TransBoost: A Boosting-Tree Kernel Transfer Learning Algorithm for Improving Financial Inclusion

Yiheng Sun,[1] **Tian Lu,**[2*] **Cong Wang,**[3*] **Yuan Li,**[†] **Huaiyu Fu,**[4] **Jingran Dong,**[1] **Yunjie Xu**[4]
[1] Tencent Weixin Group
[2] Heinz College, Carnegie Mellon University
[3] Guanghua School of Management, Peking University
[4] School of Management, Fudan University
elisun@tencent.com, lutiansteven@gmail.com, wangcong@gsm.pku.edu.cn, Liyuanlp@google.com, hyfu20@fudan.edu.cn, jingrandong@tencent.com, yunjiexu@fudan.edu.cn

Abstract

The prosperity of mobile and financial technologies has bred and expanded various kinds of financial products to a broader scope of people, which contributes to financial inclusion. It brings non-trivial social benefits of diminishing financial inequality. However, the technical challenges in individual financial risk evaluation exacerbated by the unforeseen user characteristic distribution and limited credit history of new users, as well as the inexperience of newly-entered companies in handling complex data and obtaining accurate labels, impede further promotion of financial inclusion. To tackle these challenges, this paper develops a novel transfer learning algorithm (i.e., *TransBoost*) that combines the merits of tree-based models and kernel methods[1]. The TransBoost is designed with a parallel tree structure and efficient weights updating mechanism with theoretical guarantee, which enables it to excel in tackling real-world data with high dimensional features and sparsity in $O(n)$ time complexity. We conduct extensive experiments on two public datasets and a unique large-scale dataset from Tencent Mobile Payment. The results show that the TransBoost outperforms other state-of-the-art benchmark transfer learning algorithms in terms of prediction accuracy with superior efficiency, demonstrate stronger robustness to data sparsity, and provide meaningful model interpretation. Besides, given a financial risk level, the TransBoost enables financial service providers to serve the largest number of users including those who would otherwise be excluded by other algorithms. That is, the TransBoost improves financial inclusion.

Introduction

The rapid growth of mobile and financial technologies has bred various kinds of financial products such as microloan, consumer debt, and internet insurance (Teja 2017). Many companies have joined the boat and explored ways of offering novel financial products (Musto et al. 2015). These emerging financial products have dramatically expanded financial services to a broader range of people, better satisfying their financial needs (Guild 2017). It greatly advocates *financial inclusion*. Realizing financial inclusion has significant social benefits. Approximately two billion people do not have a basic bank account till 2017 (World-Bank 2018). Advancing financial inclusion can help reduce poverty by helping people invest in the future, smooth their consumption, and manage income shocks and financial risks, and thus increase sustainability of the entire society (Dev 2006). In the highly competitive financial markets, financial service providers are also incentivized to pursue more users. The consumer pool expansion is critical to improve their commercial sustainability (Demirgüç-Kunt and Singer 2017).

However, attenting to the pool of underserved users causes challenges in financial risk evaluation. The first challenge concerns the users who have limited credit history records, resulting in the lack of useful information for model training. This problem is severer in developing countries or regions that do not have mature social credit system. Another challenge is that new users might demonstrate different characteristic distributions from the existing ones (Bassem 2012). The patterns learned from the existing users may have limited generalizability and lead to poor predictive performance for new news. This problem is more serious for new financial service providers who are inexperienced in obtaining sufficient and accurate financial risk labels for model training and in handling complex large-scale user data.

Transfer learning techniques emerge as useful tools for improving financial risk evaluation. Transfer learning deals with how systems can quickly adapt themselves to new situations, tasks, and environments by transferring knowledge from a related task learned (Yang et al. 2020). By leveraging transfer learning and the user datasets with financial risk labels (which usually come from users of existing products or data of similar contexts), financial practitioners can improve financial risk assessment accuracy for the nascent products for financial inclusion.

State-of-the-art transfer learning algorithms include kernel-based algorithms such as Kernel Mean Matching (KMM) (Huang et al. 2006), JDA (Long et al. 2013), and deep learning algorithms such as DAAN (Yu et al. 2019),

[*]corresponding author
[†]This project was done while the author was at Google.

[1]Our implementation can be found here.

DDAN (Wang et al. 2020), and DSAN (Zhu et al. 2020). Although the kernel-based algorithms have sound theoretical properties, these algorithms are usually computationally expensive, typically at least $O(n\sqrt{n})$ time (Alessandro Rudi 2017). The deep learning algorithms have succeeded in visual, audio and natural language processing tasks. However, these algorithms lack model interpretability (Jung, Bollmann, and Lee 2020).

Another challenge in financial risk evaluation is the sparsity of the large-scale dataset. Missing values and outliers are prevalent in real-world applications. Although some of the above-mentioned techniques can tackle empty value and outlier issues, most of them are costly and ineffective in handling complex business settings with highly dimensional features. The shortcomings lead to a low adoption rate of these algorithms in financial industry. By contrast, *tree-*based algorithms have shown desired properties in *effectiveness* and *efficiency* for financial risk evaluation. Especially, tree-based algorithms could outperform deep learning algorithms when being applied to structured dataset without underlying temporal or spatial interactions among features (Chen and Guestrin 2016). Besides, because tree-based algorithms grow in a binary split approach, they are naturally *robust to sparse and noisy input* (Friedman 2001). Finally, due to rule-based model structure, tree-based algorithms are usually more *interpretable* than deep learning algorithms and even linear algorithms (Lundberg et al. 2020).

We propose the *TransBoost*, a novel transfer learning algorithm that combines the merits of tree-based models and kernel methods. Through a specially designed parallel boosting-tree framework, the algorithm can train the transfer learning classifier in only $O(n)$ time while outperforming state-of-the-art algorithms in prediction accuracy. We have proved that the algorithm is theoretically a generalization of KMM framework with boosting trees being a more flexible kernel method.

The contributions of this study are multi-fold:

1. To the best of our knowledge, we are the first to demonstrate the value of transfer learning for financial inclusion. We showcase how practitioners can leverage (open-source) existing dataset with labels to learn individual financial risk for a new domain, and thus expand financial products and services to a broader range of people.
2. Given that state-of-the-art transfer learning algorithms are either computationally expensive or lack of flexibility, we take advantage of tree-based and kernel methods, and develop an innovative boosting-tree transfer learning algorithm as a more flexible kernel trick for KMM. Theoretically and empirically, we show it is an efficient algorithm that yields KMM weights in $O(n)$ time.
3. Our proposed boosting-tree kernel algorithm not only outperforms other algorithms in terms of prediction accuracy, but also is robust in handling noisy and high-dimensional data with superior efficiency, and yields meaningful model interpretation. This is non-trivial for practice because data sparsity, noise, and lack of interpretability have restrained many advanced algorithms from being applied for financial risk assessment in real-world business.

Related Work

Financial Risk Evaluation and Financial Inclusion

Scholars have made great strides in improving financial risk evaluation accuracy from data and algorithm perspectives. Soto et al. (2021) did a complete summary.

It is important for modern financial companies to pursue financial inclusion. Financial inclusion refers to the delivery of financial services at affordable costs to vast sections of disadvantaged and low-income groups (Dev 2006). Academic studies, government white books, and commercial reports have amply documented the importance of financial inclusion. Karlan and Zinman (2010) and Cull, Ehrbeck, and Holle (2014) summarized the benefits of financial inclusion and Klapper, El-Zoghbi, and Hess (2016) reviewed how financial inclusion can help achieve the Sustainable Development Goals. Recently, financial companies have collected data from nontraditional sources to help assess the creditworthiness of individual users and small business owners of ”thin-file” (Óskarsdóttir et al. 2020). However, this approach heavily relies on the availability of user data, which is always difficult and costly to obtain for most financial companies.

Transfer Learning

A vital task for transfer learning is to minimize the distribution discrepancy across domains. Various transfer learning approaches have been proposed to fully leverage the known information, which can be categorized into two types: instance weighting and feature representation. Instance weighting methods estimate and assign different weights to known instances to mimic the target domain. For example, KMM approach estimates the weights via minimizing the mean across the source and target domains in a reproducing kernel Hilbert space (RKHS) (Huang et al. 2006). Tradaboost follows the design principle of AdaBoost ensemble method to iteratively update the weights of training data (Dai et al. 2007). Feature representation methods seek to find a common feature representation between the source and target domains to achieve knowledge transfer. Such common representation is usually carried out by mapping the source and target domain to another space, including transfer component analysis (TCA) (Pan et al. 2010), joint distribution adaptation (JDA) (Long et al. 2013), balanced distribution adaptation (BDA) (Wang et al. 2017), etc., or aligning one to the other, e.g., Correlation Alignment (CORAL) (Sun, Feng, and Saenko 2016). Some of these approaches have been proposed to solve complex problems in natural language processing as well as computer vision (Zhu et al. 2020; Wang et al. 2020).

Gradient Boosting Decision Tree (GBDT)

GBDT (Friedman 2001) is a widely used ensemble tree model framework. There are some successful implementations of GBDT in academia and industry, including XGBoost (Chen and Guestrin 2016), LightGBM (Ke et al. 2017) and CATBoost (Dorogush, Ershov, and Gulin 2018). In brief, GBDT is a model with *M* additive regression trees as its output. The model is trained iteratively. In each iteration, the tree is optimized by the first-order (GBDT) or second-order (XGBoost) Taylor approximation of the regularized objective function. Formally, given a dataset $D = \{(x, y)\}$, the model is defined as

$$h(x) = \sum_{k}^{M} T_k\left(x, \Theta_k\right),$$

where $h(x)$ is GBDT and M is the total number of trees. T_k is the tree of the *k*-th iteration and Θ_k is the parameter. For the output of every tree, $T_k\left(x, \Theta_k\right) = \{q(x), w(q(x))\}$, where $q(x)$ is the tree structure function mapping instance x to a certain leaf j with $w(j)$ as its weight.

TransBoost Learning Algorithm

A common transfer learning scenario in industry involves a large-sized labeled source domain from a matured product and a limited-sized labeled target domain from a newly launched product. The target of the TransBoost algorithm is to leverage the instances from matured product to improve model performance on target domain. Formally, assuming that $\mathcal{D} = \left\{\mathcal{D}^S, \mathcal{D}^T\right\}$ is a dataset of source domain and target domain. $\mathcal{D}^S = \left\{(x, y) \mid \left(x_i^S, y_i^S\right) \in P^S, i = 1, \ldots, N_S\right\}$ denotes n_S instances from source domain and $\mathcal{D}^T = \left\{(x, y) \mid \left(x_j^T, y_j^T\right) \in P^T, j = 1, \ldots, N_T\right\}$ denotes n_T instances from target domain with m features ($x \in \mathbb{R}^m, y \in \mathbb{R}$). Our goal is to learn a classifier $h_t(x) = P\left(Y^T \mid X^T\right)$ for target domain.

Next, we first define the learning objective of the Transboost as a regularized sum of loss on target domain and weighted source domain. We then purpose our algorithm — an innovative transfer learning framework of two parallel boosting-tree models that share identical tree structures but different node weights. The special design keeps the merit of robustness and interpretability of tree-based models. Besides, it enables us to train classifier and adjust for distributional differences between domains simultaneously in $O(n)$ complexity, which is much more efficient than traditional kernel method of $O(n\sqrt{n})$ complexity. Finally, we prove that our algorithm is in nature a generalization of the KMM framework *but* using boosting trees as a more flexible kernel method. It also extends KMM to match joint distribution between domains instead of marginal distributions, which potentially improves accuracy in classification problems.

Transfer Learning Objective

We start with a simple learning problem. Given sufficient instances from the target domain, we can directly optimize h_t with the following objective:

$$\begin{aligned} \mathcal{L}_T\left(h_t\right) &= \mathbb{E}_{(x,y)\sim P^T}\left[\mathcal{L}\left(x, y; h_t\right)\right] \\ &= \frac{1}{n_T}\sum_{j}^{n_T}\left[\mathcal{L}\left(x_j^T, y_j^T; h_t\right)\right], \end{aligned} \tag{1}$$

where $\mathcal{L}_T(h)$ is the expected loss of classifier h_t over instances on target domain. $\mathcal{L}$ is cross-entropy loss. However, due to the scarcity of instances from the target domain and the curse of high dimensionality, directly optimizing classifier h_t often leads to poor prediction performance. To alleviate the issue, transfer learning methods could be used to utilize the information from a source domain as prior knowledge to improve the model performance on the target domain. However, applying the model trained from the source domain may *not* work well because, in many real-world applications, the source domain and target domains are still different. To address that, a common strategy is to assign weights to instances from source domain in the loss function:

$$\begin{aligned} \mathcal{L}_T\left(h_t\right) &= \mathbb{E}_{(x,y)\sim P^S}\left[\frac{P^T(x,y)}{P^S(x,y)}\mathcal{L}\left(x, y; h_t\right)\right] \\ &\approx \frac{1}{n_s}\sum_{i}^{n_S}\left[\beta_i\mathcal{L}\left(x_i^S, y_i^S; h_t\right)\right] \\ &= \mathcal{L}_S\left(h_t, \beta\right). \end{aligned} \tag{2}$$

Unlike Eq. (1), the loss is estimated with re-weighted instances from the source domain and the weights $\beta_i = \frac{P^T\left(x_i^S, y_i^S\right)}{P^S\left(x_i^S, y_i^S\right)}$ is the ratio between the joint distribution $P^T(x, y)$ and $P^S(x, y)$. We then have

$$\frac{P^T(x,y)}{P^S(x,y)} = \frac{P^T(x)}{P^S(x)} * \frac{P^T(y \mid x)}{P^S(y \mid x)},$$

where we split the distribution ratio into two parts - marginal distribution ratio $P^T(x)/P^S(x)$ and conditional distribution ratio $P^T(y \mid x)/P^S(y \mid x)$. Later, we will show how to estimate these two parts with boosted tree models.

In practice, the overall objective of TransBoost is defined on both the reweighted source domain and the target domain so that target-specific feature weight can also be learnt during the training process:

$$\mathcal{L}\left(h_t\right) = \lambda\mathcal{L}_S\left(h_t, \beta\right) + \mathcal{L}_T\left(h_t\right) + \Omega\left(h_t\right), \tag{3}$$

where β is the weights to match the joint distribution, λ is the hyper-parameter to balance L_T and L_S , and Ω is the regularizing term for h_t.

Learning Algorithm

In this section, we discuss the TransBoost in details. The overall architecture of TransBoost is presented in Figure 1. The TransBoost model consists of two parallel GBDTs, a main boosting tree and an ancillary boosting tree, for the target domain and the source domain respetively. The final model is learnt by the main boosting tree which still enjoys effectiveness, interpretability and robustness. At each iteration in the TransBoost, there are two key steps:

Same Structure Tree Boosting. At the *k*-th iteration, the added new tree for the target domain is trained to optimize

$$\mathcal{L}\left(h_t\right) = \lambda\mathcal{L}_S\left(h_t, \beta^k\right) + \mathcal{L}_T\left(h_t\right) + \Omega\left(h_t\right),$$

while the added new tree for the source domain is trained to optimize

$$\mathcal{L}\left(h_s\right) = \mathcal{L}_S\left(h_s\right) + \Omega\left(h_s\right).$$

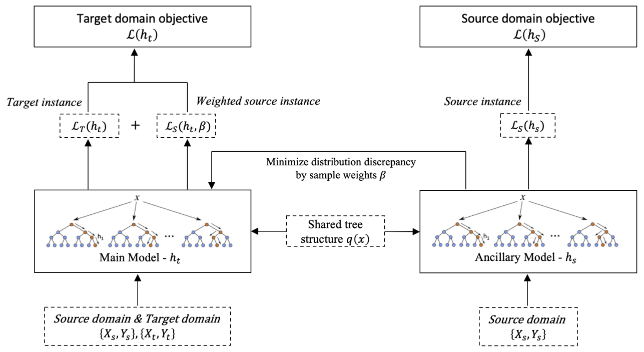

Figure 1: The TransBoost learning algorithm.

However, although the tree trained on the source domain is constraint to have the identical base tree structures and split values as the tree trained for the target domain, it is allowed to have different node weights. At the high level, the identical base tree structures will induce the same partition on both source domain and target domain, which ensures matched marginal distributions in the learning process. On the other hand, different weights provide flexibility to learn different conditional distributions.

Efficient Weights Update. A major drawback of traditional kernel-trick-based method to reweight source domain is the computational cost of solving the quadratic optimization problem, typically at least in $O(n\sqrt{n})$ time. In our algorithm, the same-structure GBDTs enable to apply boosting-tree as an innovative kernel method. We only use the k-th tree to construct the kernel without solving the quadratic optimization problem. Thus an analytical solution to the reweights could be derived in $O(n)$ time. More specifically, after adding boosting tree at the k-th iteration, we can then update the weights in the objective function as

$$\beta_i^{k+1} = \frac{n_{I_{q(x_i)}} N_S}{m_{I_{q(x_i)}} N_T} \frac{y_i h_t^k(x_i) + (1 - y_i)\left(1 - h_t^k(x_i)\right)}{y_i h_s^k(x_i) + (1 - y_i)\left(1 - h_s^k(x_i)\right)}, \tag{4}$$

where $I_{q(x_i)}$ as the instance set of leaf that the source domain instance x_i belongs to. $n_{I_{q(x_i)}}$ is the number of target instances from $I_{q(x_i)}$ and $m_{I_{q(x_i)}}$ is the number of source instances from $I_{q(x_i)}$. $h_t^k(x_i)_s$ is the estimation of x_i from the main model for the target domain at the k-the iteration while $h_s^k(x_i)$ is the estimation of x_i from the ancillary model for the source domain. The TransBoost algorithm proceeds as the *Algorithm 1*.

Theoretical Analysis

In high-dimensional data, $\frac{P^T(x,y)}{P^S(x,y)}$ is difficult to estimate (Fuzhen Zhuang 2020). KMM (Huang et al. 2006) has been used to correct sampling bias from marginal distribution for unlabelled data. In our work, we generalize and customize the KMM framework in three-ways: first, we adopt boosting trees as kernels. Compared to traditional kernels, the kernel is derived from the training data hence more flexible (Alex Davies 2014; Scornet 2015; Breiman 2000); second, we only use the last added tree to construct the tree kernel so that the instance weights could be calculated efficiently; third, for the binary classification problem, we extend KMM to match joint distributions instead of marginal distributions.

Algorithm 1 TransBoost Algorithm

```
Input: Source and target features $\{X_s, X_t\}$ and labels
       $\{Y_s, Y_t\}$
Parameter: Number of estimator M, balance factor $\lambda$,
       regularization parameter $\Omega$
1: $h_t \leftarrow \{\}$ # main model; $h_s \leftarrow \{\}$ # ancillary model
2: $\beta^0 \leftarrow 1$ # source domain weight
3: for $k = 1$ to $M$ do:
4:     Train a base tree classifer $T_k^t = \{q_k^t(x), w_k^t\}$ to
       main model by minimizing Tylor's approximation
       of objective $L(h_t)$
5:     Add a base tree classifer $T_k^s = \{q_k^s(x), w_k^s\}$ to ancillary model and set $q_k^s(x) \leftarrow q_k^t(x)$ # shared tree
       structure
6:     Compute $w_k^s$ by minimizing Tylor's approximation of objective $L(h_s)$
7:     Update the prediction by main model $\hat{h}_t^k(X_s)$
8:     Update the prediction by ancillary model $\hat{h}_s^k(X_s)$
9:     Update source domain weight $\beta^{i+1}$ using Eq. (4).
Output: Main model $h_t$ (i.e., the classifier for the target
       domain)
```

For a tree with tree structure function $q(x)$, the tree kernel is defined as the connection function of the tree, $K(x,y) = \mathbb{1}_{q(x)=q(y)}$.

Proposition 1 *When using tree kernels, The best sampling weights adjustment for the marginal distribution in KMM is* $\beta_i^{marginal} = \frac{n_{I_{q(x_i)}} N_S}{m_{I_{q(x_i)}} N_T}$, *where* $m_{I_{q(x)}}$ *is the number of source instances from* $I_{q(x)}$ *and* $n_{I_{q(x)}}$ *is the number of target instances from* $I_{q(x)}$, N_S *is the total number of samples in the source domain and* N_T *is the total number of samples in the target domain.*

Proof. We can first order the nodes in the source domain by their leaves and then $K(x_i, x_j)$ over the source domain will be a block diagonal matrix

$$K = \begin{bmatrix} J_{m_{I_1}} & 0 & \cdots & 0 \\ 0 & J_{m_{I_2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_{m_{I_T}} \end{bmatrix},$$

where $J_{m_{I_t}}$ is a (m_{I_t} by m_{I_t}) matrix of ones. Note that this kernel function is positive semi-definite. Then we can follow (Huang et al. 2006) to find the best ratio estimations by optimizing β_i via

$$\underset{\beta}{minimize} \ \frac{1}{2}\beta^T K \beta - k^T \beta$$

where $K_{i,j}$ is the above block diagonal matrix and $k_i = \frac{N_S}{N_T}\sum_{j=1}^{n_T} K\left(x_i, x_j'\right)$, N_s is the number of elements from the source domain and N_t is the number of elements from the target domain, x_i is data from the source domain and x'_j is data from the target domain.

To derive the optimal weights β_i, let's rearrange the objective function by leaves. Noticing that when x_i, x_j are from the same leaf, they have the same weight $\beta_{I_{q(x)}}$, we'll have

$$\frac{1}{2}\beta^T K\beta - k^T\beta = \frac{1}{2}\sum_{j=1}^{T} m_{I_j}^2 \beta_{I_j}^2 - \sum_{j=1}^{T} \frac{m_{I_j} n_{I_j} N_S}{N_T}\beta_{I_j}. \quad (5)$$

It's easy to show that these are just T quadratic functions and the optimal weight for leaf t is $\frac{n_{I_t} N_S}{m_{I_t} N_T}$. $\square$

Proposition 2 *For binary classification, the ratio of conditional distribution* $\frac{P^T(y_i|x_i)}{P^S(y_i|x_i)}$ can be approximated by $\beta_i^{conditional} = \frac{y_i h_t(x_i) + (1-y_i)(1-h_t(x_i))}{y_i h_s(x_i) + (1-y_i)(1-hs(x_i))}$, where $h(x_i) = {}^p(y_i = 1|x_i)$.

We train two parallel GBDTs to fit both $h_t(x)$ and $h_s(x)$ and when the fitting is reasonably good, we could use them to approximate the conditional probability.

With these two propositions, we can proceed handle marginal distributions and conditional distributions.

Proposition 3 *The optimal weights that match the joint distribution is* $\beta_i = \beta_i^{marginal} * \beta_i^{conditional}$.

Proof. To generalize the KMM approach to matching joint distribution for binary classification problem, we define a new kernel-like function based on the above tree connection function. More specifically, $K_{new}([x_i, y_i], [x_j, y_j]) = \mathbb{1}_{q(x_i)=q(x_j)} * \mathbb{1}_{y_i=y_j}$

$$K_{new} = \begin{bmatrix} J_{m_{I_1},1} & 0 & \cdots & 0 & 0 \\ 0 & J_{m_{I_1},0} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & J_{m_{I_T},1} & 0 \\ 0 & 0 & \cdots & 0 & J_{m_{I_T},0} \end{bmatrix},$$

where $J_{m_{I_t},l}$ is a ($m_{I_t} * P^S_{I_k,l}$ by $m_{I_t} * P^S_{I_k,l}$) matrix of ones and $P^S_{I_k,l} = P^S(y = l|I_k)$.

Then we can rearrange the objective function by leaves and labels. The objective function takes the following form:

$$\frac{1}{2}\beta^T K\beta - k^T\beta = \frac{1}{2}\sum_{j=1}^{T}\sum_{l=0}^{1}\left(m_{I_j} p^S_{I_j,l}\right)^2 \beta^2_{I_{j,l}} - \sum_{j=1}^{T}\sum_{l=0}^{1}\frac{m_{I_j} P^S_{I_j,l} n_{I_j} P^T_{I_j,l} N_S}{N_T}\beta_{j\,I_j,l},$$

and the analytical solution is

$$\beta_i = \frac{n_{I_{q(x_i)}} N_S}{m_{I_{q(x_i)}} N_T}\frac{p^T_{I_{q(x_i)},l(x_i)}}{p^S_{I_{q(x_i)},l(x_i)}}.$$

by proposition 2, we can estimate the ratio of conditional distributions by $\frac{y_i h_t(x_i) + (1-y_i)(1-h_t(x_i))}{y_i h_s(x_i) + (1-y_i)(1-hs(x_i))}$ and the optimal weight is

$$\begin{aligned}\beta_i &= \beta_i^{marginal} * \beta_i^{conditional} \\ &= \frac{n_{I_{q(x_i)}} N_S}{m_{I_{q(x_i)}} N_T}\frac{y_i h_t(x_i) + (1-y_i)(1-h_t(x_i))}{y_i h_s(x_i) + (1-y_i)(1-hs(x_i))}.\end{aligned} \quad (6)$$

$\square$

Algorithm Limitations. Finally, because the TransBoost is a *domain-knowledge-independent* algorithm and the algorithm is *only* based on local smoothness assumption, there would be cases when the framework may fail. We elaborate on three different situations that can be addressed in future studies:

- *When data is generated from the composition of features*: In this case, there could be non-local features shared between source domain and target domain (e.g., spatial features in image and temporal features in text), and the boosting-tree framework may not work well.
- *When domain knowledge plays an important role*: from the regularization perspective, domain knowledge could be modeled as a special regularization term. Because our algorithm only applies local smoothness regularization, in this case, it may have sub-optimal performance.
- *When source and target domains are irrelevant*: in this case, the conditional weight update, $\frac{y_i h_t(x_i) + (1-y_i)(1-h_t(x_i))}{y_i h_s(x_i) + (1-y_i)(1-hs(x_i))}$, will quickly detect that the tree-structure from source domain is less relevant and thus increase the weights of the target domain. As the weights of the source domain decay, the performance of the algorithm will be very comparable to the model that only uses target domain as input.

Experiments

Datasets

We adopt three datasets to evaluate the TransBoost against several baseline methods. The *LendingClub* and *Wine Quality* are public datasets with small sample sizes. They are used to compare the TransBoost with the traditional kernel-related methods that have high computational overhead characteristics. The *Tencent Mobile Payment* dataset is a private large-scale dataset used to showcase the application of our method and compare model performance in the real-world industrial environment.

Tencent Mobile Payment Dataset. We obtain a unique real-world financial fraudulent detection dataset from the Tencent Mobile Payment.[2] The dataset is sampled from two

[2]The dataset in this paper is properly sampled only for testing purpose and does not imply any commercial information. All private information of users is removed from the dataset. Besides, the experiment was conducted locally on Tencent's server by formal employees who strictly followed data protection regulations.

business scenarios — a mature financial product (source domain) with 1.49 million users and a newly-launched product(target domain) with 85,133 users. Each user has 2,714 features and is labeled as fraudulent (i.e., positive) or not (i.e., negative). We randomly choose 44.4% of users from the target domain for testing. The data description is shown in Table 1.

Domain	#Total	Positive Rate
Source for Training	1,487,904	4.99 %
Target for Training	49,669	4.85 %
Target for Testing	39,735	4.69 %

Table 1: Description of Tencent mobile payment dataset.

LendingClub Dataset. LendingClub dataset is an open-source micro-lending dataset from LendingClub (Kaggle@Yash 2020). The dataset includes loans for different purposes in many years. To increase the distribution difference, we set the medical loans in 2015 as the source domain and the car loans in 2016Q1 as the target domain. The features include 110 borrower and loan characteristics such as borrower's income, loan's interest rate, etc. The labels are given by loans' final status: fully paid or charged off. We use 5,080 labeled source instances and 1,000 labeled target instances for training, and 3,000 target instances for testing.

Wine Quality Dataset. Wine quality dataset is a popular dataset from UCI Machine Learning Repository (Dua and Graff 2017) with two datasets related to red and white wine samples. We set the red wine as the source domain and white wine as the target domain. There are 11 physical and chemical features of wine in the dataset, and the binary outcome label is whether the quality score is higher than 5. We use 3,918 labeled source instances and 500 labeled target instances for training, and 960 target instances for testing.

Algorithms for Comparison

We compared our proposed TransBoost with two types of state-of-the-art transfer learning methods: the traditional and deep transfer learning methods.

- Traditional transfer learning methods:

[1] KMM: KMM (Huang et al. 2006) estimates the source instances weight by matching the means in RKHS to adapt marginal distribution.

[2] TrAdaBoost: TrAdaBoost (Dai et al. 2007) extends AdaBoost to assign larger training weight to same distribution samples.

[3] JDA: JDA (Long et al. 2013) minimizes both the marginal and the conditional distribution difference in a low-dimensional feature space with fixed balance factor.

[4] CORAL: CORAL (Sun, Feng, and Saenko 2016) aligns the second-order statistic features by constructing the transformation matrix.

[5] BDA: BDA (Wang et al. 2017) minimizes both the marginal and the conditional distribution difference in a low-dimensional feature space with dynamic balance factor.

- Deep transfer learning methods:

[6] DAAN: DAAN (Yu et al. 2019) is a deep adversarial network that dynamically learns the domain-invariant representations by evaluating the importance of global and local distribution.

[7] DDAN: DDAN (Wang et al. 2020), also named as DeepMEDA, is a deep network that quantitatively evaluates the importance of marginal and conditional distribution.

[8] DSAN: DSAN (Zhu et al. 2020) is a deep transfer network that aligns the relevant subdomain distributions based on local maximum mean discrepancy.

Implementation Details

We implement the TransBoost algorithm based on the source code of XGBoost. For fair comparison, we also take XGBoost as the base classifier for traditional benchmarks [1—5]. In the LendingClub and Wine quality datasets, we set the number of estimators (i.e., booster) to 40 and tree depth to 4 for TransBoost and benchmark algorithms. In the Tencent mobile payment dataset, we set the number of estimator = 400 and tree depth = 4. We use standard one-layer Multi-layer perceptron in PyTorch for deep transfer learning benchmarks [6—8] , which is sufficient to solve the classification problems in our experiments (Shen et al. 2018). The learning rate is set to 0.01 and the number of epochs is set to 10.

Considering that some benchmarks [3, 5, 6, 7, 8] are designed to solve unsupervised domain adaptation problems, for fair comparison, we fine tune them to make full use of the labeled target domain instances. Specifically, we replace the pseudo label with true label in the domain adaption process and add the training of labeled target instances.

All experiments are conducted on the Tencent's server with 64-core CPU and 128G memory. We compare all eight benchmarks on the three datasets. Besides, we conduct sensitivity analysis to validate that TransBoost can outperform benchmark algorithms with different sizes of the target domain. To simulate the growth of newly launched product, the models are trained with all source domain samples and different fractions of the target domain samples. We use a fix-sized testing dataset to evaluate the model performance with AUC.

Results

Prediction Accuracy. The AUCs of TransBoost and other baseline methods on LendingClub, Wine quality, and Tencent mobile payment datasets are shown in Tables 2, 3, and Figure 2, respectively. Note that fewer results are shown in Figure 2 because some of the traditional benchmark models ([1, 3, 5, 7]) cannot handle the large volume and high sparsity of the Tencent mobile payment dataset. This further shows the scalability of our algorithm. All methods can achieve better prediction results with a larger size of the target domain. The TransBoost achieves rather stable prediction results even with small-sized target domain data. The TransBoost outperforms other baselines in almost all cases, showing its superiority in prediction.

	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Transboost	**0.7209**	**0.7253**	**0.7229**	**0.7238**	**0.7227**	**0.7276**	**0.7223**	**0.7213**	**0.7245**	**0.7204**
Tradaboost	0.7185	0.7059	0.7136	0.7047	0.7081	0.7164	0.7075	0.7068	0.7069	0.7071
BDA	0.6899	0.7035	0.7015	0.6908	0.7076	0.7016	0.6971	0.7058	0.7008	0.7041
JDA	0.7069	0.6913	0.6994	0.6889	0.7032	0.6979	0.7073	0.7087	0.7136	0.7061
CORAL	0.7107	0.7176	0.7089	0.7136	0.7066	0.7131	0.7152	0.7154	0.7154	0.7181
KMM	0.6995	0.7019	0.7044	0.7081	0.7077	0.7089	0.7147	0.7205	0.7195	0.7190
DSAN	-	-	-	0.6793	0.6978	0.7086	0.6931	0.6954	0.7001	0.6997
DAAN	-	-	-	0.6744	0.6909	0.7003	0.6976	0.7067	0.7074	0.7077
DeepMEDA	-	-	-	0.6745	0.7016	0.6999	0.7015	0.7014	0.7061	0.7084

Table 2: AUCs on LendingClub dataset output by different algorithms given a sample fraction of target domain for training.

	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Transboost	0.7821	**0.8114**	**0.8117**	**0.8268**	**0.8224**	**0.8225**	**0.8261**	**0.8254**	**0.8279**	0.8290
Tradaboost	0.7776	0.8023	0.8008	0.8039	0.8156	0.8093	0.8208	0.8240	0.8211	**0.8325**
BDA	**0.8080**	0.7873	0.7888	0.8117	0.8151	0.8005	0.8085	0.8187	0.8257	0.8191
JDA	0.7898	0.7914	0.7987	0.7977	0.8015	0.7993	0.8024	0.8003	0.8080	0.8112
CORAL	0.7777	0.7916	0.8105	0.8072	0.8119	0.8065	0.8158	0.8174	0.8207	0.8215
KMM	0.7765	0.7989	0.8013	0.7878	0.7918	0.8022	0.7962	0.7971	0.8052	0.8067
DSAN	0.7484	0.7892	0.8097	0.8103	0.8139	0.8092	0.8101	0.8123	0.8148	0.8133
DAAN	0.7724	0.7961	0.8112	0.8066	0.8008	0.8015	0.8008	0.8018	0.8053	0.8073
DeepMEDA	0.7661	0.7897	0.8117	0.8114	0.8117	0.8111	0.8093	0.8115	0.8127	0.8096

Table 3: AUCs on Wine quality dataset output by different algorithms given a sample fraction of target domain for training.

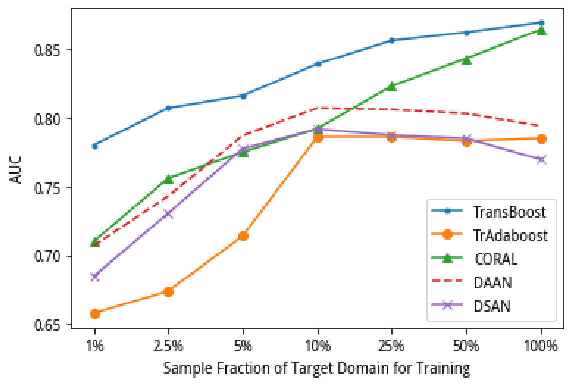

Figure 2: AUCs on Tencent mobile payment dataset output by different algorithms given a sample fraction of target domain for training.

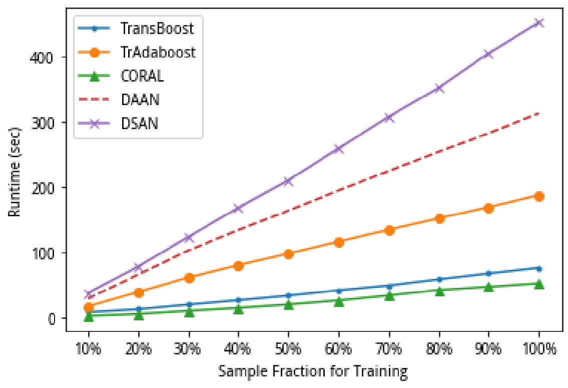

Figure 3: Runtime of different algorithms on Tencent mobile payment dataset with different training sample sizes.

Model Runtime. To further compare the efficiency of different models, we run each model 10 times with different fractions of the Tencent mobile payment dataset, and calculate the average runtime as shown in Figure 3 (again, several benchmarks fail to converge on this large-scale dataset). Overall, the TransBoost presents superior efficiency compared with most baseline models such as DSAN and DAAN. The runtime grows linearly to sample size. Considering both efficiency and prediction accuracy, we conclude that the TranBoost achieves high prediction accuracy without sacrificing efficiency, which fits perfectly for industrial application.

Robustness to Data Sparsity. The Tencent mobile payment dataset is sparse in nature with an overall non-missing rate of 26.8%. To validate that our algorithm is robust in more sparse contexts, we further lower the non-missing rate and compare the model performance. Specifically, we randomly set more features to *NULL* value in the training samples of Tencent dataset to simulate datasets with varied sparsity of [1% to 25%] and test the TransBoost against other baselines. All the experiments are carried out 10 times and the average performance is shown in Figure 4. TransBoost is robust enough and outperform all the baselines at various sparsity levels.

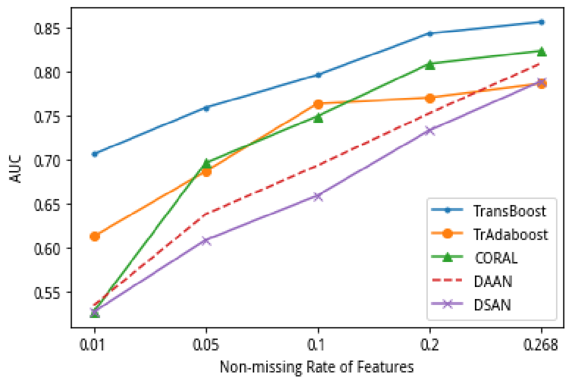

Figure 4: AUCs of different models under diverse data sparsity levels on Tencent mobile payment dataset.

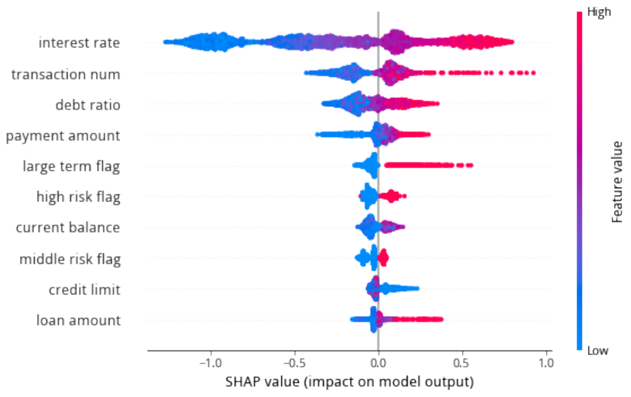

Figure 5: SHAP Value of top 10 important features for predicting financial risk output TransBoost model based on LendingClub dataset.

Model Interpretation. The tree-based TransBoost has the natural advantage in model interpretation. Figure 5 presents the Shapley value of important features (Lundberg and Lee 2017) for financial risk evaluation based on the LendingClub dataset. The figure offers meaningful insights by indicating that loan interest rate, transaction number, debt ratio, and payment amount, are significant for predicting loan default behavior. These findings are consistent with previous studies (e.g., Lu, Zhang, and Li (2020)).

Financial Inclusion. Finally, to examine whether the TransBoost can better advance finance inclusion than do other algorithms. Based on the LendingClub dataset,[3] we calculate the loan approval ratios under different algorithms given a default rate from 8% to 12%. Table 4 reveals that at all the given financial risk levels, the TransBoost yields highest loan approval ratio; that is, the TransBoost enables financial service providers to serve the largest number of users.

We further analyze two typical user features of the approved users. In specific, we calculate the average annual income and ratio of house renting for the approved users given the default rate of 8% and 10%. Table 5 indicates that generally, the users selected by TransBoost have lower income and higher proportion of house renting than do the users selected by other algorithms. It suggests that compared with other algorithms, the TransBoost not only offers financial service access to broader users but also covers users who would otherwise be excluded by other algorithms due to their less desirable background. Therefore, the TransBoost improves financial inclusion.

[3]We checked on all the three datasets and obtained consistent findings. Due to the Tencent data protection regulations, we do not report the results on its dataset.

	8%	9%	10%	11%	12%
TransBoost	**0.5423**	**0.6333**	**0.7477**	**0.8010**	**0.8567**
TrAdaBoost	0.4917	0.5607	0.6543	0.7523	0.8263
BDA	0.5043	0.6123	0.6613	0.7607	0.8280
JDA	0.5170	0.6210	0.6913	0.7617	0.8140
CORAL	0.4920	0.5933	0.6790	0.7640	0.8120
KMM	0.5403	0.6260	0.7087	0.7823	0.8460
DSAN	0.4513	0.5390	0.6040	0.6767	0.7457
DAAN	0.4047	0.5460	0.6560	0.7460	0.7950
DeepMEDA	0.4963	0.5833	0.6493	0.7037	0.7983

Table 4: Loan approval ratios under different algorithms given a default rate.

	8%		**10%**	
	Income	Rent ratio	Income	Rent ratio
TransBoost	76,257	**0.4336**	**72,902**	**0.4505**
TrAdaBoost	76,284	0.4315	73,780	0.4465
BDA	**74,254**	0.3908	72,988	0.4446
JDA	76,717	0.4161	73,212	0.4457
CORAL	76,370	0.4108	74,437	0.4445
KMM	77,137	0.4235	72,986	0.4419

Table 5: Socioeconomic features of the approved users by different algorithms at default rate of 8% and 10% (unit of Income: USD).

Conclusion

Achieving financial inclusion is crucial for mitigating financial inequality. However, reaching out to broader users incurs new challenges to financial risk evaluation. This paper develops a novel TransBoost algorithm which combines the merits of tree-based models and kernel methods. The experiments on three datasets show that the TransBoost outperforms all baselines in prediction accuracy and efficiency, shows stronger robustness to data sparsity, and provides clear model interpretation. Moreover, we demonstrate the value of the TransBoost in improving financial inclusion.

Acknowledgement

T. Lu, C. Wang, and Y. Xu thank the National Natural Science Foundation of China of Grants No. 71872050, 72101007, and 71531006, respectively.

References

Alessandro Rudi, L. R., Luigi Carratino. 2017. FALKON: An Optimal Large Scale Kernel Method. *arXiv preprint arXiv:1705.10958*.

Alex Davies, Z. G. 2014. The Random Forest Kernel and creating other kernels for big data from random partitions. *arXiv preprint arXiv:1402.4293*.

Bassem, B. S. 2012. Social and financial performance of microfinance institutions: Is there a trade-off? *Journal of Economics and International Finance*, 4(4): 92–100.

Breiman, L. 2000. Some infinity theory for predictor ensembles. *Technical Report 579*.

Chen, T.; and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Cull, R.; Ehrbeck, T.; and Holle, N. 2014. Financial Inclusion and Development: Recent Impact Evidence.—2014. *Focus Note*, 92.

Dai, W.; Yang, Q.; Guirong, X.; and Yong, Y. 2007. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning, Corvallis, USA*, 193–200.

Demirgüç-Kunt, A.; and Singer, D. 2017. Financial inclusion and inclusive growth: A review of recent empirical evidence. *World Bank Policy Research Working Paper*, (8040).

Dev, S. M. 2006. Financial inclusion: Issues and challenges. *Economic and political weekly*, 4310–4313.

Dorogush, A. V.; Ershov, V.; and Gulin, A. 2018. CatBoost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

Friedman, J. H. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Fuzhen Zhuang, K. D. D. X. Y. Z. H. Z. H. X. Q. H., Zhiyuan Qi. 2020. A Comprehensive Survey on Transfer Learning. *arXiv preprint arXiv:1911.02685*.

Guild, J. 2017. Fintech and the Future of Finance. *Asian Journal of Public Affairs*, 17–20.

Huang, J.; Gretton, A.; Borgwardt, K.; Schölkopf, B.; and Smola, A. 2006. Correcting sample selection bias by unlabeled data. *Advances in neural information processing systems*, 19: 601–608.

Jung, W.; Bollmann, S.; and Lee, J. 2020. Overview of quantitative susceptibility mapping using deep learning: Current status, challenges and opportunities. *NMR in Biomedicine*, e4292.

Kaggle@Yash. 2020. Lending Club 2007-2020Q3. https://www.kaggle.com/ethon0426/lending-club-20072020q1.

Karlan, D.; and Zinman, J. 2010. Expanding credit access: Using randomized supply decisions to estimate the impacts. *The Review of Financial Studies*, 23(1): 433–464.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30: 3146–3154.

Klapper, L.; El-Zoghbi, M.; and Hess, J. 2016. Achieving the sustainable development goals. *The role of financial inclusion. Available online: http://www. ccgap. org. Accessed*, 23(5): 2016.

Long, M.; Wang, J.; Ding, G.; Sun, J.; and Yu, P. S. 2013. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision*, 2200–2207.

Lu, T.; Zhang, Y.; and Li, B. 2020. Profit vs. Equality? The Case of Financial Risk Assessment and A New Perspective of Alternative Data. *The Case of Financial Risk Assessment and A New Perspective of Alternative Data (December 31, 2020)*.

Lundberg, S. M.; Erion, G.; Chen, H.; DeGrave, A.; Prutkin, J. M.; Nair, B.; Katz, R.; Himmelfarb, J.; Bansal, N.; and Lee, S.-I. 2020. From local explanations to global understanding with explainable AI for trees. *Nature machine intelligence*, 2(1): 56–67.

Lundberg, S. M.; and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, 4768–4777.

Musto, C.; Semeraro, G.; Lops, P.; De Gemmis, M.; and Lekkas, G. 2015. Personalized finance advisory through case-based recommender systems and diversification strategies. *Decision Support Systems*, 77: 100–111.

Óskarsdóttir, M.; Bravo, C.; Sarraute, C.; Baesens, B.; and Vanthienen, J. 2020. Credit scoring for good: Enhancing financial inclusion with smartphone-based microlending. *arXiv preprint arXiv:2001.10994*.

Pan, S. J.; Tsang, I. W.; Kwok, J. T.; and Yang, Q. 2010. Domain adaptation via transfer component analysis. *IEEE transactions on neural networks*, 22(2): 199–210.

Scornet, E. 2015. Random forests and kernel methods. *citeseer*.

Shen, J.; Qu, Y.; Zhang, W.; and Yu, Y. 2018. Wasserstein Distance Guided Representation Learning for Domain Adaptation. In McIlraith, S. A.; and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 4058–4065. AAAI Press.

Soto, E. A.; Bosman, L. B.; Wollega, E.; and Leon-Salas, W. D. 2021. Peer-to-peer energy trading: A review of the literature. *Applied Energy*, 283: 116268.

Sun, B.; Feng, J.; and Saenko, K. 2016. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Teja, A. 2017. Indonesian FinTech business: New innovations or foster and collaborate in business ecosystems? *The Asian Journal of Technology Management*, 10(1): 10.

Wang, J.; Chen, Y.; Feng, W.; Yu, H.; Huang, M.; and Yang, Q. 2020. Transfer Learning with Dynamic Distribution Adaptation. *ACM Trans. Intell. Syst. Technol.*, 11(1): 6:1–6:25.

Wang, J.; Chen, Y.; Hao, S.; Feng, W.; and Shen, Z. 2017. Balanced distribution adaptation for transfer learning. In *2017 IEEE international conference on data mining (ICDM)*, 1129–1134. IEEE.

World-Bank. 2018. Financial Inclusion: Financial inclusion is a key enabler to reducing poverty and boosting prosperity. https://www.worldbank.org/en/topic/financialinclusion. Accessed 11/05/2021.

Yang, Q.; Zhang, Y.; Dai, W.; and Pan, S. J. 2020. *Transfer learning*. Cambridge University Press.

Yu, C.; Wang, J.; Chen, Y.; and Huang, M. 2019. Transfer learning with dynamic adversarial adaptation network. In *2019 IEEE International Conference on Data Mining (ICDM)*, 778–786. IEEE.

Zhu, Y.; Zhuang, F.; Wang, J.; Ke, G.; Chen, J.; Bian, J.; Xiong, H.; and He, Q. 2020. Deep subdomain adaptation network for image classification. *IEEE transactions on neural networks and learning systems*, 32(4): 1713–1722.