计 算 机 研 究 与 发 展
Journal of Computer Research and Development

DOI: 10.7544/issn1000-1239.202220232
60(7): 1603−1614, 2023

一种新的半监督归纳迁移学习框架: Co-Transfer

文益民[1,2]　员　喆[1]　余　航[3]

[1]（桂林电子科技大学计算机与信息安全学院　广西桂林　541004）

[2]（广西图像图形与智能处理重点实验室（桂林电子科技大学）广西桂林　541004）

[3]（上海大学计算机工程与科学学院　上海　200444）

（ymwen@guet.edu.cn）

A New Semi-Supervised Inductive Transfer Learning Framework: Co-Transfer

Wen Yimin[1,2], Yuan Zhe[1], and Yu Hang[3]

[1]（*School of Computer Science and Information Safety*, *Guilin University of Electronic Technology*, *Guilin*, *Guangxi* 541004）

[2]（*Guangxi Key Laboratory of Image and Graphic Intelligent Processing*(*Guilin University of Electronic Technology*), *Guilin*, *Guangxi* 541004）

[3]（*School of Computer Engineering and Science*, *Shanghai University*, *Shanghai* 200444）

Abstract　In many practical data mining scenarios, such as network intrusion detection, Twitter spam detection, and computer-aided diagnosis, source domain that is different but related to a target domain is very common. Generally, a large amount of unlabeled data is available in both source domain and target domain, but labeling each of them is difficult, expensive, time-consuming, and sometime unnecessary. Therefore, it is very important and worthwhile to fully explore the labeled and unlabeled data in source domain and target domain to handle classification tasks in target domain. To leverage transfer learning and semi-supervised learning, we propose a new inductive transfer learning framework named Co-Transfer. Co-Transfer first generates three TrAdaBoost classifiers for transfer learning from the original source domain to the original target domain, and meanwhile another three TrAdaBoost classifiers are generated for transfer learning from the original target domain to the original source domain by bootstrapping samples from the original labeled data. In each round of Co-Transfer, each group of TrAdaBoost classifiers is refined by using the carefully labeled data, one part of which is the original labeled samples, the second part is the samples labeled by one group of TrAdaBoost classifiers, and the other samples are labeled by another group of TrAdaBoost classifiers. Finally, the group of TrAdaBoost classifiers learned to transfer from the original source domain to the original target domain to produce the final hypothesis. Experimental results on UCI and text classification task datasets illustrate that Co-Transfer can significantly improve generalization performance by exploring labeled and unlabeled data across different tasks.

Key words　semi-supervised learning; transfer learning; multi-task learning; bi-directional transfer; ensemble learning

摘　要　在许多实际的数据挖掘应用场景,如网络入侵检测、Twitter 垃圾邮件检测、计算机辅助诊断等中,与目标域分布不同但相关的源域普遍存在.一般情况下,在源域和目标域中都有大量未标记样本,对其中

收稿日期: 2022-03-18; **修回日期**: 2022-07-13

基金项目: 国家自然科学基金项目(61866007); 广西自然科学基金项目(2018GXNSFDA138006); 广西重点研发计划项目(桂科 AB21220023); 广西图像图形与智能处理重点实验室项目(GIIP2005, GIIP201505)

The work was supported by the National Natural Science Foundation of China (61866007), the Guangxi Natural Science Foundation (2018GXNSFDA138006), the Key Research and Development Program of Guangxi (桂科 AB21220023), and the Program of Guangxi Key Laboratory of Image and Graphic Intelligent Processing (GIIP2005, GIIP201505).

的每个样本都进行标记是件困难的、昂贵的、耗时的事,有时也没必要.因此,充分挖掘源域和目标域中标记和未标记样本来解决目标域中的分类任务非常重要且有意义.结合归纳迁移学习和半监督学习,提出一种名为Co-Transfer的半监督归纳迁移学习框架.Co-Transfer首先生成3个TrAdaBoost分类器用于实现从原始源域到原始目标域的迁移学习,同时生成另外3个TrAdaBoost分类器用于实现从原始目标域到原始源域的迁移学习.这2组分类器都使用从原始源域和原始目标域的原有标记样本的有放回抽样来训练.在Co-Transfer的每一轮迭代中,每组TrAdaBoost分类器使用新的训练集更新,其中一部分训练样本是原有的标记样本,一部分是由本组TrAdaBoost分类器标记的样本,还有一部分则由另一组TrAdaBoost分类器标记.迭代终止后,把从原始源域到原始目标域的3个TrAdaBoost分类器的集成作为原始目标域分类器.在UCI数据集和文本分类数据集上的实验结果表明,Co-Transfer可以有效地学习源域和目标域的标记和未标记样本从而提升泛化性能.

关键词 半监督学习;迁移学习;多任务学习;双向迁移;集成学习

中图法分类号 TP391

迁移学习已被广泛应用于将知识从源域迁移到相关的目标域的任务[1-2].根据Pan等人[1]的工作,迁移学习可分为3类:归纳迁移学习、直推式迁移学习和无监督迁移学习.有着与前面这3种迁移学习不同的设置,一种名为“半监督迁移学习”的研究开始被学术界关注[3-6],它被用于解决许多实际应用问题,其学习范式一般是目标域中仅有少量样本被标记,而源域中的所有样本都被标记或者是源域中有一个预训练模型.但是,与这种“半监督迁移学习”不一样,在许多实际应用中,源域和目标域中包含标记和未标记样本的情形则非常常见.例如,在计算机辅助诊断(computer-aided diagnosis, CAD)系统[7]应用中,由于标注大量的医学图像非常耗时且成本昂贵,医学专家仅能够仔细诊断标注少量图像.由于设备的老化或升级,先前采集的医学图像很可能与当前采集的医学图像的分布不再相同[8-9].也就是说,在2个不同时间间隔内采集的数据分布不相同.因此,接下来的挑战是如何从源域中的标记和未标记样本有效地学习,以实现对目标域中样本的更准确分类?

半监督多任务学习可用于处理上述问题.文献[10]通过Dirichlet过程的变体在多个分类器参数上使用软共享先验来耦合几个参数化的半监督分类器.文献[11]在高斯过程的协方差中结合了数据的几何形状和任务之间的相似性.文献[12−14]使用深度学习提取多个半监督任务之间共享的特征表示.然而,文献[10−14]所述的这些方法都需要依赖特定类型的分类器.

为应对上述挑战,本文提出一种新的半监督归纳迁移学习框架,称为Co-Transfer,它将半监督学习[15-16]与归纳迁移学习相结合.Co-Transfer首先生成3个TrAdaBoost[17]分类器,用于将知识从原始源域迁移到原始目标域,同时生成另外3个TrAdaBoost分类器,用于将知识从原始目标域迁移到原始源域.这2组分类器都使用从原始源域和原始目标域的原有标记样本的有放回抽样来训练.在Co-Transfer的每一轮迭代中,每组TrAdaBoost分类器使用新的训练集更新,其中一部分训练集是原有的标记样本,一部分是本组TrAdaBoost分类器标记的样本,另一部分则由另一组TrAdaBoost分类器标记.值得强调的是,本文使用Tri-training[18]方式来挑选可靠的伪标记样本.迭代终止后,将从原始源域迁移到原始目标域的3个TrAdaBoost分类器集成作为原始目标域的分类器.在4个UCI数据集[19]和文本分类数据集[17]上的实验结果表明,Co-Transfer可以有效地学习标记和未标记样本提升泛化性能.

本文的主要贡献体现在2点:

1)提出一种源域和目标域都只有部分样本被标记的半监督归纳迁移学习类型.这种学习类型在实际应用中其实很常见.

2)提出一种新的半监督归纳迁移学习框架Co-Transfer.该学习框架在源域和目标域之间进行双向同步的半监督学习和迁移学习,它能很好地适用于源域和目标域都仅有部分样本被标记的迁移学习且不需要特定类型的分类器.

1 相关工作

本节简要回顾半监督学习、基于实例的迁移学习、半监督迁移学习以及半监督多任务学习等相关工作.

1.1 半监督学习

半监督学习[15]的主要思想是学习少量标记样本和大量未标记样本以提高模型泛化能力. 半监督学习假设未标记样本与标记样本具有相同分布. 已有的半监督学习方法主要分为四大类：生成式模型[20]、包装方法[18,21]、低密度分离模型[22]以及基于图的方法[23]等. 其中，包装方法是一类简单且受欢迎的方法，它首先在原有标记样本上训练分类器，然后利用训练好的分类器来预测未标记样本的类别，使用置信的伪标记样本与原有标记样本一起重新训练分类器.

Blum 等人[24]提出了 Co-training 算法，它需要数据有 2 个充分且冗余的视图. Co-training 分别在 2 个不同的视图上训练 2 个分类器，并且使用每个分类器置信的伪标记样本来增强另一个分类器. Dasgupta 等人[25]证明了当数据具有 2 个充分且冗余的视图时，Co-training 训练的分类器可以通过最大化不同组件分类器对未标记样本的一致性来降低分类错误率. 在实际应用场景中，由于数据很难满足这一要求，Zhou 等人[18]提出 Tri-training 算法，该算法不再要求训练数据有 2 个充分且冗余的视图. Tri-training 采用“多数教少数”的方法来避免对分类置信度的估计. Li 等人[21]提出 Co-forest 算法将 Tri-training 扩展到可使用更多的组件分类器. 在 Tri-training 和 Co-forest 中，使用模糊集理论来确保新的伪标记样本能起到积极效果. Van Engelen 等人[15]综述了一些其他包装方法，如 Self-training 与 Boosting 系列. 此外，Triguero 等人[26]提供了一个关于半监督分类中自标记技术的全面综述.

1.2 基于实例的迁移学习

基于实例的迁移学习方法其直观思想是将源域中部分样本与目标域样本一起训练得到一个更高准确率的分类器.

Dai 等人[17]通过扩展 AdaBoost 提出 TrAdaBoost 迁移学习算法. TrAdaBoost 迭代地重新加权源域和目标域样本以减少“坏”的源域样本的权值，同时增大“好”的目标域样本的权值. 此外，Dai 等人[17]还从理论上分析了 TrAdaBoost 的可收敛性. Kamishima 等人[27]通过扩展 bagging 提出 TrBagg 方法. TrBagg 训练过程包括 2 个阶段：学习和过滤. 在学习阶段，使用源域和目标域的抽样样本来训练弱分类器；在过滤阶段，使用目标域样本来评估这些弱分类器. 分类精度低的弱分类器会被丢弃，而分类精度高的弱分类器会被选择来产生最终分类器. Shi 等人[28]通过扩展 Co-training 算法提出 COITL 算法. COITL 算法的关键思想是通过将 Co-training 算法中给样本打伪标记的操作替换为对源域样本进行加权来扩充训练样本集. 在算法 COITL 中，首先使用目标域样本训练 2 个组件分类器，然后每个分类器使用另一个分类器加权的源域样本进行增强.

1.3 半监督迁移学习

目前半监督迁移学习的工作还非常有限，与本文要解决的问题不同，已有的半监督迁移学习旨在解决源域是监督设置但目标域是半监督设置的问题. 根据源域形式的不同，主要分为 2 类：源域样本直接可用或源域样本不可用但源域上的预训练模型可用.

Liu 等人[3]提出一种基于 Tri-training 的半监督迁移学习方法 TriTransfer. 在 TriTransfer 中，使用从源域抽样的样本与目标域的标记样本一起训练 3 个组件分类器，然后使用“多数教少数”的策略来重新训练这 3 个组件分类器，最后加权集成这 3 个组件分类器来产生最终分类器.

Tang 等人[4]提出一种半监督迁移方法以解决汉字识别问题. 首先，使用大量源域样本来训练卷积神经网络；然后使用目标域中少量的标记样本对卷积神经网络模型微调；最后使用目标域中大量的未标记样本和少量的标记样本对微调后的卷积神经网络继续训练以最小化多核最大均值差异（multiple kernel maximum mean discrepancy, MK-MMD）损失.

Abuduweili 等人[5]提出一种半监督迁移学习算法，它可以有效地利用源域的预训练模型以及目标域中的标记和未标记样本. 在所提出的算法中利用自适应一致性正则化方法来将预训练模型与目标域的标记和未标记样本结合在一起训练. 自适应一致性正则化包括 2 个含义：源模型和目标模型之间的自适应一致性；有标记和无标记样本之间的自适应一致性.

Wei 等人[6]提出一种用于图像去雨的半监督迁移网络. 该网络经过合成的带雨图像样本训练后可以迁移于真实且不同类型的带雨图像，解决了目标域缺少训练样本及真实图像与合成图像间分布差异的问题.

1.4 半监督多任务学习

半监督多任务学习旨在有效挖掘任务之间的相关性并探索每个任务中未标记样本的有用信息. Liu 等人[10]基于 Dirichlet 过程的变体使用任务聚簇方法对不同任务进行聚类，且在每个任务中，使用随机游走过程来探索未标记样本中的有用知识. Skolidis 等人[11]使用共区域化的内在模型在高斯过程的框架下编码任务之间的相关性. 此外，半监督多任务学习与深度学习被结合用于解决实际任务，如疾病演化[12]、

药物间相互作用[13]、语义解析[14]和文本挖掘等[29].

2 Co-Transfer

在本文中，为了形式化定义所提出的半监督迁移学习框架，我们引入一些符号.

本文假设源域D_{S}和目标域D_{T}具有相同的特征空间但分布不同. X表示特征空间，$Y\in\{0,1\}$表示类别空间. D_{S}和D_{T}均包含标记和未标记样本. 定义$D_{\mathrm{SL}}=\{(x_1^{\mathrm{S}},y_1^{\mathrm{S}}),(x_2^{\mathrm{S}},y_2^{\mathrm{S}}),\cdots,(x_l^{\mathrm{S}},y_l^{\mathrm{S}})\}$和$D_{\mathrm{TL}}=\{(x_1^{\mathrm{T}},y_1^{\mathrm{T}}),(x_2^{\mathrm{T}},y_2^{\mathrm{T}}),\cdots,(x_p^{\mathrm{T}},y_p^{\mathrm{T}})\}$分别表示源域和目标域中的标记样本，其中$x_i^{\mathrm{S}}\in X$是源域中的一个实例，$y_i^{\mathrm{S}}\in Y$是对应的类别标记，而$x_i^{\mathrm{T}}\in X$则是目标域中的一个实例，$y_i^{\mathrm{T}}\in Y$是对应的类别标记. 定义$D_{\mathrm{SU}}=\{(x_{l+1}^{\mathrm{S}},x_{l+2}^{\mathrm{S}},\cdots,x_{l+m}^{\mathrm{S}})\}$和$D_{\mathrm{TU}}=\left\{(x_{p+1}^{\mathrm{T}},x_{p+2}^{\mathrm{T}},\cdots,x_{p+n}^{\mathrm{T}})\right\}$分别表示源域和目标域中的未标记样本. 此外，还存在测试数据集$D_{\mathrm{Test}}=\{(x_1^{\mathrm{T}},x_2^{\mathrm{T}},\cdots,x_r^{\mathrm{T}})\}$可用，它与目标域分布相同. 通常，未标记样本的数量要远大于标记样本的数量，设置$m\gg l$和$n\gg p$. 所提出的半监督归纳迁移学习的目标是利用$L=[D_{\mathrm{SL}},D_{\mathrm{TL}}]$和$U=[D_{\mathrm{SU}},D_{\mathrm{TU}}]$在目标域上学习一个函数$f:X\to Y$，使得$f$能正确预测$D_{\mathrm{Test}}$中样本的类别标记. 为简洁起见，本文使用$L[0]$和$L[1]$分别表示$D_{\mathrm{SL}}$和$D_{\mathrm{TL}}$，使用$U[0]$和$U[1]$分别表示$D_{\mathrm{SU}}$和$D_{\mathrm{TU}}$.

在Co-Transfer中，需训练2个集成分类器$H^0=\{h_1^0,h_2^0,h_3^0\}$和$H^1=\{h_1^1,h_2^1,h_3^1\}$. 初始时，$H^0$中的每个组件分类器使用目标域与源域的原有标记样本的有放回抽样来训练，采用TrAdaBoost算法实现从目标域到源域的迁移学习. 相对应地，H^1中的每个组件分类器也使用源域与目标域的原有标记样本的有放回抽样来训练，采用TrAdaBoost算法实现从源域到目标域的迁移学习. 抽样策略能保持H^0和H^1各自的组件分类器之间的多样性. 像这样的源域和目标域之间的双向同步迁移学习会迭代很多轮，在每一轮中使用各自新增的伪标记样本分别对H^0和H^1中每个组件分类器进行更新. 由于Co-Transfer在源域和目标域之间实施双向同步迁移学习，本文将原本的源域和目标域分别称为原始源域和原始目标域. 在从原始源域到原始目标域的迁移学习中，原始源域仍然是源域，原始目标域仍然是目标域；而在从原始目标域到原始源域的迁移学习中，原始目标域被当成源域，原始源域则被当成目标域.

算法1. Co-Transfer算法.

输入：训练数据集D_{SL}，D_{SU}，D_{TL}，D_{TU}，迭代次数N，树的深度D；

输出：集成分类器$H^1(x)$.

① $L=[D_{\mathrm{SL}},D_{\mathrm{TL}}]$，$U=[D_{\mathrm{SU}},D_{\mathrm{TU}}]$；
② for $d\in\{0,1\}$ do
③ $H^d\leftarrow\varnothing$；
④ for $i\in\{1,2,3\}$ do
⑤ $S_i^d\leftarrow Bootstrap(L[(d+1)\%2])$；
⑥ $T_i^d\leftarrow Bootstrap(L[d])$；
⑦ $h_i^d\leftarrow TrAdaBoost(S_i^d,T_i^d,N,D)$；
⑧ $H^d\leftarrow H^d\cup h_i^d$，$e_i^{d'}\leftarrow 0.5$，$l_i^{d'}\leftarrow 0$；
⑨ end for
⑩ $e^{d'}\leftarrow 0.5$，$l^{d'}\leftarrow 0$；
⑪ end for
⑫ repeat
⑬ for $d\in\{0,1\}$ do
⑭ $L^d\leftarrow\varnothing$，$Update^d\leftarrow$FALSE；
⑮ $e^d=MeasureEnsembleError(H^d,L[d])$；
⑯ for $i\in\{1,2,3\}$ do
⑰ $L_i^d\leftarrow\varnothing$，$Update_i^d\leftarrow$FALSE；
⑱ $e_i^d=MeasureClassifierError(h_j^d\&h_k^d,L[d])$；
⑲ if $e_i^d<e_i^{d'}$ do
⑳ $L_i^d\leftarrow PseudoLabel(h_j^d\&h_k^d,U[d])$；
㉑ if $l_i^{d'}<\left|L_i^d\right|$ do
㉒ if $e_i^d\left|L_i^d\right|<e_i^{d'}l_i^{d'}$ do
㉓ $Update_i^d\leftarrow$TRUE；
㉔ else if $l_i^{d'}>\dfrac{e_i^d}{e_i^{d'}-e_i^d}$ do
㉕ $L_i^d\leftarrow Subsample\left(L_i^d,\left\lceil\dfrac{e_i^{d'}l_i^{d'}}{e_i^d}-1\right\rceil\right)$；
/*使用式(2)抽样*/
㉖ $Update_i^d\leftarrow$TRUE；
㉗ end if
㉘ end if
㉙ end if
㉚ end for
㉛ if $e^d<e^{d'}$ do
㉜ $L^d\leftarrow ScreenPseudoLabel(H^d,L_i^d,L_j^d,L_k^d)$；
㉝ if $l^{d'}<\left|L^d\right|$ do
㉞ if $e^d\left|L^d\right|<e^{d'}l^{d'}$ do
㉟ $Update^d\leftarrow$TRUE；
㊱ else if $l^{d'}>\dfrac{e^d}{e^{d'}-e^d}$ do
㊲ $L^d\leftarrow Subsample\left(L^d,\left\lceil\dfrac{e^{d'}l^{d'}}{e^d}-1\right\rceil\right)$；

/*使用式(4)抽样*/

㊳ $Update^d \leftarrow$ TRUE;

㊴ end if

㊵ end if

㊶ end if

㊷ end for

㊸ for $d \in \{0, 1\}$ do

㊹ for $i \in \{1, 2, 3\}$ do

㊺ if $Update_i^d \cap Update^{(d+1)\%2}$ do

㊻ $NS_i^d \leftarrow L[(d+1)\%2] \cup L^{(d+1)\%2}$;

㊼ $NT_i^d \leftarrow L[d] \cup L_i^d$;

㊽ $Nh_i^d \leftarrow TrAdaBoost(NS_i^d, NT_i^d, N, D)$;

㊾ $h_i^d \leftarrow Nh_i^d$, $e_i^{d'} \leftarrow e_i^d$, $l_i^{d'} \leftarrow |L_i^d|$;

㊿ end if

(51) $e^{d'} \leftarrow e^d$, $l^{d'} \leftarrow |L^d|$;

(52) end for

(53) end for

(54) until $H^d(d \in \{0, 1\})$ 中的每个组件分类器都不再被重新训练.

更详细地，在 Co-Transfer 的每一轮双向同步迁移学习中，将集成分类器$H^d(d \in \{0, 1\}$，$d = 0$对应原始源域，而$d = 1$对应原始目标域)的组件分类器$h_i^d(i \in \{1, 2, 3\})$作为初始分类器，使用 Tri-training 方式从$U[d]$中选择样本打上标记. 具体做法是：若其他 2 个组件分类器$H_i^d = h_j^d \cup h_k^d(j \neq k \neq i)$对未标记样本给出的类别标记相同，则该未标记样本才会被标记，将该伪标记样本添加到数据集$L_i^d(i \in \{1, 2, 3\})$. 另外，若集成分类器$H^d(d \in \{0, 1\})$中的 3 个组件分类器对未标记样本都给出了相同标记，则该未标记样本会被选中作为伪标记样本而添加到数据集L^d. 然后，将L^d与原始标记样本$L[d]$合并作为新的源域数据，再将$L_i^{(d+1)\%2}(i \in \{1, 2, 3\})$分别与$L[(d+1)\%2]$合并作为新的目标域数据. 最后，利用 TrAdaBoost 算法进行训练，以实现从源域到目标域的迁移学习，而得到由新的组件分类器$h_i^{(d+1)\%2}(i \in \{1, 2, 3\})$构成的集成分类器$H^{(d+1)\%2}$. 值得注意的是：由$H^d$和$H_i^d$选择的所有未标记样本都不会从$U[d]$中删除. 因此，它们可能会在接下来的迭代中被再次选中. $d = 0$和$d = 1$所对应的 2 个迁移学习过程被同步执行. 当$H^d(d \in \{0, 1\})$中的每个组件分类器都不再被重新训练时，迭代结束，将最终得到的H^1作为原始目标域上的分类器.

与 Tri-training[18] 一样，使用集成分类器$H^d(d \in \{0, 1\})$来打伪标记，不仅可以避免使用复杂的置信度和可迁移性的估计，而且使得对未标记样本的标记比单个分类器更可靠. 然而，尽管集成分类器比单个分类器对未标记样本的类别估计更可靠，但错误标记样本无法避免. 因此，$H^d(d \in \{0, 1\})$将会接收噪声伪标记样本进行训练，这可能会影响$H^d(d \in \{0, 1\})$的更新. 幸运的是，Zhou 等人[18] 提出：这种噪声造成的负面影响可以通过在特定条件下用足够数量的伪标记样本来补偿. 因此，受到 Zhou 等人[18] 工作的启发，本文提出在 2 种情况下采取一定策略来限制噪声伪标记样本造成的影响.

第 1 种情况是如何选择$U[d]$中未标记样本作为目标域样本以重新训练$h_i^d(i \in \{1, 2, 3\})$. 令$L_i^{d,t}$和$L_i^{d,t-1}$表示在第 t 轮和第 $t-1$ 轮迭代中使用 Tri-training 方式从$U[d]$中选择可打标记的样本，大小分别为$|L_i^{d,t}|$和$|L_i^{d,t-1}|$. $e_i^{d,t}$和$e_i^{d,t-1}$分别表示$H_i^d = h_j^d \cup h_k^d$在第 t 轮和第 $t-1$ 轮迭代中的分类错误率的上界. 因此，根据 Zhou 等人[18] 提出的理论，需满足约束：

$$0 < \frac{e_i^{d,t}}{e_i^{d,t-1}} < \frac{|L_i^{d,t-1}|}{|L_i^{d,t}|} < 1. \tag{1}$$

根据式(1)，当$e_i^{d,t} < e_i^{d,t-1}$和$|L_i^{d,t}| > |L_i^{d,t-1}|$满足时，$e_i^{d,t}|L_i^{d,t}| < e_i^{d,t-1}|L_i^{d,t-1}|$仍可能不成立，因为可能存在$|L_i^{d,t}| \gg |L_i^{d,t-1}|$. 为了使式(1)在这种情况下再次成立，必须对$L_i^{d,t}$进行 2 次采样，以使得$|L_i^{d,t}| < e_i^{d,t-1}|L_i^{d,t-1}|/e_i^{d,t}$成立. 因此，对$L_i^{d,t}$需使用式(2)方法进行随机下采样：

$$L_i^{d,t} \leftarrow Subsample\left(L_i^{d,t}, \lceil \frac{e_i^{d,t-1}|L_i^{d,t-1}|}{e_i^{d,t}} - 1 \rceil\right). \tag{2}$$

第 2 种情况是如何选择$U[(d+1)\%2]$中未标记的样本作为源域样本以重新训练$h_i^d(i \in \{1, 2, 3\})$. 令$L^{d,t}$和$L^{d,t-1}$分别表示在第 t 轮和第 $t-1$ 轮迭代中从$U[d]$中选择可打标记的样本，其大小分别为$|L^{d,t}|$和$|L^{d,t-1}|$. $e^{d,t}$和$e^{d,t-1}$分别表示H^d在第 t 轮迭代和第 $t-1$ 轮迭代中的分类错误率的上界. 因此，同样根据 Zhou 等人[18] 提出的理论，需满足约束：

$$0 < \frac{e^{d,t}}{e^{d,t-1}} < \frac{|L^{d,t-1}|}{|L^{d,t}|} < 1. \tag{3}$$

根据式(3)，当$e^{d,t} < e^{d,t-1}$和$|L^{d,t}| > |L^{d,t-1}|$满足时，$e^{d,t}|L^{d,t}| < e^{d,t-1}|L^{d,t-1}|$仍可能不成立，因为可能存在$|L^{d,t}| \gg |L^{d,t-1}|$. 为了使式(3)在这种情况下再次成立，必须对$|L^{d,t}|$进行 2 次采样，使得$|L^{d,t}| < e^{d,t-1}|L^{d,t-1}|/e^{d,t}$. 因此，对$L^{d,t}$需使用以下方法进行随机下采样：

$$L^{d,t} \leftarrow Subsample\left(L^{d,t}, \left\lceil \frac{e^{d,t-1}|L^{d,t-1}|}{e^{d,t}} - 1 \right\rceil\right). \tag{4}$$

第 2 种情况引入的约束使得标记更可靠的伪标记样本加入源域，使源域数据分布更接近真实数据分布，进而导致更可靠的伪标记样本添加到目标域中，这对于 Co-Transfer 中的迭代过程非常重要.

采用决策树作为组件分类器的 Co-Transfer 算法伪代码在算法 1 中给出，其中参数 N 为 *TrAdaBoost* 的迭代次数. 行⑤~⑦表示分别从 $L[(d+1)\%2]$ 和 $L[d]$ 通过函数 *Bootstrap* 抽样而得到 S_i^d 和 T_i^d. 将 S_i^d 和 T_i^d 分别作为源域和目标域数据，用 *TrAdaBoost* 学习得到初始的集成分类器. 在行⑮中，函数 *MeasureEnsembleError* 用于估计集成分类器 H^d 的分类错误率. 在行⑱中，函数 *MeasureClassifierError* 则用于估计 $H_i^d = h_j^d \cup h_k^d$ 的分类错误率. 这 2 个错误率都是在原有标记数据集 $L[d]$ 上估计的. 详细来说，估算 H^d 的错误率是通过将 H^d 的 3 个组件分类器 $h_i^d (i \in \{1,2,3\})$ 预测一致但分错的标记样本数与预测一致的标记样本数之比来近似得到. 类似地，H_i^d 的错误率是通过将 H_i^d 预测一致但被分错的标记样本数与预测一致的标记样本数之比来近似估算. 在行⑳中，函数 *PseudoLabel* 是从 $U[d]$ 中选择 h_j^d 和 h_k^d 给出相同标记的样本打上伪标记. 在行㉜中，函数 *ScreenPseudoLabel* 是从 L_i^d, L_j^d, L_k^d 中选择出伪标记相同的样本. 函数 *Subsample*(S,t) 从 S 中随机删除 t 个样本. 行㊸~㊸使用新的源域数据和目标域数据来重新训练 H^d 的每个组件分类器. 图 1 中的数据流图描述了 Co-Transfer 的训练过程.

3 实 验

3.1 数据集

本文使用迁移学习研究[17,28]中常用的 4 个 UCI 数据集和文本分类数据集 Reuters 进行实验. 这些数据集都已被证明原始源域能有效地助力原始目标域

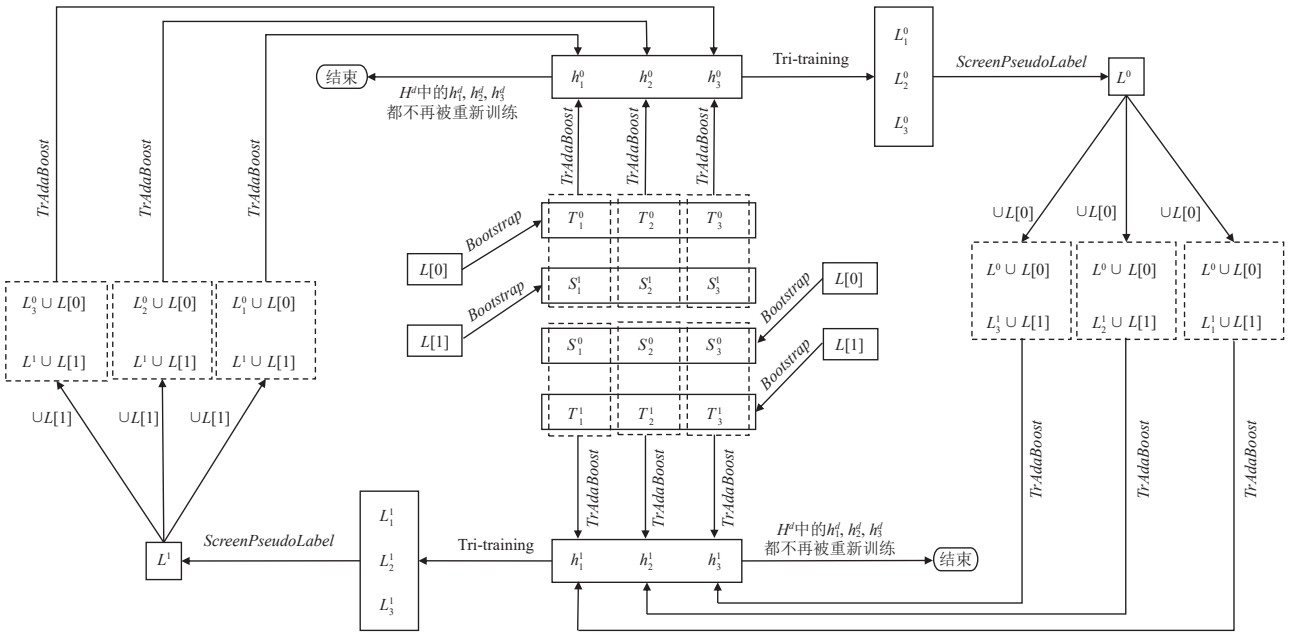

Fig. 1 Data flow diagram of Co-Transfer training process

图 1 Co-Transfer 训练过程的数据流图

Table 1 Experimental Data Sets

表 1 实验数据集

数据集	属性数	$\|D_S\|$	源域正负比	$\|D_T\|$	目标域正负比	类别数
Mushroom	22	4 608	0.562 5/0.437 5	3 516	0.459 6/0.540 4	2
Waveform	21	1 722	0.508 1/0.491 9	1 582	0.494 3/0.505 7	2
Magic	10	9 718	0.707 5/0.292 5	9 302	0.586 5/0.413 5	2
Splice	60	795	0.459 1/0.540 9	740	0.543 2/0.456 8	2
Orgs vs People	4 771	1 237	0.524 7/0.475 3	1 208	0.514 1/0.485 9	2
Orgs vs Places	4 415	1 016	0.578 7/0.421 3	1 043	0.562 8/0.437 2	2
People vs Places	4 562	1 077	0.602 6/0.397 4	1 077	0.576 6/0.423 4	2

的学习. 表 1 罗列了这些数据集的详细信息.

3.2 实验设置

对于表 1 中每个数据集的目标域数据使用 5 折交叉验证进行分类错误率评价. 在每折交叉验证中, 将目标域中的样本划分为目标域训练数据集D_T和测试数据集D_{Test}, 而将源域中的全部样本当作训练数据集D_S. 再根据预设的有标记样本比, 随机从源域训练数据集D_S和目标域训练数据集D_T中挑选样本构成标记样本集D_{SL}和D_{TL},剩余的样本构成未标记样本集D_{SU}和D_{TU}. 为增加样本标记的随机性, 在每折交叉验证中, 重复 2 次对源域训练数据集D_S随机划分为D_{SL}和D_{SU}, 重复 3 次对目标域训练数据集D_T随机划分为D_{TL}和D_{TU}. 因此, 最终的错误率是 30(5×6)次测试结果的平均值.

为评测与分析 Co-Transfer 是否可以有效学习源域和目标域中的标记样本与未标记样本, 引入 7 个算法进行对比, 包括决策树(decision tree, DT)、TrAdaBoost、Tri-training、TrAdaBoost_A、Co-Transfer_S、TrAdaBoost_S、Co-Transfer_T.

算法 DT 表示仅在D_{TL}上训练决策树对测试数据进行分类; 算法 TrAdaBoost 表示在D_{SL}和D_{TL}上使用 TrAdaBoost 方法训练分类器对测试数据进行分类; 算法 Tri-training 表示在D_{TL}和D_{TU}上使用 Tri-training 方法训练分类器对测试数据进行分类; 算法TrAdaBoost_A表示在将D_S和D_T中的全部样本打上标记再使用 TrAdaBoost 方法训练分类器对测试数据进行分类; 算法 Co-Transfer_S表示采用 Co-Transfer 的框架进行学习, 但在迭代过程中目标域的样本只使用D_{TL}, 也就是说在双向迭代过程中不会在D_{TL}的基础上再增加伪标记样本; 算法TrAdaBoost_S表示将D_S中的全部样本打上标记和D_{TL}使用 TrAdaBoost 方法训练分类器对测试数据进行分类; 算法 Co-Transfer_T则表示采用 Co-Transfer 的框架进行学习, 但在迭代过程中源域样本只使用D_{SL}, 也就是说在双向迭代过程中不会在D_{SL}的基础上再增加伪标记样本. 表 2 列出了 Co-Transfer 和各对比算法的样本使用策略.

由表 2 可知: 当原始源域到原始目标域能实现正迁移时, DT 的分类错误率应该最高, TrAdaBoost_A的分类错误率则应该最低. 与 Co-Transfer 相比较, Co-Transfer_S不使用D_{TU}, 而 Co-Transfer_T不使用D_{SU}. TrAdaBoost 则既不使用D_{SU}, 也不使用D_{TU}. 因此, 将 Co-Transfer 分别与 TrAdaBoost, Co-Transfer_S, Co-Transfer_T相比较, 可观察 Co-Transfer 是否可以有效地从原始源域和原始目标域的未标记样本中学习. 与 TrAdaBoost 相比, TrAdaBoost_S多用了D_{SU}, 因此, 将TrAdaBoost_S与 TrAdaBoost 对比可知利用D_{SU}能否辅助原始目标域学习得更好.

Table 2 Data Strategy Used by the Algorithms

表 2 各算法的数据使用策略

方法	训练数据				测试数据
	D_{SL}	D_{SU}	D_{TL}	D_{TU}	
DT	×	×	√	×	D_{Test}
TrAdaBoost	√	×	√	×	D_{Test}
Tri-training	×	×	√	/	D_{Test}
TrAdaBoost_A	√	√	√	√	D_{Test}
Co-Transfer_S	√	/	√	×	D_{Test}
TrAdaBoost_S	√	√	√	×	D_{Test}
Co-Transfer_T	√	×	√	/	D_{Test}
Co-Transfer	√	/	√	/	D_{Test}

注: “×”表示未使用该数据; “/”表示使用该数据但未使用样本的类别标记; “√”则表示使用该数据和全部样本的类别标记.

本文使用标准 t 统计检验检查 Co-Transfer 和每个对比算法的泛化能力的区别是否具有 95% 置信度的显著性.

各算法的参数设置: 对于 Tri-training, 使用 C4.5 决策树作为组件分类器并且不做剪枝处理; 而对于 TrAdaBoost, TrAdaBoost_A, Co-Transfer_S, TrAdaBoost_S, Co-Transfer_T, Co-Transfer, 使用相同的参数, 即对于数据集 Mushroom 设置迭代次数 N=10, 树的深度 D=10; 对于数据集 Waveform 设置 N=65, D=4; 对于数据集 Magic 设置 N=35, D=20; 对于数据集 Splice 设置 N=15, D=50; 对于数据集 Orgs vs People 设置 N=50, D=5; 对于数据集 Orgs vs Places 设置 N=20, D=4; 对于数据集 People vs Places 设置 N=50, D=3.

3.3 实验结果和分析

表 3~6 显示, 当原始源域和原始目标域的标记比率相同时, 不同标记比率下 Co-Transfer 和所有对比算法在原始目标域测试集上的分类错误率. 从表 3~6 可以观察到: 在各种标记比率条件下, DT 的分类错误率最高. 当标记比率为 10% 和 20% 时, TrAdaBoost_A显然要比 Co-Transfer 的泛化能力强; 而当标记比率为 40% 和 50% 时, TrAdaBoost_A与 Co-Transfer 的泛化能力则变得相当; 在各种标记比率条件下, Co-Transfer 的泛化能力比 TrAdaBoost, Tri-training, Co-Transfer_S的泛化能力都要好, 这说明 Co-Transfer 能有效利用源域的标记样本和目标域的未标记样本.

从表 3~6 还可以观察到: Co-Transfer 的泛化能力不比 Co-Transfer_T弱. 为分析 Co-Transfer 能否有效利

Table 3 Error Rates of the Comparative Algorithms Under the Label Rate 10% of Original Source Domain and Target Domain

表 3 在原始源域与原始目标域标记比率均为 10% 下所对比算法的错误率

数据集	DT	TrAdaBoost	Tri-training	$\text{TrAdaBoost}_\text{A}$	Co-Transfer	$\text{Co-Transfer}_\text{S}$	$\text{Co-Transfer}_\text{T}$
Mushroom	0.004 ★	0.006 ★	0.004 ★	0.0 ○	0.006	0.006 ★	0.006 ★
Waveform	0.198 ●	0.145 ●	0.190 ●	0.160 ●	0.136	0.147 ●	0.138 ★
Magic	0.156 ●	0.159 ●	0.149 ●	0.107 ○	0.139	0.152 ●	0.140 ★
Splice	0.134 ●	0.118 ●	0.112 ●	0.056 ○	0.084	0.103 ●	0.094 ●
Orgs vs People	0.264 ●	0.216 ●	0.258 ●	0.158 ○	0.192	0.209 ●	0.199 ★
Orgs vs Places	0.279 ●	0.266 ●	0.274 ●	0.178 ○	0.229	0.266 ●	0.239 ●
People vs Places	0.274 ●	0.228 ●	0.264 ●	0.154 ○	0.198	0.236 ●	0.209 ★
平均错误率	0.187	0.163	0.179	0.116	0.141	0.160	0.146
赢/平/输的次数	6/1/0	6/1/0	6/1/0	1/0/6		6/1/0	2/5/0

注：“●”或“○”表示有 95% 的置信度，说明 Co-Transfer 显著优于或差于所比较的算法；“★”表示 Co-Transfer 没有以 95% 的置信度优于所比较的算法.

Table 4 Error Rates of the Comparative Algorithms Under the Label Rate 20% of Original Source Domain and Target Domain

表 4 在原始源域与原始目标域标记比率均为 20% 下所对比算法的错误率

数据集	DT	TrAdaBoost	Tri-training	$\text{TrAdaBoost}_\text{A}$	Co-Transfer	$\text{Co-Transfer}_\text{S}$	$\text{Co-Transfer}_\text{T}$
Mushroom	0.002 ★	0.001 ★	0.002 ★	0.0 ○	0.002	0.002 ★	0.002 ★
Waveform	0.173 ●	0.148 ●	0.169 ●	0.160 ●	0.126	0.145 ●	0.131 ★
Magic	0.149 ●	0.146 ●	0.139 ●	0.107 ○	0.122	0.143 ●	0.122 ★
Splice	0.104 ●	0.096 ●	0.095 ●	0.056 ○	0.081	0.101 ●	0.085 ★
Orgs vs People	0.208 ●	0.18 ●	0.198 ●	0.158 ★	0.154	0.181 ●	0.157 ★
Orgs vs Places	0.242 ●	0.237 ●	0.231 ●	0.178 ○	0.195	0.226 ●	0.208 ●
People vs Places	0.217 ●	0.206 ●	0.199 ●	0.154 ○	0.17	0.207 ●	0.178 ★
平均错误率	0.156	0.145	0.148	0.116	0.121	0.144	0.126
赢/平/输的次数	6/1/0	6/1/0	6/1/0	1/1/5		6/1/0	1/6/0

注：“●”或“○”表示有 95% 的置信度，说明 Co-Transfer 显著优于或差于所比较的算法；“★”表示 Co-Transfer 没有以 95% 的置信度优于所比较的算法.

Table 5 Error Rates of the Comparative Algorithms Under the Label Rate 40% of Original Source Domain and Target Domain

表 5 在原始源域与原始目标域标记比率均为 40% 下所对比算法的错误率

数据集	DT	TrAdaBoost	Tri-training	$\text{TrAdaBoost}_\text{A}$	Co-Transfer	$\text{Co-Transfer}_\text{S}$	$\text{Co-Transfer}_\text{T}$
Mushroom	0.0 ★	0.0 ★	0.0 ★	0.0 ★	0.0	0.0 ★	0.0 ★
Waveform	0.165 ●	0.145 ●	0.165 ●	0.160 ●	0.132	0.145 ●	0.131 ★
Magic	0.142 ●	0.135 ●	0.130 ●	0.107 ★	0.109	0.132 ●	0.112 ★
Splice	0.084 ★	0.088 ●	0.070 ★	0.056 ○	0.077	0.081 ★	0.074 ★
Orgs vs People	0.180 ●	0.185 ●	0.165 ●	0.158 ★	0.155	0.183 ●	0.155★
Orgs vs Places	0.197 ●	0.219 ●	0.190 ★	0.178 ★	0.185	0.220 ●	0.192 ★
People vs Places	0.190 ●	0.192 ●	0.177 ●	0.154 ●	0.142	0.192 ●	0.145 ★
平均错误率	0.137	0.138	0.128	0.116	0.114	0.136	0.116
赢/平/输的次数	5/2/0	6/1/0	4/3/0	2/4/1		5/2/0	0/7/0

注：“●”或“○”表示有 95% 的置信度，说明 Co-Transfer 显著优于或差于所比较的算法；“★”表示 Co-Transfer 没有以 95% 的置信度优于所比较的算法.

用原始源域的未标记样本D_{SU}，我们将$\text{TrAdaBoost}_\text{S}$与TrAdaBoost进行了比较，结果如表 7 所示. 从表 7 中可以观察到：1)标记比率超过 20% 后，提升标记比率不太可能会带来$\text{TrAdaBoost}_\text{S}$的泛化能力的显著提升；2)在各种标记比率下，对于数据集 Mushroom, Waveform, Magic, $\text{TrAdaBoost}_\text{S}$与 TrAdaBoost 在原始目标域上的泛化能力没有显著区别；3)对于数据集 Splice, Orgs vs People, Orgs vs Places, People vs Places, 增加源域的标记样本数量可能得到正向迁移的效果. 再结合表 3~6 可知：Co-Transfer 能否有效利用原始源域的未标记

Table 6 Error Rates of the Comparative Algorithms Under the Label Rate 50% of Original Source Domain and Target Domain

表 6 在原始源域与原始目标域标记比率均为 50% 下所对比算法的错误率

数据集	DT	TrAdaBoost	Tri-training	$\text{TrAdaBoost}_\text{A}$	Co-Transfer	$\text{Co-Transfer}_\text{S}$	$\text{Co-Transfer}_\text{T}$
Mushroom	0.0 ★	0.0 ★	0.0 ★	0.0 ★	0.0	0.0 ★	0.0 ★
Waveform	0.169 ●	0.151 ●	0.167 ●	0.160 ●	0.139	0.152 ●	0.139 ★
Magic	0.141 ●	0.137 ●	0.130 ●	0.107 ★	0.108	0.129 ●	0.108 ★
Splice	0.081 ●	0.080 ●	0.069 ★	0.056 ○	0.071	0.071 ★	0.070 ★
Orgs vs People	0.170 ●	0.199 ●	0.162 ●	0.158 ★	0.149	0.182 ●	0.151 ★
Orgs vs Places	0.188 ●	0.214 ●	0.168 ★	0.178 ★	0.176	0.212 ●	0.175 ★
People vs Places	0.183 ●	0.200 ●	0.166 ●	0.154 ●	0.135	0.183 ●	0.143 ★
平均错误率	0.133	0.140	0.123	0.116	0.111	0.133	0.112
赢/平/输的次数	6/1/0	6/1/0	4/3/0	2/4/1		5/2/0	0/7/0

注：“●”或“○”表示有 95% 的置信度，说明 Co-Transfer 显著优于或差于所比较的算法；“★”表示 Co-Transfer 没有以 95% 的置信度优于所比较的算法.

Table 7 Error Rates of TrAdaBoost and $\text{TrAdaBoost}_\text{S}$ Under Different Label Rates

表 7 不同标记比率下 TrAdaBoost 与$\text{TrAdaBoost}_\text{S}$算法的错误率

数据集	10% 标记比例		20% 标记比例		40% 标记比例		50% 标记比例	
	TrAdaBoost	$\text{TrAdaBoost}_\text{S}$	TrAdaBoost	$\text{TrAdaBoost}_\text{S}$	TrAdaBoost	$\text{TrAdaBoost}_\text{S}$	TrAdaBoost	$\text{TrAdaBoost}_\text{S}$
Mushroom	0.006 ★	0.005	0.001 ★	0.002	0.0 ★	0.0	0.0 ★	0.0
Waveform	0.145 ★	0.148	0.148 ★	0.143	0.145 ★	0.148	0.151 ★	0.157
Magic	0.159 ★	0.157	0.146 ★	0.145	0.135 ★	0.13	0.137 ★	0.138
Splice	0.118 ●	0.098	0.096 ★	0.097	0.088 ●	0.075	0.08 ★	0.079
Orgs vs People	0.216 ●	0.199	0.18 ★	0.172	0.185 ★	0.189	0.199 ★	0.185
Orgs vs Places	0.266 ★	0.262	0.237 ●	0.215	0.219 ★	0.226	0.214 ★	0.222
People vs Places	0.228 ★	0.221	0.206 ○	0.223	0.192 ★	0.182	0.2 ●	0.175
平均错误率	0.163	0.156	0.145	0.142	0.138	0.136	0.14	0.137

注：“●”或“○”表示有 95% 的置信度，说明$\text{TrAdaBoost}_\text{S}$显著优于或差于 TrAdaBoost；“★”表示$\text{TrAdaBoost}_\text{S}$没有以 95% 的置信度优于 TrAdaBoost.

样本应该与数据集的自身特性相关.

为更深入观察 Co-Transfer 的学习过程，我们平均不同标记比率下各种对比算法在文本分类数据集上每轮迭代的分类错误率. 图 2 描述了各算法从初始迭代到最终迭代的平均错误率的变化. 需要解释的是：1)在与 Co-Transfer 对比的算法中，仅有 Tri-training, $\text{Co-Transfer}_\text{S}$, $\text{Co-Transfer}_\text{T}$使用了无标记样本，其他算法都只使用标记样本；2)当一个算法提前终止了，我们保持其错误率补齐了数据. 从图 2 中可以观察到：1)不使用无标记样本的算法无迭代过程，仅有最终分类错误率，DT 错误率最高，其次是 TrAdaBoost 和$\text{TrAdaBoost}_\text{S}$，最后是$\text{TrAdaBoost}_\text{A}$；2)初始迭代时，Co-Transfer, $\text{Co-Transfer}_\text{S}$, $\text{Co-Transfer}_\text{T}$的平均分类错误率几乎相同但略微低于 Tri-training，随着迭代不断进行，Co-Transfer 的平均分类错误率不断降低，并快速收敛.

此外，还观察了原始源域的标记比率要高于原始目标域的标记比率这一情况. 表 8 和表 9 显示原始源域标记比率为 50%，而原始目标域的标记比率分别为 10% 和 20% 时，Co-Transfer 与各对比算法的分类错误率. 可以观察到：DT 的分类错误率最高；$\text{TrAdaBoost}_\text{A}$显然要比 Co-Transfer 的泛化能力强；在各种标记比率条件下，Co-Transfer 的泛化能力比 TrAdaBoost, Tri-training, $\text{Co-Transfer}_\text{S}$的泛化能力都要好. 这说明：在原始源域的标记比率比原始目标域的标记比率高时，Co-Transfer 还能有效地利用原始源域的标记样本和原始目标域的未标记样本.

从表 8 和表 9 中还可观察到：Co-Transfer 的泛化能力不比 $\text{Co-Transfer}_\text{T}$弱. 为分析 Co-Transfer 能否有效利用原始源域的未标记样本D_{SU}，同样将$\text{TrAdaBoost}_\text{S}$与 TrAdaBoost 进行比较，结果如表 10 所示. 可以观察到：当原始源域标记比率较高时，增加原始源域标记样本的数量很可能不带来原始目标域分类错误率的显著下降. 因此，表 8~10 中暂不能解释 Co-Transfer 的泛化能力不比 $\text{Co-Transfer}_\text{T}$弱.

综合表 3~10 中实验结果，可以进一步推测：原始

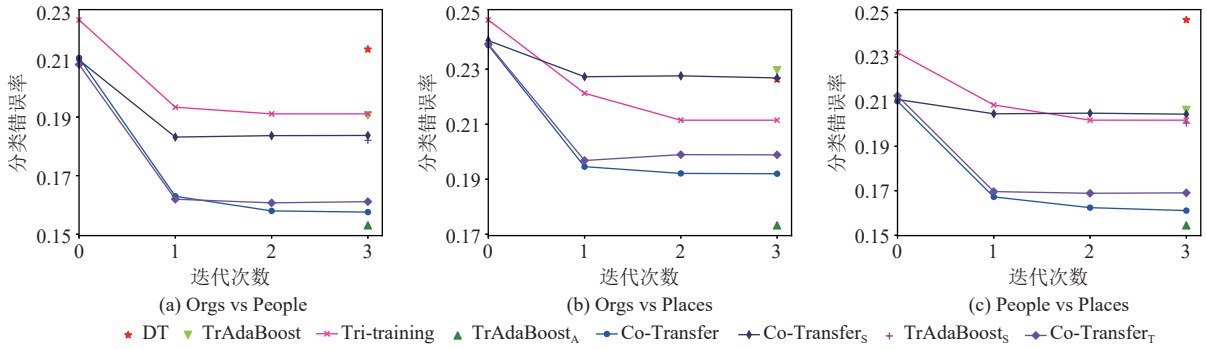

Fig. 2 Error rates of the comparative algorithms during their iteration on the text classification tasks

图 2 文本分类任务上各对比算法在迭代过程中的错误率

Table 8 Error Rates of the Comparative Algorithms Under the Label Rate 50% of the Original Source Domain and 10% of the Original Target Domain

表 8 在原始源域标记比率为 50%、原始目标域标记比率为 10% 下所有对比算法的错误率

数据集	DT	TrAdaBoost	Tri-training	TrAdaBoost_A	Co-Transfer	Co-Transfer_S	Co-Transfer_T
Mushroom	0.004 ★	0.006 ★	0.004 ★	0.0 ○	0.006	0.006 ★	0.005 ★
Waveform	0.198 ●	0.147 ●	0.190 ●	0.160 ●	0.131	0.148 ●	0.140 ●
Magic	0.156 ●	0.156 ●	0.149 ●	0.107 ○	0.137	0.154 ●	0.140 ★
Splice	0.134 ●	0.109 ●	0.112 ●	0.056 ○	0.079	0.106 ●	0.085 ★
Orgs vs People	0.264 ●	0.216 ●	0.258 ●	0.158 ○	0.186	0.212 ●	0.194 ★
Orgs vs Places	0.279 ●	0.254 ●	0.274 ●	0.178 ○	0.220	0.257 ●	0.234 ●
People vs Places	0.274 ●	0.222 ●	0.264 ●	0.154 ○	0.196	0.221 ●	0.208 ★
平均错误率	0.187	0.159	0.179	0.116	0.136	0.158	0.144
赢/平/输的次数	6/1/0	6/1/0	6/1/0	1/0/6		6/1/0	2/5/0

注：“●”或“○”表示有 95% 的置信度，说明 Co-Transfer 显著优于或差于所比较的算法；“★”表示 Co-Transfer 没有以 95% 的置信度优于所比较的算法.

Table 9 Error Rates of the Comparative Algorithms Under the Label Rate 50% of the Original Source Domain and 20% of the Original Target Domain

表 9 在原始源域标记比率为 50%、原始目标域标记比率为 20% 下所有对比算法的错误率

数据集	DT	TrAdaBoost	Tri-training	TrAdaBoost_A	Co-Transfer	Co-Transfer_S	Co-Transfer_T
Mushroom	0.002 ★	0.003 ★	0.002 ★	0.0 ○	0.001	0.002 ★	0.002 ★
Waveform	0.173 ●	0.149 ●	0.169 ●	0.160 ●	0.133	0.142 ●	0.130 ★
Magic	0.149 ●	0.143 ●	0.139 ●	0.107 ○	0.124	0.144 ●	0.123 ★
Splice	0.104 ●	0.091 ●	0.095 ●	0.056 ○	0.082	0.094 ●	0.077 ★
Orgs vs People	0.208 ●	0.176 ★	0.198 ●	0.158 ○	0.168	0.177 ●	0.174 ★
Orgs vs Places	0.242 ●	0.234 ●	0.231 ●	0.178 ○	0.195	0.231 ●	0.2 ★
People vs Places	0.217 ●	0.216 ●	0.199 ●	0.154 ○	0.173	0.204 ●	0.179 ★
平均错误率	0.156	0.145	0.148	0.116	0.127	0.142	0.124
赢/平/输的次数	6/1/0	5/2/0	6/1/0	1/6/0		5/2/0	0/7/0

注：“●”或“○”表示有 95% 的置信度，说明 Co-Transfer 显著优于或差于所比较的算法；“★”表示 Co-Transfer 没有以 95% 的置信度优于所比较的算法.

源域和原始目标域的标记比率的同步提升对于 Co-Transfer 的学习过程的影响比较复杂. 这是因为：仅增加源域标记样本数量并不一定总能提升迁移学习的效果；目标域标记样本数量的增加会掩盖源域标记样本带来的正向迁移. 因此，在 Co-Transfer 的学习过程中，源域和目标域很可能存在一个博弈过程.

当 Co-Transfer 的分类器采用决策树时，考虑到决策树的深度 D 和 TrAdaBoost 的迭代次数 N 可能会影响 Co-Transfer 的泛化能力. 因此，在文本分类数据集上进一步研究了树的深度 D 和迭代次数 N 对 Co-Transfer 泛化能力的影响. 图 3 描述了将不同的标记比率条件下的分类错误率平均后 Co-Transfer 的变化

Table 10 Error Rates of TrAdaBoost and TrAdaBoost$_S$ Under the Label Rate 50% of the Original Source Domain and 10% and 20% of the Original Target Domain, Respectively

表 10 在原始源域标记比率为 50%、原始目标域标记比率分别为 10% 和 20% 下 TrAdaBoost 与TrAdaBoost$_S$的错误率

数据集	源域 50%, 目标域 10%		源域 50%, 目标域 20%	
	TrAdaBoost	TrAdaBoost$_S$	TrAdaBoost	TrAdaBoost$_S$
Mushroom	0.006 ★	0.005	0.003 ★	0.002
Waveform	0.147 ★	0.148	0.149 ★	0.143
Magic	0.156 ★	0.157	0.143 ★	0.145
Splice	0.109 ★	0.098	0.091 ★	0.097
Orgs vs People	0.216 ●	0.199	0.176 ★	0.172
Orgs vs Places	0.254 ★	0.262	0.234 ●	0.215
People vs Places	0.222 ★	0.221	0.216 ★	0.223
平均错误率	0.159	0.156	0.145	0.142

注: “●”或“○”表示有 95% 的置信度，说明TrAdaBoost$_S$显著优于或差于 TrAdaBoost; “★”表示TrAdaBoost$_S$没有以 95% 的置信度优于 TrAdaBoost.

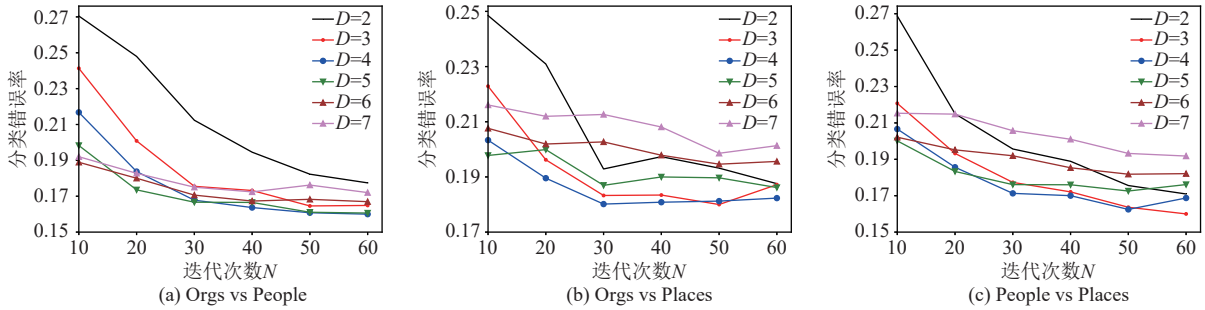

Fig. 3 Error rates of Co-Transfer with different N and D

图 3 不同 N 和 D 条件下 Co-Transfer 的错误率

曲线. 从中可以观察到: 在文本分类数据集上, 参数 N 和 D 对 Co-Transfer 算法分类错误率有较大影响. 当固定 D 时, 随着 N 的增大 Co-Transfer 的分类错误率越来越低; 然而当 N 较小时, Co-Transfer 的分类错误率随参数 D 的变化较大; 随着 N 的增加, Co-Transfer 的分类错误率对参数 D 的敏感性变小.

4 结 论

本文提出了一种半监督归纳迁移学习框架——Co-Transfer. 在 4 个 UCI 和文本分类任务数据集上与 7 个算法的对比实验表明: Co-Transfer 的这种双向同步迁移的机制可以有效地学习源域和目标域的标记和未标记样本来提升泛化性能. 我们未来的工作包括: 使用多种类型的组件分类器来评测 Co-Transfer 的分类性能, 如神经网络、朴素贝叶斯等; 源域和目标域之间的双向可迁移性对 Co-Transfer 的影响. 此外, 还应该使用更多的数据集, 尤其是实际应用中的数据集来评测 Co-Transfer 的泛化能力. 本文伪代码可以从 https://gitee.com/ymw12345/co-transfer.git 下载.

作者贡献声明: 文益民负责算法研讨、算法设计及论文修改; 员喆负责算法设计、论文初稿撰写及论文修改; 余航参与算法研讨、论文修改.

参 考 文 献

[1] Pan S, Yang Qiang. A survey on transfer learning[J]. IEEE Transactions on Knowledge and Data Engineering, 2010, 22(10): 1345−1359

[2] Zhuang Fuzhen, Qi Zhiyuan, Duan Keyu, et al. A comprehensive survey on transfer learning[J]. Proceedings of the IEEE, 2020, 109(1): 43−76

[3] Liu Xiaobo, Zhang H, Cai Zhihua, et al. A Tri-training based transfer learning algorithm[C]//Proc of the 24th Int Conf on Tools with Artificial Intelligence. Piscataway. NJ: IEEE, 2012: 698−703

[4] Tang Yejun, Wu Bin, Peng Liangrui, et al. Semi-supervised transfer learning for convolution neural network based Chinese character recognition[C]//Proc of the 14th IAPR Int Conf on Document Analysis and Recognition (ICDAR). Piscataway, NJ: IEEE, 2017: 441−447

[5] Abuduweili A, Li Xingjian, Shi H, et al. Adaptive consistency regularization for semi-supervised transfer learning[C]//Proc of the 34th IEEE/CVF Conf on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2021: 6923−6932

[6] Wei Wei, Meng Deyu, Zhao Qian, et al. Semi-supervised transfer

learning for image rain removal[C]//Proc of the 32nd IEEE/CVF Conf on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2019: 3877−3886

[7] Chebli A, Djebbar A, Marouani H. Semi-supervised learning for medical application: A survey[C]//Proc of the 9th Int Conf on Applied Smart Systems (ICASS). Piscataway, NJ: IEEE, 2018: 1−9

[8] Mohanasundaram R, Malhotra A, Arun R, et al. Deep Learning and Semi-supervised and Transfer Learning Algorithms for Medical Imaging[M]// Deep Learning and Parallel Computing Environment for Bioengineering Systems. New York: Academic Press, 2019: 139−151

[9] Liu Quande, Yu Lequan, Luo Luyang, et al. Semi-supervised medical image classification with relation-driven self-ensembling model[J]. IEEE Transactions on Medical Imaging, 2020, 39(11): 3429−3440

[10] Liu Qinhua, Liao Xuejun, Li Hui, et al. Semi-supervised multitask learning[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, 31(6): 1074−1086

[11] Skolidis G, Sanguinetti G. Semisupervised multitask learning with Gaussian processes[J]. IEEE Transactions on Neural Networks and Learning Systems, 2013, 24(12): 2101−2112

[12] Naji K, Ulas B. Semi-supervised multi-task learning for lung cancer diagnosis[C]//Proc of the 40th Annual Int Conf of the IEEE Engineering in Medicine and Biology Society. Piscataway, NJ: IEEE, 2018: 710−713

[13] Chu Xu, Lin Yang, Wang Yasha, et al. Mlrda: A multi-task semi-supervised learning framework for drug-drug interaction prediction[C]//Proc of the 28th Int Joint Conf on Artificial Intelligence. New York: ACM, 2019: 4518−4524

[14] Qi Qi, Wang Xiaolu, Sun Haifeng, et al. A novel multi-task learning framework for semi-supervised semantic parsing[J]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2020, 28: 2552−2560

[15] Van Engelen J, Hoos H. A survey on semi-supervised learning[J]. Machine Learning, 2020, 109(2): 373−440

[16] Xu Mengfan, Li Xinghua, Liu Hai, et al. An intrusion detection scheme based on semi-supervised learning and information gain ratio[J]. Journal of Computer Research and Development, 2017, 54(10): 2255−2267 (in Chinese)
(许勐璠, 李兴华, 刘海, 等. 基于半监督学习和信息增益率的入侵检测方案[J]. 计算机研究与发展, 2017, 54(10): 2255−2267)

[17] Dai Wenyuan, Yang Qiang, Xue Guirong, et al. Boosting for transfer learning[C]//Proc of the 24th Int Conf on Machine Learning (ICML '07). New York: ACM, 2007: 193–200

[18] Zhou Zhihua, Li Ming. Tri-training: Exploiting unlabeled data using three classifiers[J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(11): 1529−1541

[19] Bache K, Lichman M. UCI machine learning repository [DB/OL]. 2013[2022-06-08].https://archive.ics.uci.edu/ml/index.php

[20] Scrucca L. A fast and efficient modal EM algorithm for Gaussian mixtures[J]. The ASA Data Science Journal:Statistical Analysis and Data Mining, 2021, 14(4): 305−314

[21] Li Ming, Zhou Zhihua. Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples[J]. IEEE Transactions on Systems, Man, and Cybernetics-Part A:Systems and Humans, 2007, 37(6): 1088−1098

[22] Cervantes J, Garcia-Lamont F, Rodriguez-Mazahua L, et al. A comprehensive survey on support vector machine classification: Applications, challenges and trends[J]. Neurocomputing, 2020, 408: 189−215

[23] Shimomura L, Oyamada R, Vieira M et al. A survey on graph-based methods for similarity searches in metric spaces[J]. Information Systems, 2021, 95: 101507

[24] Blum A, Mitchell T. Combining labeled and unlabeled data with co-training[C]//Proc of the 8th Annual Conf on Computational Learning Theory. New York: ACM, 1998: 92−100

[25] Dasgupta S, Littman M, McAllester D. PAC generalization bounds for co-training[C]//Proc of the 14th Int Conf on Neural Information Processing Systems. Cambridge, MA: MIT Press, 2002: 375−382

[26] Triguero I, García S, Herrera F. Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study[J]. Knowledge and Information Systems, 2015, 42(2): 245−284

[27] Kamishima T, Hamasaki M, Akaho S. TrBagg: A simple transfer learning method and its application to personalization in collaborative tagging[C]//Proc of the 9th IEEE Int Conf on Data Mining. Piscataway, NJ: IEEE, 2009: 219−228

[28] Shi Yuan, Lan Zhenzhong, Liu Wei, et al. Extending semi-supervised learning methods for inductive transfer learning[C]//Proc of the 9th IEEE Int Conf on Data Mining. Piscataway, NJ: IEEE, 2009: 483−492

[29] Liu Zhuang, Liu Chang, Wayne L, et al. Pretraining financial language model with multi-task learning for financial text mining[J]. Journal of Computer Research and Development, 2021, 58(8): 1761−1772 (in Chinese)
(刘壮, 刘畅, Wayne L, 等. 用于金融文本挖掘的多任务学习预训练金融语言模型[J]. 计算机研究与发展, 2021, 58(8): 1761−1772)

Wen Yimin, born in 1969. PhD, professor, PhD supervisor. Distinguished member of CCF. His main research interests include machine learning, recommendation systems, and big data analysis.

文益民, 1969 年生. 博士, 教授, 博士生导师. CCF 杰出会员. 主要研究方向为机器学习、推荐系统和大数据分析.

Yuan Zhe, born in 1995. Master candidate. His main research interests include machine learning and data mining.

员 喆, 1995 年生. 硕士研究生. 主要研究方向为机器学习和数据挖掘.

Yu Hang, born in 1991. PhD, professor, PhD supervisor. Member of IEEE and CCF. His main research interests include online machine learning, knowledge graph and multi-agent system.

余 航, 1991 年生. 博士, 教授, 博士生导师. IEEE 和 CCF 会员. 主要研究方向为在线机器学习、知识图谱和多智能体系统.