，在$(\widehat{A},\widehat{U})$是式（2.4）的最优解时，要满足当且仅当$(\widehat{W},\widehat{D})=(\widehat{U}\widehat{A},\widehat{U}\mathrm{Diag}(\hat{\lambda})\widehat{U}^T)$是式（2.7）的最优解，

$$\text{其中}\hat{\lambda}_i:=\frac{\|\hat{a}^i\|_2}{\|\widehat{A}\|_{2,1}} \tag{2.8}$$

在式（2.7）中，给出了矩阵 D 的迹的约束，否则得到最优解时会使 $D=\infty$，且会仅仅最小化式（2.6）中的经验误差项。相似地，给出值域的约束来保证判罚项有下界，且远离零。如果没有这些约束，很可能会在矩阵 W 不满秩时，对于$\sum_{t=1}^{T}\langle w_t,D^+w_t\rangle=trace(W^TD^+W)=0$，存在一个矩阵 D，使得 $DW=0$。

矩阵 D 的秩体现了任务间共享的共同特征有多少，事实上，在等式（2.8）中，矩阵 D 的秩等于矩阵 A 的非零行的个数。

式（2.6）中的函数 $\mathcal{R}$ 在矩阵 W 和 D 上是联合凸的，那么定义函数 $f(w,D)=w^TD^+w$，其中 $D\in S_+^d$且 $w\in range(D)$，在其他情况下 $f(w,D)=+\infty$。显然，在 f 是凸函数时，$\mathcal{R}$ 也是凸函数，因为有直接的计算表明：f 是凸函数族的上确界，即 $f(w,D)=sup\{w^Tv+trace(ED:E\in S^d,v\in\Re^d,4E+vv^T\le 0\}$。

2.2.1.3 交替最小化算法(Alternating Minimization Algorithm)

下面通过交替最小化关于 D 和w_t的函数 $\mathcal{R}$ 来求解式（2.7）。

先固定矩阵 D，使用正则化方法单独学习参数w_t来最小化式（2.7），正则化方法可以选择 SVM 或岭回归，这样就可以得到w_t；对于得到的向量w_t，通过求解下列最小化问题来得到 D：

$$\min\left\{\sum_{t=1}^{T}\langle w_t, D^{+}w_t\rangle : D \in S_{+}^{d}, trace(D) \le 1, range(W) \subseteq range(D)\right\} \quad (2.9)$$

下面的定理详述了式（2.9）的最优解。

定理 2.2.1.2 式（2.9）的最优解为：

$$D = \frac{(WW^T)^{\frac{1}{2}}}{trace(WW^T)^{\frac{1}{2}}}$$ 且最优值等于$(trace(WW^T)^{\frac{1}{2}})^2$。

引理 2.2.1.3 对于任意的 $b = (b_1,\ldots,b_d) \in \Re^d$，有：

$$\inf\left\{\sum_{i=1}^{d}\frac{b_i^2}{\lambda_i} : \lambda_i > 0, \sum_{i=1}^{d}\lambda_i \le 1\right\} = \|b\|_1^2$$

且任意最小化的序列都收敛到$\hat{\lambda}_i = \frac{|b_i|}{\|b\|_1}, i \in \mathbb{N}_d$。

$trace(WW^T)^{\frac{1}{2}}$是矩阵 W 的奇异值的和，有时也成为迹范数。迹范数在单位球内是 *rank*(*W*)的凸包络，这也给出了秩与γ关系的另一种解释。使用迹范数可以使式（2.7）变成仅依赖于 W 的正则化问题。但是迹范数是不光滑的。

2.2.1.4 算法流程

在算法中，实际上是交替的进行了有监督和无监督步骤。在无监督步骤中，学习任务中的共同特征表示；在有监督步骤中，使用上一步学到的特征表示来学习各个任务特定的函数（task-specific functions）。

算法流程如下：

Algorithm	Multi-Task Feature Learning
输入	数据集$\{(x_{ti}, y_{ti})\}_{i=1}^{m}$， $t \in \mathbb{N}_T$
参数	正则化参数γ
输出	$d \times d$ 维矩阵 D， $d \times T$ 维回归矩阵 $W = [w_1,\ldots,w_T]$
初始化	设定 $D = \frac{I_{d\times d}}{d}$

当初始条件不满足时

For t=1,⋯,T

计算$w_t = argmin\{\sum_{i=1}^{m} L(y_{ti},\langle w,x_{ti}\rangle) + \gamma\langle w,D^{+}w\rangle : w \in \Re^{d}, w \in range(D)\}$

结束 for 循环

设置 $D = \frac{(WW^T)^{\frac{1}{2}}}{trace(WW^T)^{\frac{1}{2}}}$

结束 while 循环

2.2.2 无监督特征构建——Self-taught Learning

在现实生活中，有标签数据（labeled data）不易得到，但无标签数据（unlabeled data）很容易获取，在该算法中，将利用无标签数据的特征信息，来帮助有监督学习更好的学习标签数据。在这里，前提假设是：无标签数据的隐含类别标签（class labels）和一般分布（generative distribution）和有标签数据的不一定相同。

Self-taught Learning 包含两个步骤：

（1）学习无标签数据的表示（representation）；

（2）用学到的表示（representation）应用于有标签数据，将得到的新数据用

于分类任务。

2.2.2.1 问题表述

给定 m 个样本的有标签训练集：$\left\{\left(x_l^{(1)},y^{(1)}\right),\left(x_l^{(2)},y^{(2)}\right),...,\left(x_l^{(m)},y^{(m)}\right)\right\}$，这些样本独立同分布与分布 $\mathcal{D}$。$x_l^{(i)} \in \Re^n$是输入的特征向量，l 表示有标签样本；$y^{(i)} \in \{1,...,C\}$是对应的类别标签。再给定 k 个样本的无标签集合：$x_u^{(1)},x_u^{(2)},...,x_u^{(k)} \in \Re^n$，$u$ 表示有标签样本。不假定无标签数据$x_u^{(j)}$和有标签数据来自同一个分布或有同样的类别标签。

首先，学习无标签数据的高维表示。无标签数据$X_u = \left(x_u^{(1)},x_u^{(2)},...,x_u^{(k)}\right)^T$中，$x_u^{(i)} \in \Re^n$。通过如下优化问题来学习：

$$\min_{b,a} \sum_i \left\| x_u^{(i)} - \sum_j a_j^{(i)} b_j \right\|_2^2 + \beta \left\| a^{(i)} \right\|_1 \qquad (2.10)$$

$$\text{s.t. } \left\| b_j \right\|_2 \le 1,\ \forall j \in 1,...,s$$

在式（2.10）中，优化变量是基向量 $b = \{b_1,b_2,...,b_s\}$，其中$b_j \in \Re^n$；激活矩阵 $a = \{a^{(1)},a^{(2)},...,a^{(k)}\}$，其中$a^{(i)} \in \Re^s$，$a_j^{(i)}$是样本$x_u^{(i)}$在基向量$b_j$上的激活元素。基的个数 s 可以比输入维度 n 大很多。优化目标函数（2.10）平衡了两个项：（i）左面的二次项促使输入$x_u^{(i)}$被重构为基b_j的加权线性组合（对应的权重是激活元

素$a_j^{(i)}$）；（ii）右面的项令激活向量有较小的 L1 范数（产生稀疏性）。在对式（2.10）求解时，固定 a,b 中一个参数后，迭代优化另一个参数来得到优化结果。

其次，使用稀疏编码得到无标签数据的基 b 后，对有标签数据中的每个样本 $x_l^{(i)} \in \Re^n$，通过求解下面的优化问题来得到特征$\hat{a}(x_l^{(i)}) \in \Re^s$：

$$\hat{a}\left(x_l^{(i)}\right) = arg\min_{a^{(i)}} \left\| x_l^{(i)} - \sum_j a_j^{(i)} b_j \right\|_2^2 + \beta \left\| a^{(i)} \right\|_1 \qquad (2.11)$$

求解后的$\hat{a}(x_l^{(i)})$就是$x_l^{(i)}$的新表示，使用新数据$\left\{(\hat{a}(x_l^{(i)}), y^{(i)})\right\}_{i=1}^m$作为分类的训练集数据进行训练。

2.2.2.2 算法流程

Algorithm	Self-taught Learning via Sparse Coding
输入	有标签数据集 $T = \{(x_l^{(1)}, y^{(1)}), (x_l^{(2)}, y^{(2)}), \ldots, (x_l^{(m)}, y^{(m)})\}$ 无标签数据$\{x_u^{(1)}, x_u^{(2)}, \ldots, x_u^{(k)}\}$
输出	分类任务的已学习到的分类器
算法 **Step 1：**	利用无标签数据$\{x_u^{(i)}\}$，求解优化问题(2.10)得到基 b $$\min_{b,a} \sum_i \left\| x_u^{(i)} - \sum_j a_j^{(i)} b_j \right\|_2^2 + \beta \left\| a^{(i)} \right\|_1 \qquad (1)$$ $$\text{s.t. } \left\| b_j \right\|_2 \le 1,\ \forall j \in 1, \ldots, s$$
Step 2：	使用新得到的基 b，在有标签数据上求解优化问题(2.11)，得到新的有标签数据集$\hat{T} = \left\{(\hat{a}(x_l^{(i)}), y^{(i)})\right\}_{i=1}^m$ $$\hat{a}\left(x_l^{(i)}\right) = arg\min_{a^{(i)}} \left\| x_l^{(i)} - \sum_j a_j^{(i)} b_j \right\|_2^2 + \beta \left\| a^{(i)} \right\|_1 \qquad (2)$$ 在有标签数据集$\hat{T}$上应用有监督学习算法(如 SVM）来习得分类器 C
返回	习得的分类器 C

2.3 参数迁移——Regularized Multi-Task Learning

在很多实际的统计模型应用场景中，存在很多有相关关系的任务(task)，在学习这些任务时，同时学习多任务要优于独立地学习单个任务。在该部分，将介绍多任务的学习方法，这是基于核函数的单任务学习方法（比如 SVM）的延伸。

2.3.1 数学符号表示

现在有 T 个学习任务，假设所有任务的数据都来自相同的空间 $X \times Y$，其中 $X \subset \Re^d, Y \subset R$。在每个任务中，有 m 个数据点$\{(x_{1t},y_{1t}),(x_{2t},y_{2t}),...,(x_{mt},y_{mt})\}$，这 m 个样本属于空间 $X \times Y$ 且服从分布P_t。因此所有任务的数据可表示为：

$$\{\{(x_{11},y_{11}),...,(x_{m1},y_{m1})\},...,\{(x_{1T},y_{1T}),...,(x_{mt},y_{mt})\}\}$$

在这里假设每个任务的分布P_t不同但相关。目标是学习 T 个函数$f_1,f_2,...,f_T$，其中$f_t(x_{it}) \approx y_{it}$，$T = 1$ 时是单任务学习问题。

在这个设定下有很多情况。比较简单的情况是所有的任务有相同的输入数据 x_{it}，即：对于每个 $i \in \{1,...,m\}$，向量x_{it}对所有的 $t \in \{1,...,T\}$都是相同的，但是每个 t 对应的输出值y_{it}是不同的。比如，在市场偏好建模中，消费者会面临同样的选择范围（看成 x），每个消费者都有自己的偏好（看成 y），假设个体的偏好有共同点，那么这个共同关系可以看成函数f_t。

也可以考虑有相同输出（y）、不同输入（x）的情况，这是一个混合数据的信息整合问题；还有多模型学习或按成分学习的情况，在这个场景中，数据的（x,y）不属于空间 $X \times Y$，但是任务 t 的数据属于空间$X_t \times Y_t$，比如，在计算机视觉中，识别面部时通常先识别面部的组成部分，如五官。

2.3.2 多任务学习的方法

为了简化问题，先假定第 t 个任务的函数f_t是一个超平面，即$f_t(x) = w_t \cdot x$，其中“$\cdot$”表示$\Re^d$上的标准内积。对于非线性模型，可以通过使用再生核希尔伯特空间（Reproducing Kernel Hilbert Spaces,RKHS）来建立。在分类场景中，$y_{it} = \pm 1$，f_t是$w_t \cdot x$ 的 sign 函数。在回归中也可以用近似方法来解决。

之前的多任务学习大多基于任务相关的数学公式定义，相关性是通过多任务学习方法来刻画的。比如，等级贝叶斯方法（hierarchical Bayesian methods）假设所有的函数w_t都来源于特殊的概率分布，如高斯分布，这就表明所有的w_t都接近均值函数w_0（高斯分布的均值）。

借用等级贝叶斯的思想，特别地，假设对每个 $t \in \{1,...,T\}$，w_t都可以写成如下形式：

$$w_t = w_0 + v_t \tag{2.12}$$

其中，在任务彼此相似时，向量v_t很小。换句话说，在任务相关时，真实的模型会接近于模型w_0（等级贝叶斯中的高斯分布的均值）。在式（2.12）中，同时估计w_0和v_t。最后要解决下列优化问题，这个问题和单任务学习的 SVM 相似：

问题 2.3.1

$$\min_{w_0,v_t,\xi_{it}} \left\{ J(w_0,v_t,\xi_{it}) := \sum_{t=1}^{T}\sum_{i=1}^{m} \xi_{it} + \frac{\lambda_1}{T}\sum_{t=1}^{T} \|v_t\|^2 + \lambda_2 \|w_0\|^2 \right\} \tag{2.13}$$

使得对于所有的 $i \in \{1,...,m\}$和 $t \in \{1,...,T\}$，都满足

$$y_{it}(w_0 + v_t) \cdot x_{it} \geq 1 - \xi_{it} \quad (2.14)$$

$$\xi_{it} \geq 0$$

在式（2.14）中，λ_1和λ_2是正的正则化参数，ξ_{it}是衡量每个最终模型w_t的误差的松弛变量。提出平均模型w_0的正则化约束，通过控制v_t的数量来控制式（2.14）的解w_t的差异大小。直觉上来看，给定λ_2后，$\frac{\lambda_1}{\lambda_2}$的值越大，就越倾向于建立相同的模型（即$v_t$几乎等于 0）；给定$\lambda_1$后，$\frac{\lambda_1}{\lambda_2}$的值越小，任务越不相关（$w_0$几乎等于 0）。当$\lambda_1$无穷大时，问题 2.3.1 会降为单任务学习（对每个 $t \in \{1,...,T\}$，$v_t = 0$ 时，寻找满足条件的w_0）；当λ_2无穷大时，问题 2.3.1 会变为独立地求解 T 个任务（$w_t = 0$ 时，寻找满足条件的v_t）。

假设w_0^*和v_t^*是问题 2.3.1 的最优解，且满足$w_t^* := w_0^* + v_t^*$。下面的引理给出了部分变量的关系。

引理 2.3.1 式（2.14）的最优解满足以下方程：

$$w_0^* = \frac{\lambda_1}{\lambda_2+\lambda_1}\frac{1}{T}\sum_{t=1}^{T} w_t^* \quad (2.15)$$

这个引理表明，在式（2.14）中，可以使用v_t的表达式来代替w_0，从而得到仅包含v_t的优化方程。之后再选择合适的正则化参数，就引出了下面的引理：

引理 2.3.2 问题 2.3.1 等价于解决下面的优化问题：

问题 2.3.2

$$\min_{w_t,\xi_{it}} \left\{ \sum_{t=1}^{T}\sum_{i=1}^{m} \xi_{it} + \rho_1 \sum_{t=1}^{T} \|w_t\|^2 + \rho_2 \sum_{t=1}^{T} \left\| w_t - \frac{1}{T}\sum_{s=1}^{T} w_s \right\|^2 \right\} \quad (2.16)$$

使得对所有的 $i \in \{1,...,m\}$和 $t \in \{1,...,T\}$，都满足

$$y_{it} w_t \cdot x_{it} \geq 1 - \xi_{it} \quad (2.17)$$

$$\xi_{it} \geq 0$$

其中，ρ_1、ρ_2和λ_1、λ_2的关系式如下：

$$\rho_1 = \frac{1}{T}\frac{\lambda_1\lambda_2}{\lambda_1+\lambda_2} \quad (2.18)$$

$$\rho_2 = \frac{1}{T}\frac{\lambda_1^2}{\lambda_1+\lambda_2} \quad (2.19)$$

使用问题 2.3.2 中的正则化方法，可以找到每个模型参数个数和这些模型参数与平均模型参数接近程度的一个权衡。在 SVM 类方法中，可以找到每个 SVM 有的大边界（定义为$\frac{1}{\|w_t\|}$）和每个 SVM 与平均 SVM 的接近程度的一个权衡。

2.3.3 对偶优化问题（Dual Optimization Problem）

现在来求得问题 2.3.1 的对偶问题，下面的方法可以直接将问题 2.3.1 和标准 SVM 的对偶问题联系起来。

函数集$f_t(x) = w_t \cdot x, t = 1,...,T$ 可以视为实值函数 $F{:}X \times \{1,...,T\} \to R$.定义函数：

$$F(x,t) = f_t(x) \tag{2.20}$$

学习这个函数要求形如((x,t),y)的样本，其中$(x,t) \in X \times \{1,...,T\}$且 $y \in \{-1,1\}$。

假设核的特征映射的概念已知，详见文献[]。F 可以用特征映射来表示：

$$\Phi\big((x,t)\big) = \left(\frac{x}{\sqrt{\mu}}, \underbrace{0,\ldots,0}_{t-1}, x, \underbrace{0,\ldots,0}_{T-t}\right) \tag{2.21}$$

其中 $\mathbf{0}$ 是$\Re^d$上的坐标全为 0 的向量，$\mu = \frac{T\lambda_2}{\lambda_1}$。现在的目标是估计向量 $w = (\sqrt{\mu}w_0,v_1,...,v_T))$。在构建函数Φ后，可得：

$$w \cdot \Phi((x,t)) = (w_0 + v_t) \cdot x$$

$$\|w\|^2 = \sum_{t=1}^{T} \|v_t\|^2 + \mu\|w_0\|^2$$

很明显，解决 SVM 多任务学习问题（2.12）等价于学习式（2.20）的函数 F，可以通过使用基于特征映射（2.21）的核函数的标准 SVM 来实现。在使用标准 SVM 对偶问题时，有以下定理：

定理 2.3.3 令 $C := \frac{T}{2\lambda_1}, \mu = \frac{T\lambda_2}{\lambda_1}$，定义核函数：

$$K_{st}(x,z) := \left(\frac{1}{\mu} + \delta_{st}\right) \mathrm{x} \cdot \mathrm{z} \;\; s,t = 1,...,T \tag{2.22}$$

问题 2.3.1 的对偶问题如下：

问题 2.3.3

$K_{st}\big(x_{is},x_{jt}\big)$使得对于所有的 $i \in \{1,...,m\}$和 $t \in \{1,...,T\}$，都满足

$$0 \le \alpha_{it} \le C$$

而且，若α_{it}^*是上述问题的解，那么问题 2.3.1 的解为：

$$f_t^*(x) = \sum_{i=1}^{m}\sum_{s=1}^{T} \alpha_{is}^* K_{st}(x_{is},x)$$

因此需要筛选两个参数：（1）C：在标准 SVM 中与训练误差有关；（2）μ：捕捉了任务间的相似性。这两个参数可以通过使用验证集或使用交叉验证来获

得。比如，筛选μ时可以通过任务交叉验证（task-cross-validation）而不是像典型交叉验证中留出部分训练数据的做法，任务交叉验证是预留任务来进行验证。对于μ

的筛选，没有理论方法，所以这是一个开放性问题。

解决优化问题 2.3.3 时，要求解决 Tm 个训练数据的标准 SVM 问题。假设标准 SVM 的训练方法用来训练 O(训练样本数)次，那么要解决问题 2.3.3 需要 O(T^3m^3)次。独立求解每个任务时，仅需要 O(Tm^3)次。

2.3.4 非线性多任务学习

SVM 一个重要的特点是，可以通过使用核函数来估计高度非线性的函数，那么可以根据 SVM 的核函数方法把线性情况推广到非线性情况，而且定理 2.3.3 的情况可以被推广到非线性情况。可以通过简单的使用非线性特征映射来学习函数 F，特征映射为：$\Phi: X \times \{1,...,T\} \to H$，其中 $\mathcal{H}$ 是可分离的希尔伯特空间（特征空间）。和Φ相关的核函数为：$G((x,t),(\mathbf{z},s)) = \langle \Phi((\mathbf{z},s)) \rangle$，其中$\langle \cdot, \cdot \rangle$是 $\mathcal{H}$ 上的内积。一般地，考虑每个任务由不同数量的样本来训练，即，抽样得到的样本为$((x_i,t_i),y_i) \in (X \times \{1,...,T\}) \times \{-1,1\}, \mathrm{i} = 1,...,N$ 且从函数中可得到系数β_i，这个函数为：

$$F(\boldsymbol{x},t) = \sum_{i=1}^{N} \beta_i G((\boldsymbol{x},t),(x_i,t_i)) \qquad (2.23)$$

系数β_i可通过求解有核函数 G 标准 SVM 的对偶问题来得到，那么可得到问题 2.2.4：

问题 2.2.4

$$\max_{\beta_i} \left\{ \sum_{i=1}^{N} \beta_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \beta_i y_i \beta_j y_j G((x_i,t_i),(x_j,t_j)) \right\} \qquad (2.24)$$

使得对每个 i 满足约束$\beta_i \in [0,C]$。

若设定 $N = mT$，那么问题 2.3.3 可以降为问题 2.3.4，对 $i \in \{1,...,N\}$可定义（mod 表示求余）：

$$x_i = x_{(i \bmod T)(i \bmod m)}$$

$$y_i = y_{(i \bmod T)(i \bmod m)}$$

且使用核函数 $G((x_i,t_i),(x_j,t_j)) = K_{t_i t_j}(x_i,x_j)$，其中由式（2.22）可以给出非线性核函数 K 来代替点积 $\mathrm{x} \cdot \mathrm{z}$ 来用于标准 SVM。

2.4 关系迁移——Mapping and Revising Markov Logic Networks

MLNs 结合了一阶 logic 表达和概率复杂度。MLN 有两个方面：结构，或者

一阶从句；权重。当权重学习比较快时，结构学习计算会很密集。因此，在这里关注 MLN 的结构学习，因为这对迁移学习有益。

在向一个新域迁移 MLN 时，有两个子任务：谓词(predicate)映射和理论提炼。一般地，用来描述源域和目标域数据的谓词集合可能是部分或全部不同的，因此第一个迁移任务是建立从源域到目标域的谓词的一个映射。创建了映射后，源域的句子可以被翻译到目标域。但是这些句子翻译的可能不准确，那么被翻译的句子需要再修正、增补和重赋权来得到适合目标域的模型。

2.4.1 背景知识介绍

2.4.1.1 一般术语和符号

一阶逻辑四种不同的符号：常量、变量、谓词和函数。常量描述了域中的信息，可以有不同形式；变量作为占位符用来量化数据；谓词表现了域的关系，比如 WorkedFor；函数体现了数据集的函数关系。假设域中没有函数，用小写字母的字符串来表示常量，用单独的大写字母来表示变量，用大写字母的字符串来表示谓词。一般来说，项可以是常量、变量或函数的组合。基本的项(term)不包含变量。谓词作为项中的原子(atom)，正的字面量(literal)是原子，负的字面量是负原子，在这里称只包含基本项的基字面量为 gliteral（即只包含常量、谓词、函数的字面量）。字面量的公式由逻辑运算符（如：与、或），字句由分离的字面量组成。$\neg p \vee q$ 在逻辑上等价于 $p \Longrightarrow q$，这样就是把句子重写为潜在含义，但是不改变原意。比如，$\mathrm{Director(A)} \vee \mathrm{Actor(A)}$可以写成两种形式：（1）$\neg\mathrm{Actor(A)} \Longrightarrow \mathrm{Director(A)}$，或（2）$\neg\mathrm{Director(A)} \Longrightarrow \mathrm{Actor(A)}$。通常称字面量的左部分为前提（antecedents），右部分为结论（conclusion）。

2.4.1.2 马尔科夫逻辑网络（Markov Logic Networks）

MLN 由权重公式组成，它提供了一种通过创建一个不是所有公式都满足条件的场景来减轻一阶逻辑（first-order logic）。假设公式是字句的格式，图 1 中给出了域的 MLN 的例子，这是一个关于拍摄电影的成员例子。

1.5	$\mathrm{Director(A)} \vee \mathrm{Actor(A)}$
0.1	$\mathrm{MovieMember(M,A)} \vee \neg\mathrm{Director(A)}$
1.3	$\neg\mathrm{WorkerFor(A,B)} \vee \neg\mathrm{Director(A)}$
0.5	$\mathrm{Director(A)} \vee \neg\mathrm{MovieMember(M,A)} \vee \neg\mathrm{WorkerFor(B,A)}$

图 1：MLN 的一个例子

令 **X** 为描述世界的主题的集合（即，所有的 gliteral 由域中的谓词和常量组合而成），$\mathcal{F}$ 为 MLN 中句子的集合，w_i是句子$f_i \in \mathcal{F}$ 对应的权重，$\mathcal{G}_{f_i}$为有常量的句子f_i的可能的基（groundings）集合，Z 是正规化分割函数（normalizing partition

function）。特定的真值指派（truth assignment）x 到 **X** 的概率 $P(\boldsymbol{X}=\mathrm{x})=(1/Z)\exp\left(\sum_{f_i\in\mathcal{F}} w_i \sum_{g\in \mathcal{G}_{f_i}} g(\mathrm{x})\right)$，不同条件下的 $g(\mathrm{x})$的值为 1 或 0。因此在真实给定当前指派（assignment）**X** 后，$\sum_{g\in \mathcal{G}_{f_i}} g(\mathrm{x})$是$f_i$的基（groundings）的数量。

为了对给定的 MLN——L 进行推断，需要找到 L 对应的马尔科夫网络。这包含节点和特征，节点对应 L 中的谓词的每个可能的 gliteral，特征对应 L 中的句子的每个基（grounding）。基句的节点是成组的。为了计算存疑的 gliteral 为真的概率，在马尔科夫网络中 进行吉布斯（Gibbs）抽样，吉布斯抽样是通过对每个存疑的 gliteral 分配一个真值，给定 gliteral $\boldsymbol{X}$ 的马尔科夫覆盖层MB_X的真值后（节点在基句中），对 gliteral $\boldsymbol{X}$ 的值进行多轮重采样，表示为下面的公式：

$$P(X=x|\mathrm{MB}_X=m)=\frac{e^{S_X(x,m)}}{e^{S_X(0,m)}+e^{S_X(1,m)}} \qquad (2.25)$$

在这里，$S_X(x,m)=\sum_{g\in \mathcal{G}_{f_i}} w_i g_i(X=x,\mathrm{MB}_X=m)$，其中$\mathcal{G}_X$是句子 X 的基的集合，m 是MB_X的当前真值指派。

通常称当前最好的 MLN 结构学习算法为 KD，KD 要么使用定向搜索、要么使用最短有限搜索。KD 定向搜索的几轮中，每轮中把发现的最好的句子加入 MLN。使用加权伪对数似然估计（weighted pseudo log-likelihood measure,WPLL）来评估句子，给定句子的马尔科夫覆盖层（Markov blanket）后，把每个节点的对数似然加总，给句子加权来保证 gliterals 的谓词不会控制结果。每轮定向搜索从所有 single-vliteral 句子开始，对初始句进行所有可能的修正来产生候补，保持最好的定向尺寸（beamSize）的句子，对这些句子尽可能修正后来产生新的候补，保持最好的定向尺寸，然后继续这个过程，直到候补（candidates）停止提升 WPLL。此时，得到的最好的候补被加入 MLN，然后开始新一轮的定向搜索。在计算 WPLL 之前，需要在给定了结构之后再去学习权重。权重训练作为一个优化过程来有效实施，因此不需要迁移权重，因为 KD 要么从分支开始学习，要么从给定的 MLN 开始学习，它是一个修正算法，但是 KD 没有谓词映射能力。

2.4.1.3 关系寻路（Relational Pathfinding）

关系寻路（Relational Pathfinding，RPF）是数据驱动的方法，对用以克服平衡和局部最大情况的方法来进行规范化。通常在修正阶段中使用 RPF 来发现目标域中特定的关系。RPF 将相关域视为一个图 G，在这个图中，常量是至高点，如果两个常量在真 gliteral 中共同出现的话，那么这两个常量通过边（edge）联系起来。数据中既有真 gliteral，也有假 gliteral。直观上来看，RPF 构建句子，这些句子的结论（conclusion）是特定的真 gliteral，且前提（antecedents）由在关系图 G 中定义了路径的 gliterals 组成。这些句子之后会被变量化。RPF 在两个通过边联系的常量c_1、c_2中，对 G 搜索长度至少为 2 的交替路径。如果这条路径被发现了的话，它会被转化为如下的句子：前提被创建为谓词的连接词，这些谓词标记了路径的边；结论是字面量，这个字面量标记了连接常量c_1、c_2的边。使用爬山法（hil-climbing search）通过对句子的前提加入一元谓词来提升句子。

比如，假定图 2 列出了域中的所有真实实例，图 3 展示了这个域中的关系图。

粗边构建了一条在 jack 和 jill 间可替代的路径，构建了句子 $\text{MovieMember(T,A)} \wedge \text{MovieMember(T,B)} \Longrightarrow \text{WorkedFor(A,B)}$。爬山法可能会对前提添加 Actor(A)和 Director(B)。

Director(jack) Actor(jill)
MovieMember(movie1,jack) MovieMember(movie1,jill)
MovieMember(movie2,jill) WorkderFor(jill,jack)

图 2 关系数据范例

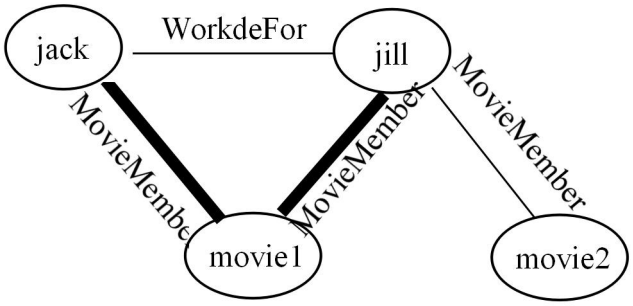

图 3 关系图范例

2.4.2 MLN 结构迁移

在 MLN 结构迁移中，首先将源域的 MLN 映射到到目标域，之后修正这些句子。

2.4.2.1 谓词映射（Predicate Mapping）

这一步的目标是发现最优途径来将源域的 MLN 映射到目标域 MLN，映射的质量好坏通过映射 MLN 在目标数据上的表现来衡量，即计算 WPLL 得分来衡量。映射的方法一般有两种：（1）对每个源域的谓词建立其到目标谓词的映射，之后使用这个映射来翻译整个源域 MLN，这称为全局映射（global mapping）；（2）局部映射（local mapping）是通过独立于其他句子的映射结构，仅仅对每个源句中的谓词构造映射，从而找到每个源句的最佳映射。

大多数情况中，寻找全局最优映射是不可能的，因为计算量在谓词上是指数增长的，一般都是寻找局部最优映射，因为局部最优映射的计算量远远小于全局最优映射。因此在这个算法中，集中于寻找局部最优的谓词映射。

为了寻找一个句子的最优谓词映射，要对所有合法映射（legal mappings）空间进行遍历搜索。下面给出一些名词解释。

（1）合法映射（legal mapping）

如果在给定句子后，每个源谓词要么映射到相容（compatible）的目标谓词，要么映射到空的谓词（空谓词：从句子中抹去源谓词的所有字面量(literals)），那么这个映射就是合法的。

（2）相容（compatible）

如果两个谓词有相同的元数（函数的参数数量），而且它们的参数形式和当前形式约束是相容的（compatible），那么就说这两个谓词是相容的。

数学中，在形式化的逻辑系统中，其相容性是指其中没有矛盾，或更精确的说，不存在一个命题 P，P 和非 P 都可以在这个系统中证明。

在微分方程的数值计算中，若当剖分的各个区间长度趋向于零时，所构造的差分方程能与原微分方程充分接近，这个事实称为差分格式的相容性。

对于任意合法映射，在源域中的形式被映射到目标域中至多一个对应的形式，而且形式约束（type constrains）是要求这些形式的映射在句子的所有为此上都一致。比如，如果当前的形式约束是空的，那么源谓词 Publication(title,person)看成与目标谓词 Gender(person,gend)是相容的，形式约束 title → person 和 person → gend 会被加到当前的形式约束中。

在现在的句子里，所有后来的谓词映射都要符合已有的约束。比如在已有约束下，源谓词 SamePerson(person ,person)和目标谓词 SameGender(gend,gend)是相容的，但是和目标谓词 SamePerson(person,person)是不相容的。

在构造了法定映射后，通过计算只由翻译了的句子组成的 MLN 的 WPLL，来对翻译的句子进行评估。一个谓词映射翻译的句子中，WPLL 得分最高的就是这个句子的最优的谓词映射。重复这个过程来对每个源句寻找最优谓词映射。下面的算法是对源句寻找法定映射，这个流程的循环版本用来寻找源句的所有法定映射。图 4 展示了这个算法在给定源句后找到的最优谓词映射。

Algorithm	Find a legal mapping for a source clause 寻找源句子的法定映射

```
1： procedure LEGAL_MAPPING(srcClause,tarPreds) [法定映射(源句,目标句)]
2：     predsMapping← ∅   谓词映射为空集
3：     typeConstraints← ∅ 形式约束为空集
4：     repeat
5：         选取未被映射的源谓词 srcPred
6：         for 每个未被映射的目标谓词 tarPred do
7：             if isCompatible(srcPred , tarPred) then
8：                 把这个映射加入 predsMapping
9：                 更新 typeConstraints
10：                退出 for 循环
11：            end if
12：        end for
13：    until srcClause 中所有的谓词都被映射到 target 中
14：    return predsMapping
15： end procedure
```

Source clause:

$$\text{Publication}(T,A) \wedge \text{Publication}(T,B) \wedge \text{Professor}(A) \wedge \text{Student}(B) \wedge \neg\text{SamePerson}(A,B) \Longrightarrow \text{AdvisedBy}(B,A)$$

Best mapping:

$$\text{Publication}(\text{title},\text{person}) \longrightarrow \text{MovieMember}(\text{movie},\text{person})$$
$$\text{Professor}(\text{person}) \longrightarrow \text{Director}(\text{person})$$
$$\text{Student}(\text{person}) \longrightarrow \text{Actor}(\text{person})$$
$$\text{SamePerson}(\text{person},\text{person}) \longrightarrow \text{SamePerson}(\text{person},\text{person})$$
$$\text{AdvisedBy}(\text{person},\text{person}) \longrightarrow \text{WorkerFor}(\text{person},\text{person})$$

图 4 谓词映射算法的输出

2.4.2.2 修正映射结构

下面描述映射结构如何被修正从而提升对数据的匹配度。修正算法的框架有三个步骤，和修正了一阶理论的 FORTE(Richards&Mooney 1995)相似。

Step 1：自诊断（self-diagnosis）

这一步集中于在 MLN 的错误部分寻找修正 。算法检查源 MLN 后，判断每个句子是否需要被缩减、延长或保留原样。对于每个句子 *C*，将 *C* 视为可能的结果，字面量中的一个会放在可能结果的右边作为结论（conclusion），剩下的字面量作为前提（antecedents）。当句子的前提成立时，推出句子的结论，然后这个句子就被激活。因此，如果一个句子得出错误结论，就要延长这个句子，因为对前提补充更多字面量或条件会让这个前提难以成立，从而防止这个句子被激活。另一方面来说，一些句子可能会得出错误的结论，因为它们的前提中有太多条件以致于它们不被激活。在这个情况中，要对句子进行缩减。

Step 2：结构更新（structure update）

当一些被认为太短的句子进行了延长后，这些句子中太长的会被缩减。

Step 3：新句子的发现（new clause discovery）

使用 RPF（2.4.1.3 中的关系寻路）后，目标域中会发现新句子。

3. 直推式迁移学习方法详述

3.1 样本迁移——KMM 算法、KLIEP 算法

样本权重法的思想：采用合理策略改变源域数据中的训练样本的权重，使其和目标域数据的测试样本相匹配，进而改善目标域数据的学习效果。

由 **1.3.2.1** 基于经验风险最小化（ERM)框架的推导，可知权重为$\frac{P(x_{S_i})}{P(x_{T_i})}$。估计出权重$\frac{P(x_{S_i})}{P(x_{T_i})}$的值，即能使得源域数据与目标域数据相匹配。

3.1.1 KMM 算法

Huang et al.[10]提出核均值匹配(Kernel-mean Matching，KMM)算法，通过匹配源域和目标域在再生核希尔伯特空间(reproducing-kernel Hilbert space，RKHS)上的均值来直接得到$\frac{P(x_{S_i})}{P(x_{T_i})}$，记 $\beta = \frac{x_S}{x_T}, \beta_i = \frac{x_{S_i}}{x_{T_i}}$。

第一步：将数据映射到一个再生核希尔伯特空间（RKHS）计算两个领域之间的核均值。

第二步：通过减小 RKHS 的距离来给源域样本实例赋予权值。

KMM 可以写为下面的二次程序(quadratic programming，QP)优化问题：

$$\min_{\beta} \frac{1}{2}\beta^T K\beta - \mathcal{K}^T\beta \qquad (3.1)$$

$$s.t.\ \ \beta_i \in [0,B] \text{ 且 } |\sum_{i=1}^{n_S}\beta_i - n_S| \le n_S\epsilon$$

其中 $K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix}$，$K_{ij} = k(x_i,x_j)$。$K_{S,S}$和$K_{T,T}$是源域和目标域的核矩阵。

当$x_{T_j} \in X_T$时，$\mathcal{K}_i = \frac{n_S}{n_T}\sum_{j=1}^{n_T} k(x_i,x_{T_j})$，其中$x_i \in X_S \cup X_T$。

可证明$\beta_i = \frac{P(x_{S_i})}{P(x_{T_i})}$[10]。使用KMM的优点在于它可以避免表达$P(x_{S_i})$和$P(x_{T_i})$的密度估计，当数据集很小时这比较困难。

算法流程如下：

Algorithm	KMM 算法
输入	源域有标签数据集 $T = \{(x_l^{(1)},y^{(1)}),(x_l^{(2)},y^{(2)}),\ldots,(x_l^{(m)},y^{(m)})\}$

目标域无标签数据 U={$x_u^{(1)}$,$x_u^{(2)}$,...,$x_u^{(k)}$}

输出 预测函数 f：X→Y

1： 利用源域有标签数据集 T，求解优化问题(1)得到权重 β

$$\min_{\beta}\frac{1}{2}\beta^T K\beta-\mathcal{K}^T\beta$$

$$s.t.\ \ \beta_i\in[0,B]\ 且\ \left|\sum_{i=1}^{n_S}\beta_i-n_S\right|\le n_S\epsilon \qquad (3.1)$$

2： 对源域有标签数据集更新权重后，得到新的有标签数据集

$$\hat{T}=\left\{\left(\beta_i(x_l^{(i)}),y^{(i)}\right)\right\}_{i=1}^{m}$$

3： 在更新权重后的有标签数据集$\hat{T}$上应用学习算法来得到预测函数 f：X→Y

3.1.2 KLIEP 算法

估计权重$\frac{P(x_{S_i})}{P(x_{T_i})}$的方法有很多，除了上述的 KMM 算法之外，Sugiyama 等人提出一种可以直接估计源域权重的 KLIEP（Kullback-Leibler Importance Estimation Procedure)算法，基于 KL 交叉熵的距离去辅助数据样本的权重，且这个方法能进行交叉验证来选择模型。

第一步：基于源域数据估计权重。

第二步：对根据权重对源域数据进行重新赋值后，在重新赋值后的数据上建立训练模型。具体的优化问题为：

3.1.2.1 问题描述

记源域有标签数据集为$T=\left\{x_s^{(i)},y_s^{(i)}\right\}_{i=1}^{n_s}$，其中$n_s$为源域有标签样本的个数，边际分布概率为$P_S(x)$。目标域无标签数据$U=\left\{x_T^{(j)}\right\}_{j=1}^{n_T}$，其中$n_T$为目标域无标签样本的个数，边际分布概率为$P_T(x)$，权重

$$w(x)=\frac{P_T(x)}{P_S(x)}$$

我们的目标是：采用一个线性模型来估计权重 w(x)，具体表示如（3.2）

$$\hat{w}(x)=\sum_{l=1}^{b}\alpha_l\varphi_l(x) \tag{3.2}$$

其中 $\{\alpha_l(x)\}_{l=1}^{b}$ 是从样本数据中学习而得的参数，$\{\varphi_l(x)\}_{l=1}^{b}$ 是一个基础函数（可取基础函数为核函数），$\varphi_l(x)\geq 0$（$l=1,2,\ldots,b$）。我们的任务是根据公式（3.2）计算出权重。

注意到 b、$\{\varphi_l(x)\}_{l=1}^{b}$ 的取值均依赖于源域数据 $\{x_s^{(i)}\}_{i=1}^{n_s}$ 和目标域数据 $\{x_T^{(j)}\}_{j=1}^{n_T}$。

目标域数据的边际分布的估计可用权重的估计来表示，表达式如（3.3）：

$$\hat{p}_D(x)=\hat{w}(x)\,p_S(x) \tag{3.3}$$

首先，我们通过使得 KL 距离最小化来确定参数 $\{\alpha_l(x)\}_{l=1}^{b}$，从 $P_T(x)$ 到 $\hat{p}_T(x)$ 的 KL 距离为：

$$\begin{aligned}\mathrm{KL}[P_T(x)\,\|\,\hat{p}_T(x)] &= \int p_T(x)\log\frac{p_T(x)}{\hat{w}(x)p_S(x)}dx \\ &= \int p_T(x)\log\frac{p_T(x)}{p_S(x)}dx-\int p_T(x)\log\hat{w}(x)dx\end{aligned} \tag{3.4}$$

注意到（3.4）中只有后一个式子与 $\{\alpha_l(x)\}_{l=1}^{b}$ 有关，将后一个式子记为 J,即有

$$\begin{aligned}J&=\int p_T(x)\log\hat{w}(x)dx \\ &\approx\frac{1}{n_T}\sum_{j=1}^{n_T}\log\hat{w}(x_T^{(j)}) \\ &=\frac{1}{n_T}\sum_{j=1}^{n_T}\log\Big(\sum_{l=1}^{b}\alpha_l\varphi_l(x_T^{(j)})\Big)\end{aligned} \tag{3.5}$$

要使得 KL 距离最小化，即等价于使得 J 最大化。

其次，在求解过程中应满足一定的约束条件：Ⅰ.权重 $\hat{w}(x)$ 为非负，即要求 $\alpha_l\geq 0$（$l=1,2,\ldots,b$）；Ⅱ.$\hat{p}_T(x)$ 作为目标域分布函数的预测，其总和应为 1，即

$$1=\int\hat{p}_T(x)dx=\int\hat{w}(x)\,p_S(x)dx$$

$$\approx \frac{1}{n_s}\sum_{i=1}^{n_s}\hat{w}(x_S^{(i)}) = \frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{l=1}^{b}\alpha_l\varphi_l(x_S^{(i)}) \qquad (3.6)$$

综合所述，求解权重估计 $\hat{w}(x)$ 的问题可等价于求解如下的优化问题：

$$\max_{\{\alpha_l\}_{l=1}^{b}} imize\left[\sum_{j=1}^{n_s}\log(\sum_{l=1}^{b}\alpha_l\varphi_l(x_T^{(j)}))\right]$$

$$\text{Subject to } \sum_{i=1}^{n_S}\sum_{l=1}^{b}\alpha_l\varphi_l(x_{n_s}^{(i)}) = n_S\text{，且}\alpha_1,\alpha_2,\cdots\alpha_b \geq 0 \qquad (3.7)$$

3.1.2.2 算法流程

Algorithm	KLIEP 算法
输入	基础函数 $m=\{\varphi_l(x)\}_{l=1}^{b}$，源域数据$\left\{x_s^{(i)}\right\}_{i=1}^{n}$，目标域数据$\left\{x_T^{(j)}\right\}_{j=1}^{n_T}$
输出	权重估计值 $\hat{w}(x)$
	$A_{j,l} \leftarrow \varphi_l(x_T^{(j)});$
	$b_l \leftarrow \frac{1}{n_s}\sum_{i=1}^{n_s}\varphi_l(x_T^{(i)});$
初始化	$\alpha(>0)$，$\varepsilon(0<\varepsilon<<1)$
重复直到收敛	$\alpha \leftarrow \alpha+\varepsilon A^T(1./A\alpha);$ $\alpha \leftarrow \alpha+(1-b^T\alpha)b/(b^Tb);$ $\alpha \leftarrow \max(0,\alpha);$ $\alpha \leftarrow \alpha/(b^T\alpha);$
结束	$\hat{w}(x) \leftarrow \sum_{l=1}^{b}\alpha_l\varphi_l(x)$

注：’./’表示按元素划分，T 表示转置。

3.2 特征迁移——Structural Correspondence Learning

识别学习(Discriminative learning methods)在自然语言处理(natural language processing,NLP)中无处不在。近十多年来，标签识别(Discriminative taggers)占据识别学习的很大一块，此外，诸如语音识别(Speech recognizers)、自动翻译(automatic translators)等端到端(end-to-end systems) 系统，越来越多地使用复杂的识别模型。

上述识别学习在新数据和训练数据来自同一分布时有很好的效果。但是，在很多时候，我们可能面对的情况是源域中有大量的有标签数据作为训练数据，而我们要处理的是目标域的无标签数据，其中源域和目标域的分布不同。这种情况下，我们需要采取措施调整模型，以使得源域数据能够用于目标域(Roark 和 Bacchiani，2003)。

调整模型的重点在于利用来自源域和目标域的无标签数据，找到一个对于源域和目标域都有意义的共同特征表示。我们推测，在源域数据上使用这个共同特征表示来进行识别学习训练，能更好地适用于目标域。寻找这个共同特征表示的方法我们称为结构一致学习(structural correspondence learning,SCL)算法，SCL 算法采用轴特征表示不同领域中特征间的对应关系，轴特征指在不同领域的识别学习中作用相同的特征，可以解释为不同领域之间的共享知识的存在方法。轴特征在不同领域的识别学习中作用相同。源域和目标域均将知识映射到轴特征上，若不同域中非轴特征都与同一轴特征相关，则假设这些非轴特征也是相关的，并在学习模型时被赋予相同的权重。

3.2.1 结构一致学习(structural correspondence learning,SCL)

SCL 算法是一种通用技术，可以应用于任何任务的基于特征的分类。

3.2.1.1 算法流程

源域和目标域中有大量的无标签数据，仅在源域中存在有标签的训练数据。我们将这部分有标签训练数据记为有监督任务。

大多数直推式迁移学习的特征迁移方法是在无监督学习框架下的。Blitzer *et al.*[11]提出了一种结构一致学习(structural correspondence learning,SCL)算法，它使用目标域中的无标签数据来提炼能减小域间差异的相关特征，即从目标域的无标签数据中提取一些可以减小不同域数据之间的差异。

Step 1 ：在两个域的无标签数据上定义轴特征(pivot feature)的数据集(轴特征的数量为 *m*)。

我们可用这些轴特征来学习得到源域和目标域的原始映射θ，从而将源域和目标域的特征空间共享，转化为一个低维实值特征空间。在有监督训练

任务中，我们在源域的原始特征空间和转化后的特征空间上都进行训练。在有监督测试任务中，我们在目标域的原始特征空间和转化后的特征空间上都进行测试。如果我们能学习到一个很好的映射θ，那么我们在源域上得到的分类器就能够很有效地使用于目标域上。

Step 2 ：把轴特征移出数据，把每个轴特征当作一个新的标签向量，建立 *m* 个二元分类问题。假定每个问题都通过线性预测器求解，如下式：

$$f_l(x) = \text{sgn}(\hat{w}_l \cdot x), \qquad l = 1 \ldots m$$

SCL 可以得到一个参数矩阵$W = [\hat{w}_1 | \ldots | \hat{w}_m]$。

权重向量$\hat{w}_l$编码了轴特征和非轴特征之间的协方差。

Step 3 ：对参数矩阵 *W* 进行奇异值分解（SVD）。令 $\boldsymbol{W = UDV^T}$，那么$\boldsymbol{\theta = U^T_{[1:h,:]}}$（*h* 是公共特征的数量）的行是 *W* 的上左奇异向量(top left singular vectors)。

θ的行是主要的轴预测器，它尽可能地捕捉了 h 维的轴预测空间的方差。此外，θ是原始特征空间到R^h(h 维轴预测空间)的投影，也就是说，θ_x是所需的(低维)共享特征表示的映射。

Step 4 ：对扩张的特征向量使用标准判别算法来建立模型。扩张的特征向量包含所有的原始特征$\boldsymbol{x_i}$加上新的公共特征$\boldsymbol{\theta_{x_i}}$。

若轴特征设定的较好，那么得到的映射编码了不同域里特征的一致性从而我们在源域的新特征上训练得到的分类器能很好地应用在目标域上。

3.2.1.2 算法描述

记源域有标签数据集为$T = \left\{x_s^{(i)}, y_s^{(i)}\right\}_{i=1}^{n_s}$，其中$n_s$为源域有标签样本的个数。源域和目标域的无标签数据记为$\{x_j\}$。

算法描述如下：

Algorithm	结构一致学习（SCL）算法
输入	源域标签数据集$T = \left\{x_s^{(i)}, y_s^{(i)}\right\}_{i=1}^{n_s}$，源域和目标域的非标签数据集$\{x_j\}$
输出	预测函数 f：X→Y
1：	选择 m 个枢纽点（pivot）特征，构造 m 个分类问题, $p_l(x), l = 1, \ldots, m$

2： For $l=1,\ldots,m$

$$\hat{w}_l = \arg\min_{w}(\sum_{j} L(w\cdot x_j, p_l(x_j)) + \lambda \| w \|^2)$$

end

3： 得到一个参数矩阵$W=[\hat{w}_1|\ldots|\hat{w}_m]$，对参数矩阵$W$进行奇异值分解（SVD），

$[UDV^T]=SVD(W)$

4： 得到线性映射函数$\theta = U^T_{[1:h,:]}$，（h是公共特征的数量），即得到源域标签数据新的特征表示，新的共享特征为θ_{x_i}。

5： 得到扩大的特征向量数据集$T_1=\left\{\left(\begin{bmatrix} x_s^{(i)} \\ \theta_{x_i} \end{bmatrix}, y_s^{(i)}\right)_{i=1}^{n_s}\right\}$其中包括原始的特征$x_i$、新的共享特征$\theta_{x_i}$

6： 调用基本分类算法，根据扩大的特征向量表示下的源域标签数据集T_1，得到分类器。

7： 将此分类器用于预测目标域非标签数据X_{tu}的标签。

注：这里 L（p，y）是一个用于二进制分类的实值损失函数。

4.无监督迁移学习方法详述

4.1 特征迁移——Self-taught Clustering

Dai *et al.*[12]提出了自学习聚类(Self-taught Clustering,STC)算法，它属于无监督迁移学习，这种方法旨在通过源域中大量无标签数据来将目标域中的一小部分无标签数据聚类。STC 试图得到域间的公共特征空间，以便在目标域中聚类。STC 的目标函数如下：

$J(\widetilde{X_T},\widetilde{X_S},\widetilde{Z}) = I(X_T,Z) - I(\widetilde{X_T},\widetilde{Z}) + \lambda[I(X_S,Z) - I(\widetilde{X_S},\widetilde{Z})]$

其中，X_S和X_T是源域和目标域的数据。Z是X_S和X_T的公共特征空间，$I(\cdot,\cdot)$是两个随机变量的共同信息。假设存在三个聚类函数C_{X_T}：$X_T \to \widetilde{X_T}$，C_{X_S}：$X_S \to \widetilde{X_S}$，

$C_Z: Z \to \widetilde{Z}$，其中$\widetilde{X_T}$、$\widetilde{X_S}$、$\widetilde{Z}$是X_T、X_S、Z 的对应聚类。STC 的目标是通过求解 $\arg\min_{\widetilde{X_T},\widetilde{X_S},\widetilde{Z}} J(\widetilde{X_T},\widetilde{X_S},\widetilde{Z})$ 得到$\widetilde{X_T}$。

4.1.1 问题描述

为了清楚起见，我们首先定义自学习聚类任务。记源域无标签数据集 $X_S=\{x_s^{(i)}\}_{i=1}^{n_s}$，目标域的非标签数据集 $X_T=\{x_T^{(j)}\}_{j=1}^{n_T}$，$X_S$ 和 X_T 共同的特征空间为 Z，$Z=\{z_1,\ldots,z_K\}$，X_S、X_T、Z 均是离散的随机变量。

设 P(X_T,Z)是关于 X_T、Z 的联合概率分布，设 P(X_S,Z)是关于 X_S、Z 的联合概率分布。一般来说，P(X_T,Z)、P(X_S,Z)可分别看作两个 $n_T\times K$ 和 $n_S\times K$ 矩阵，且可从样本数据中估计而得。

我们使用 $C_{X_T}: X_T \mapsto \widetilde{X}_T$，$C_{X_S}: X_S \mapsto \widetilde{X}_S$ 和 $C_Z: Z \mapsto \widetilde{Z}$ 来表示三个聚类函数，它们分别将各自的变量值映射到相应的聚类集群。为了简洁起见，我们用 $\widetilde{x}_T$、$\widetilde{x}_s$ 和 $\widetilde{z}$ 来表示 $C_{X_T}(X_T)$、$C_{X_S}(X_S)$ 和 $C_Z(Z)$。

4.1.2 算法流程

我们扩展了信息理论的协同聚类（Dhillon 等人，2003）来模拟我们自学的聚类算法。

在信息理论的协同聚类中，目标函数可表示为：

$$I(X_T,Z)-I(\widetilde{X}_T,\widetilde{Z}) \tag{4.1}$$

其中 $I(\cdot,\cdot)$是两个随机变量的共同信息,且

$$I(X_T,Z)=\sum_{x_T\in X_T}\sum_{z\in Z}p(x_T,z)\log\frac{p(x_T,z)}{p(x_T)p(z)} \tag{4.2}$$

$$I(\widetilde{X}_T,\widetilde{Z})=\sum_{\widetilde{x}_T\in \widetilde{X}_T}\sum_{\widetilde{z}\in \widetilde{Z}}p(\widetilde{x}_T,\widetilde{z})\log\frac{p(\widetilde{x}_T,\widetilde{z})}{p(\widetilde{x}_T)p(\widetilde{z})} \tag{4.3}$$

$$p(\widetilde{x}_T,\widetilde{z})=\sum_{x_T\in \widetilde{x}_T}\sum_{z\in \widetilde{z}}p(x_T,z) \tag{4.4}$$

在我们使用的自学聚类算法中，目标函数为：

$$J=I(X_T,Z)-I(\widetilde{X}_T,\widetilde{Z})+\lambda[I(X_S,Z)-I(\widetilde{X}_S,\widetilde{Z})] \tag{4.5}$$

其中，λ 是用来平衡目标数据和辅助数据的影响的平衡参数，$I(X_T,Z)-I(\widetilde{X}_T,\widetilde{Z})$

和 $I(X_S,Z)-I(\widetilde{X}_S,\widetilde{Z})$ 虽然分别是两个聚类的目标函数，但它们拥有相同的特征聚类 $\widetilde{Z}$,这是目标数据和辅助数据之间迁移知识的桥梁。

STC 的目标是使目标函数最小化来得到$\widetilde{X_T}$:

定义 4.1.1

记 $\tilde{p}(X_T,Z)$ 为 X_T 、Z 对应于聚类函数 (C_{X_T},C_Z) 的联合分布，记 $\tilde{q}(X_S,Z)$ 为 X_S 、Z 对应于聚类函数 (C_{X_S},C_Z) 的联合分布。

$$\tilde{p}(x_T,z)=p(\tilde{x}_T,\tilde{z})\frac{p(x_T)}{p(\tilde{x}_T)}\frac{p(z)}{p(\tilde{z})} \tag{4.6}$$

其中， $x_T\in\tilde{x}_T$， $z\in\tilde{z}$

同理，有

$$\tilde{q}(x_s,z)=q(\tilde{x}_s,\tilde{z})\frac{q(x_s)}{q(\tilde{x}_s)}\frac{q(z)}{q(\tilde{z})} \tag{4.7}$$

其中， $x_S\in\tilde{x}_S$， $z\in\tilde{z}$

引理 4.1.2

当聚类函数 C_{X_T} 、 C_{X_S} 、 C_Z 固定时，公式(4.5)的目标函数可写作：

$$\begin{aligned}J&=I(X_T;Z)-I(\widetilde{X}_T;\widetilde{Z})+\lambda[I(X_S;Z)-I(\widetilde{X}_S;\widetilde{Z})]\\&=D(p(X_T,Z)\|\tilde{p}(X_T,Z))+\lambda D(p(X_S,Z)\|\tilde{p}(X_S,Z))\end{aligned} \tag{4.8}$$

其中， $D(\cdot\|\cdot)$ 代表两个联合分布的 KL 距离， $D(p\|q)=\sum_x p(x)\log\frac{p(x)}{q(x)}$ 。

引理 4.1.3

$$\begin{aligned}&D(p(X_T,Z)\|\tilde{p}(X_T,Z))\\&=\sum_{\tilde{x}_T\in\widetilde{X}_T}\sum_{x_T\in\tilde{x}_T}p(x_T)D(p(Z|x_T)\|\tilde{p}(Z|\tilde{x}_T)) \qquad (4.9)\\&=\sum_{\tilde{z}\in\widetilde{Z}}\sum_{z\in\tilde{z}}p(z)D(p(X_T|z)\|\tilde{p}(X_T|\tilde{z})) \qquad (4.10)\end{aligned}$$

根据引理 4.1.3 和公式(4.9)，我们可以看到通过最小化单一 x_T 的

$D(p(Z|x_T)\|\tilde{p}(Z|\tilde{x}_T))$可以减小$D(p(X_T,Z)\|\tilde{p}(X_T,Z))$的值，从而可以减小(4.8)的值。因此，若我们不断地迭代选择对于每个x_T的最合适的聚类$\tilde{x}_T$，以使得$D(p(Z|x_T)\|\tilde{p}(Z|\tilde{x}_T))$最小化，目标函数也会单调最小化，即

$$C_{X_T}(x_T)=\arg\min_{\tilde{x}_T\in\tilde{X}_T} D(p(Z|x_T)\|\tilde{p}(Z|\tilde{x}_T)) \tag{4.11}$$

同样的，有

$$C_{X_S}(x_S)=\arg\min_{\tilde{x}_S\in\tilde{X}_S} D(q(Z|x_S)\|\tilde{q}(Z|\tilde{x}_S)) \tag{4.12}$$

$$C_Z(z)=\arg\min_{\tilde{z}\in\tilde{Z}} p(z)D(p(X_T|z)\|\tilde{p}(X|\tilde{z}))+\lambda\ q(z)D(q(X_S|z)\|\tilde{q}(X_S|\tilde{z})) \tag{4.13}$$

4.1.3 算法描述

基于公式(4.11)、(4.12)、(4.13)，算法描述的伪代码如下：

Algorithm	自学聚类（STC）算法
输入	源域无标签数据集$X_S=\{x_s^{(i)}\}_{i=1}^{n_s}$，目标域的非标签数据集$X_T=\{x_T^{(j)}\}_{j=1}^{n_T}$，$X_S$和$X_T$共同的特征空间Z； 初始的聚类函数$C_{X_S}^{(0)}$、$C_{X_T}^{(0)}$和$C_Z^{(0)}$； 迭代次数T。
输出	$\{x_T\}$的最终的聚类函数$C_{X_T}^{(T)}$
1：	基于X_S、X_T和Z，初始化$p(X_T,Z)$和$q(X_S,Z)$
2：	基于$p(X_T,Z)$、$C_{X_T}^{(0)}$、$C_Z^{(0)}$和公式(4.6)，初始化$\tilde{p}^{(0)}(X_T,Z)$ $$\tilde{p}(x_T,z)=p(\tilde{x}_T,\tilde{z})\frac{p(x_T)}{p(\tilde{x}_T)}\frac{p(z)}{p(\tilde{z})} \tag{4.6}$$
3：	基于$q(X_S,Z)$、$C_{X_S}^{(0)}$、$C_Z^{(0)}$和公式(4.7)，初始化$\tilde{q}^{(0)}(X_S,Z)$ $$\tilde{q}(x_S,z)=q(\tilde{x}_S,\tilde{z})\frac{p(x_S)}{p(\tilde{x}_S)}\frac{p(z)}{p(\tilde{z})} \tag{4.7}$$
4：	对于$t\leftarrow 1,\ldots,T$

5： 基于 p , $\widetilde{p}^{(t-1)}$ 和公式(4.3)，更新 $C_{X_T}^{(t)}(X_T)$

$$C_{X_T}(x_T)=\arg\min_{\widetilde{x}_T\in\widetilde{X}_T} D(p(Z\mid x_T)\,\|\,\widetilde{p}(Z\mid\widetilde{x}_T)) \tag{4.11}$$

6： 基于 q , $\widetilde{q}^{(t-1)}$ 和公式(4.4)，更新 $C_{X_S}^{(t)}(X_S)$

$$C_{X_S}(x_S)=\arg\min_{\widetilde{x}_S\in\widetilde{X}_S} D(q(Z\mid x_S)\,\|\,\widetilde{q}(Z\mid\widetilde{x}_S)) \tag{4.12}$$

7： 基于 p ， q , $\widetilde{p}^{(t-1)}$ ， $\widetilde{q}^{(t-1)}$ 和公式(4.5)，更新 $C_Z^{(t)}(X_Z)$

$$C_Z(z)=\arg\min_{\widetilde{z}\in\widetilde{Z}} p(z)D(p(x_T\mid z)\,\|\,\widetilde{p}(x_T\mid\widetilde{z}))+\lambda q(z)D(q(x_S\mid z)\,\|\,\widetilde{q}(x_s\mid\widetilde{z})) \tag{4.13}$$

8： 基于 $p(X_T,Z)$ 、 $C_{X_T}^{(t)}(X_T)$ 、 $C_Z^{(t)}(X_Z)$ 和公式（1），更新 $\widetilde{p}^{(t)}$

9： 基于 $q(X_S,Z)$ 、 $C_{X_S}^{(t)}(X_S)$ 、 $C_Z^{(t)}(X_Z)$ 和公式（2），更新 $\widetilde{q}^{(t)}$

10： 结束

11： 返回 $C_{X_T}^{(T)}$ 作为 X_T 的最终的聚类函数

定理 4.1.4

在上述伪代码中，第 t 步迭代的目标函数 J 的值为

$$J(C_{X_T}^{(t)},C_{X_S}^{(t)},C_Z^{(t)})=D(p(X,Z)\,\|\,\widetilde{p}^{(t)}(X,Z))+\lambda\ D(q(X_S,Z)\,\|\,\widetilde{q}^{(t)}(X_S,Z)) \tag{4.14}$$

从而有

$$J(C_{X_T}^{(t)},C_{X_S}^{(t)},C_Z^{(t)})\geq J(C_{X_T}^{(t+1)},C_{X_S}^{(t+1)},C_Z^{(t+1)}) \tag{4.15}$$

注意到，虽然 STC 算法能够使目标函数最小化，但它仅仅找到一个局部最优，难以找到全局最优值。

推论 4.1.5

上述的 STC 算法，能在有限的迭代次数中达到收敛。

5. 迁移学习中的常见问题

5.1 迁移边界(Transfer Bounds)

一个重要的问题在于识别迁移学习的指数极限。在文献[14]中，Hassan

Mahmud 和 Ray 分析了迁移学习使用柯尔莫哥洛夫复杂度(Kolmogorov complexity)，其中证明了一些理论边界。尤其是作者使用了条件 Kolmogorov complexity 来估计任务间的相关度以及在贝叶斯框架下对序列迁移学习任务进行了正确数量的信息迁移。

Eaton *et al.*[15]提出了一个心音的基于图像的迁移学习方法，通过使用十进制表达的可迁移性，以图像的方式嵌入已得的源模型来对源任务的关系建模。

5.2 负迁移(Negative Transfer)

负迁移指的是，在源域上学习到的知识，对于目标域上的学习产生负面作用。产生负迁移的原因主要有：

（1）源域和目标域压根不相似，谈何迁移？------【数据问题】

（2）源域和目标域是相似的，但是，迁移学习方法不够好，没找到可迁移的成分。【方法问题】

因此，在实际应用中，找到合理的相似性，并且选择或开发合理的迁移学习方法，能够避免负迁移现象。

随着研究的深入，已经有新的研究成果在逐渐克服负迁移的影响。杨强教授团队 2015 在数据挖掘领悟顶级会议 KDD 上发表了传递迁移学习文章《Transitive transfer learning》，提出了传递迁移学习的思想。传统迁移学习就好比是踩着一块石头过河，传递迁移学习就好比是踩着连续的两块石头。更进一步，杨强教授团队在 2017 年人工智能领域顶级会议 AAAI 上发表了远领域迁移学习的文章《Distant domain transfer learning》，可以用人脸来识别飞机。这就好比是踩着一连串石头过河。这些研究的意义在于，传统迁移学习只有两个领域足够相似才可以完成，而当两个领域不相似时，传递迁移学习却可以利用处于这两个领域之间的若干领域，将知识传递式的完成迁移。这个是很有意义的工作，可以视为解决负迁移的有效思想和方法。可以预见在未来会有更多的应用前景。

参考文献

[1] Pan S J, Yang Q. A Survey on Transfer Learning[J]. IEEE Transactions on Knowledge & Data Engineering, 2010, 22(10):1345-1359.

[2] Dai W, Yang Q, Xue G R, et al. Boosting for transfer learning[J]. 2007.

[3] Schölkopf B, Platt J, Hofmann T. Multi-Task Feature Learning[C]// Conference on Advances in Neural Information Processing Systems. MIT Press, 2006:41-48.

[4] Raina H, Ng A Y. Efficient sparse coding algorithms[J]. Oldbooks.nips.cc.

[5] Lawrence N D, Platt J C. Learning to learn with the informative vector machine[C]// International Conference on Machine Learning. ACM, 2004:65.

[6] Mihalkova L, Huynh T, Mooney R J. Mapping and revising Markov logic networks for transfer learning[C]// National Conference on Artificial Intelligence. AAAI Press, 2007:608-614.

[7] Richardson M, Domingos P. Markov logic networks[J]. Machine Learning, 2006,

62(1-2):107-136.
[8] S. Ramachandran and R. J. Mooney, “Theory refinement of bayesian networks with hidden variables,” in Proceedings of the 14th International Conference on Machine Learning, Madison, Wisconson, USA, July 1998, pp. 454–462.
[9] V. N. Vapnik, Statistical Learning Theory. New York: WileyInterscience, September 1998.
[10] Huang J, Smola A J, Gretton A, et al. Correcting sample selection bias by unlabeled data[C]// International Conference on Neural Information Processing Systems. MIT Press, 2006:601-608.
[11] Daumé Iii H. Frustratingly Easy Domain Adaptation[J]. ACL, 2009.
[12] Dai W, Yang Q, Xue G R, et al. Self-taught clustering[C]// International Conference on Machine Learning. ACM, 2008:200-207.
[13] Wang Z, Song Y, Zhang C. Transferred Dimensionality Reduction[C]// Machine Learning and Knowledge Discovery in Databases, European Conference, Ecml/pkdd 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings. DBLP, 2008:550-565.
[14] Mahmud M M H, Ray S R. Transfer Learning using Kolmogorov Complexity: Basic Theory and Empirical Evaluations[J]. Proc Nips, 2007:1--8.
[15] Eaton E, Desjardins M, Lane T. Modeling Transfer Relationships Between Learning Tasks for Improved Inductive Transfer[C]// European Conference on Machine Learning and Knowledge Discovery in Databases. Springer-Verlag, 2008:317-332.