Knowledge Transfer in Vision Recognition: A Survey

Ying Lu, Lingkun Luo, Di Huang, Alexandre Saidi, Yunhong Wang, Liming Chen

Knowledge Transfer in Vision Recognition: A Survey

YING LU, Ecole Centrale de Lyon, France
LINGKUN LUO, Ecole Centrale de Lyon, France
DI HUANG, Beihang University, China
ALEXANDRE SAIDI, Ecole Centrale de Lyon, France
YUNHONG WANG, Beihang University, China
LIMING CHEN, Ecole Centrale de Lyon, France

In this survey, we propose to explore and discuss the common rules behind knowledge transfer works for vision recognition tasks. To achieve this, we firstly discuss the different kinds of reusable knowledge existing in a vision recognition task, then we categorize different knowledge transfer approaches depending on where the knowledge come from and where the knowledge go to. This viewpoint is different to previous surveys on knowledge transfer. We further show that some potential research directions could be induced based on our categorization methodology.

CCS Concepts: • **Computing methodologies** → **Transfer learning**.

Additional Key Words and Phrases: knowledge transfer, transfer learning, vision recognition, computer vision, machine learning

ACM Reference Format:
Ying Lu, Lingkun Luo, Di Huang, Alexandre Saidi, Yunhong Wang, and Liming Chen. 2018. Knowledge Transfer in Vision Recognition: A Survey. 1, 1 (March 2018), 31 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Vision recognition is a core problem in computer vision. Its goal generally is to determine whether or not the input visual data contains some specific concept, object, or activity and to give corresponding predictive output about the recognized content. For example, in image classification tasks, the goal is to determine which pre-defined image class/concept (*e.g.* 'airport', 'castle', *etc.*) the input image belongs to; in object detection tasks, the goal is to determine whether the input image (or 3D data) contains some specific object (*e.g.* 'bicycle', 'dog', 'mug' *etc.*) and to output the corresponding bounding box (which defines the minimum rectangular/cubic area that contains the object) if an object exists; in semantic segmentation tasks, the goal is to predict semantic label for each pixel or super pixel in an input image. The common schema of this kind of tasks is that they all take some visual data as input and output some predictive semantic labels based on the input, therefore the classical way to solve this kind of tasks is supervised learning-based. One first learns a predictive model (*e.g.* a Convolutional Neural Network) with sufficient training data, and then applies the

Authors' addresses: Ying Lu, ying.lu@ec-lyon.fr, Ecole Centrale de Lyon, Ecully, France; Lingkun Luo, Ecole Centrale de Lyon, Ecully, France, lingkun.luo@ec-lyon.fr; Di Huang, Beihang University, Beijing, China, dhuang@buaa.edu.cn; Alexandre Saidi, Ecole Centrale de Lyon, Ecully, France, alexandre.saidi@ec-lyon.fr; Yunhong Wang, Beihang University, Beijing, China, yhwang@buaa.edu.cn; Liming Chen, Ecole Centrale de Lyon, Ecully, France, liming.chen@ec-lyon.fr.

Manuscript submitted to ACM

learned model for prediction on new data whose probabilistic distribution is assumed the same as that of the training data. This learning principle is known as the Empirical Risk Minimization in statistical learning theory [67]. In this way, each task is solved individually by learning a corresponding model from scratch. The disadvantage of this approach is obvious: the relatedness between different tasks is unexplored, thereby making the learning process inefficient. In many real-life applications, some tasks have abundant training data, but most others often have very few training data. When learning each task in an independent manner, it could be hard to solve a task which have limited training data since the available training data may not be enough for learning a reliable model.

It is thus of capital importance to be able to capitalize on previously learned knowledge. Indeed, taking into account the learned knowledge from previous tasks when learning a new task can be beneficial both in gaining extra training information and in saving training time (by avoiding training from scratch). For example, when training an image classification model for some rare categories, one may face the problem of having few training data, in this case, fine-tuning a CNN model pre-learned on some related image data as feature extractor could significantly improve the classification performance on target categories [75]. Knowledge transfer could also be applied for different kind of tasks. For example, since the ground-truth annotations for Object Detection tasks are usually harder to get than those for Image Classification tasks (the former includes not only class labels but also bounding box information), one could therefore borrow knowledge from a learned image classification task for training a new object detection model [61]. This knowledge transfer (also known as 'transfer learning') has been studied in previous works and has been attracting more and more attention from several research communities, *e.g.*, computer vision, machine learning, within the current big data era.

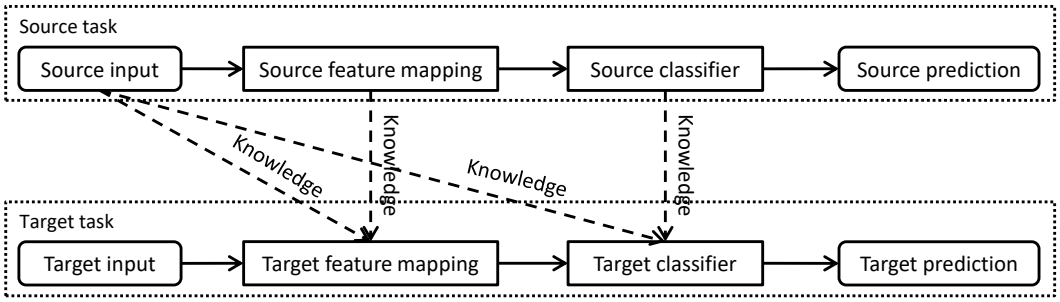

Fig. 1. Illustration of Knowledge Transfer: instead of learning a new task independently, knowledge transfer reuse existing knowledge in previous tasks for learning a new task.

Within the research community of knowledge transfer, one usually defines a *target* task, to which the knowledge will be transferred, and one or several *source* tasks, from which the knowledge will be captured or learned. Depending on assumptions on target and source tasks, knowledge transfer setting can be categorized into different scenarios, *e.g.* domain adaptation, self-taught learning, and few-shot learning.

Because of its importance, there exist an increasingly large amount of research work focused on TL and there have already been several surveys discussing state of the art research work on knowledge transfer at different time period as illustrated in Fig.2 and Fig.3. A first comprehensive survey on transfer learning was made in 2010 by Pan and Yang in [48], which discusses transfer learning methods for a broad range of applications, including vision recognition and other

types of applications. They have categorized different transfer learning works according to their assumptions (settings) and the nature of content to transfer. Specifically, as shown in the upper diagram of Fig.2, they distinguish three settings of transfer learning: (1) "*Inductive* transfer learning", where the source task could be different from the target task, some labeled target training samples should be provided to induce the target predictive model with the help of the source data or the source model. (2) "*Transductive* transfer learning", where the source and target share the same task, while having different data distributions. Therefore the source data should be adapted to the target data distribution in order to help learning an effective model for doing prediction on the target data; and (3) "*Unsupervised* transfer learning", where the target task and source task are all unsupervised tasks, *e.g.* clustering, dimensionality reduction, density estimation, *etc.*. Each setting can further depict different transfer learning scenarios with more detailed assumptions. Finally, as shown in the upper part of Figure 3, they come up with a synthesis of four different kinds of transfer learning approaches: (1) "*Instance Transfer*", where re-weighted source instances are used directly for learning the target task; (2) "*Feature representation transfer*", which tries to find a "good" feature representation which reduces the discrepancy between the source and the target distributions and increase the performance of classification and regression models on target data; (3) "*Parameter transfer*", which transfers parameters or priors from the source models to the target models; and (4) "*Relational knowledge transfer*", which builds mappings of relational knowledge from the source data to the target data.

Another survey was made in 2014 by Shao *et al.* [57] who focus on transfer learning works for vision categorization problems. As shown by the middle diagram of Figure 3, this survey categorizes transfer learning techniques into "feature representation level knowledge transfer" and "classifier level knowledge transfer", respectively.

As [48] and [57] don't cover important development in TL since 2015, a more recent survey was made in 2017 by Zhang *et al.* [79] who overview transfer learning techniques for cross dataset recognition problems. Like in [48], they also distinguish different works according to the settings. Specifically, they show what kinds of methods can be used when the available source and target data are presented in different forms. Compared to [48], they give a more detailed categorization of different cross-dataset settings, as shown in the lower diagram of Fig.2. They also summarize different kinds of criteria which could be used in solving knowledge transfer problems, as shown in the lower diagram of Fig.3. Like in [48], the transfer learning methods discussed in [79] also covers a broad range of applications. Vision recognition is only one of them.

As shown in the beginning of this section 1,the common way to solve vision recognition tasks follows the principle of Empirical Risk Minimization. Due to the specialty of visual data, this common solution could be further defined as a two-step framework: the feature extraction step and the prediction step. Following this two-step framework, [57] simply categorizes transfer learning works for vision categorization problems into two categories (as shown in figure 3). Although this categorization shows its correspondence to the common way of solving visual problems, it does not fully reveal the characteristics of knowledge transfer within this scope. Furthermore, [57] only covers research works before 2014. While with the rapid growth of vision recognition techniques, a lot of new works, especially those based on deep neural networks, are published since 2015. These recent works are not discussed in [57]. On the contrary, [79] includes more recent transfer learning works, while the authors do not focus on vision recognition.

In this survey, we propose to discuss knowledge transfer works for vision recognition tasks, and to explore the common rules behind these works. As the primary goal of TL methods is to harness learned knowledge for re-use in novel learning tasks, we overview in this survey knowledge transfer methods from the viewpoint of the knowledge being transferred. Specifically, We will firstly discuss the reusable knowledge in a vision recognition task, then we will categorize different kinds of knowledge transfer approaches by where the knowledge comes from and where the knowledge goes to. We aim at finding general rules across different TL settings instead of focusing on their particularities.

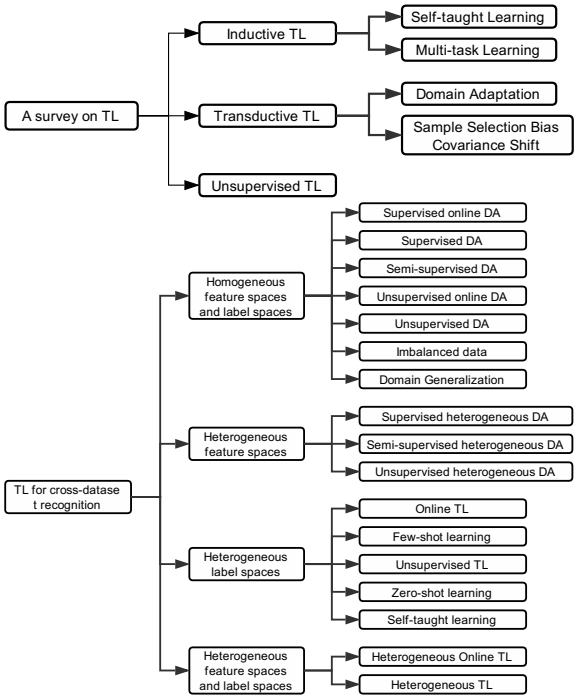

Fig. 2. Existing Categorizations of Transfer Learning Settings: The top part is the categorization of TL settings from [48] and the bottom part is the categorization of TL settings from [79] (TL stands for Transfer Learning and DA stands for Domain Adaptation)

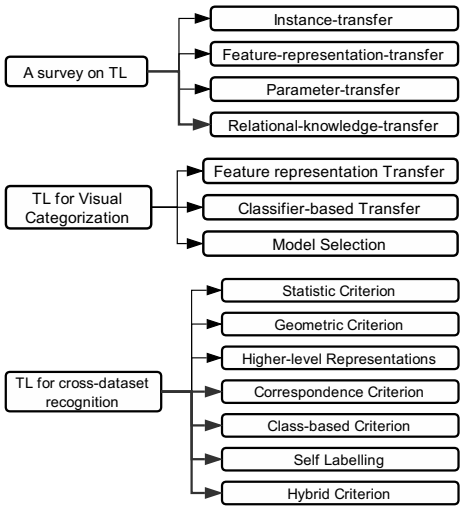

Fig. 3. Existing Categorization of Transfer Learning approaches/techniques: The top part is categorization of TL approaches from [48], the middle part is categorization of TL approaches from [57], and the bottom part is categorization of TL techniques from [79]

Table 1. Some common Transfer Learning settings concerned in this paper

Name	Same label space?	Inputs: Source Target	Objective
Inductive TL	Different	labeled labeled	single
Domain Adaptation	Same	labeled unlabeled	single
Self-taught Learning	Different	unlabeled labeled	single
Multi-task Learning	Different	labeled labeled	multiple
Zero shot Learning	Different	labeled no	single
One shot Learning	Different	labeled labeled	single

This viewpoint is in clear contrast to previous surveys on TL, *i.e.*, [48] [57][79]. However, any existing method has its own scope of applicability, and we will indicate the applicable scenarios when introducing each method. In Table 1 we have listed some common knowledge transfer settings (scenarios) discussed in this paper.

2 KNOWLEDGE IN VISION RECOGNITION

In this section we firstly introduce the notations we adopt in this paper to describe a vision recognition task and its solution. Then we discuss different types of knowledge that can be reused (transferred) from a previous vision recognition task to a new one.

The following notations are adopted in the subsequent: calligraphic letters in upper cases, *e.g.*, $\mathcal{X}$, denote sets or data spaces; bold letters in upper cases, *e.g.*, $\mathbf{M}$, denote matrices; bold letters in lower cases, *e.g.*, $\mathbf{x}$, denote column vectors.

Table 2. Characteristics which define a vision recognition task and its solution

In/Out Data	$\mathbf{X}^{tr}, \mathbf{Y}^{tr}, \mathbf{X}^{te}, \mathbf{Y}^{te}$
In/Out data spaces	$\mathcal{X}, \mathcal{Y}$
In/Out data distributions	$P(X), P(Y)$
Learned model	$f = f_K \circ \ldots \circ f_2 \circ f_1$
Feature data	$\mathbf{X}^{set,f_1}, \mathbf{X}^{set,f_2}, \ldots, \mathbf{X}^{set,f_{K-1}}$ $(set = \{tr, te\})$
Feature data spaces	$\mathcal{X}^{f_1}, \ldots, \mathcal{X}^{f_{K-1}}$
Feature data distributions	$P(X^{f_1}), P(X^{f_2}), \ldots, P(X^{f_{K-1}})$

Let's firstly define a vision recognition task $\mathcal{T}$ by two data spaces $\mathcal{X}, \mathcal{Y}$ and the corresponding data distributions $P(X), P(Y)$, where $\mathcal{X}$ is the input data space, $P(X)$ is the marginal probability distribution of the input data X, $\mathcal{Y}$ is the desired output label space, and $P(Y)$ is the probability distribution of the output data Y. Here $P(Y)$ can also be noted as $P(Y|X)$ and therefore be interpreted as the conditional distribution of Y knowing X. The goal of the vision recognition task is to find an optimal mapping $f(\cdot)$ which projects the input data X from the data space $\mathcal{X}$ to the label space $\mathcal{Y}$ so that the mapped labels Y best correspond to the ground-truth labels Y^{gt}. This mapping $f(\cdot)$ could be further decomposed into a series of projections $f = f_K \circ \ldots \circ f_2 \circ f_1$, where each f_k maps data from the space $\mathcal{X}^{f_{k-1}}$ to the space $\mathcal{X}^{f_k}$. Specifically, $\mathcal{X}^{f_0} = \mathcal{X}$ is the input data space, $\mathcal{X}^{f_K} = \mathcal{Y}$ is the output data space, and $\{\mathcal{X}^{f_1}, \ldots, \mathcal{X}^{f_{K-1}}\}$ are intermediate feature spaces. In this way, a solution to task $\mathcal{T}$ could be defined as $\mathcal{S} = \{f_1, f_2, \ldots, f_K\}$, and these projections define new feature spaces $\mathcal{F} = \{\mathcal{X}^{f_1}, \ldots, \mathcal{X}^{f_{K-1}}\}$ and the output data space given the input data space.

For example, if we use a K-layer Neural Network model to solve the task $\mathcal{T}$, the solution could be denoted as $\mathcal{S}_{KlayersNN} = \{f_1, f_2, \ldots, f_K\}$, and the corresponding feature spaces are $\mathcal{F}_{KlayersNN} = \{\mathcal{X}^{f_1}, \ldots, \mathcal{X}^{f_{K-1}}\}$, where $\mathcal{X}^{f_k}$ $(k = \{1, 2, \ldots, K-1\})$ is the feature space which contains the output of the k-th layer of the Neural Network, and $\mathcal{X}^{f_K} = \mathcal{Y}$ contains the output of the last layer, which is the desired output label space. If we use a traditional way to solve $\mathcal{T}$, for example, a SIFT feature extraction step with an SVM classifier, the solution could then be denoted as $\mathcal{S}_{SIFT-SVM} = \{f_{SIFT}, f_{SVM}\}$ along with $\mathcal{F}_{SIFT-SVM} = \{\mathcal{X}^{f_{SIFT}}\}$, where $\mathcal{X}^{f_{SIFT}}$ is the feature space defined by the output of the SIFT feature extraction.

In reality, the probability distributions $P(X)$ and $P(Y)$ for $\mathcal{T}$ are usually not given in an analytical form. Normally they could be estimated through the given training data set $\{\mathbf{X}^{tr}, \mathbf{Y}^{tr}\}$. The test data $\{\mathbf{X}^{te}, \mathbf{Y}^{te}\}$ are supposed to follow the same distribution as the training data do, therefore allowing the model learned on training data to be applicable on the test data. In the training phase, a model $f(\cdot)$ is learned with the training set $\{\mathbf{X}^{tr}, \mathbf{Y}^{tr}\}$, and then in the testing phase, predictions could be made by applying the learned model $f(\cdot)$ on the test data $\mathbf{X}^{te}$, and the performance of the model could be evaluated by comparing the predictions with the groundtruth labels $\mathbf{Y}^{te}$. Following the decomposition of f described above, we could find out that there exist a series of feature data $\{\mathbf{X}^{set,f_1}, \mathbf{X}^{set,f_2}, \ldots, \mathbf{X}^{set,f_{K-1}}\}$ $(set = \{tr, te\})$, each belongs to their corresponding feature space, *i.e.* $\mathbf{X}^{tr,f_k}$ and $\mathbf{X}^{te,f_k}$ belong to $\mathcal{X}^{f_k}$. And in each feature space the data should follow a specific probability distribution, we denote these data distributions as $\{P(X^{f_1}), P(X^{f_2}), \ldots, P(X^{f_{K-1}})\}$.

Table 2 lists the main characteristics introduced above that describe a vision recognition task and its solution. At this level, We can observe that there exist two types of *knowledge* for a given vision recognition task: the first one is directly represented by raw data which includes the input and output data, the corresponding data spaces they belong to, and their data distributions; the other is learned knowledge which includes the learned model, feature data generated by this model, feature data spaces and feature data distributions. Both these two kinds of knowledge have the possibility to be reused (transferred) in a new vision recognition task. The knowledge directly represented by raw data is more flexible to be reused since they could be adapted to the target data for learning a new model; In contrast, the learned knowledge is more restricted to the source data. Nevertheless, when the source task is well chosen (*i.e.*, well related to target task), the reuse of learned knowledge could be both efficient and effective.

For example, it has been shown that deep convolutional neural networks (CNN) have the ability to produce transferable features [75]. Therefore, one can adopt a pre-learned CNN model, fix the feature extraction layers' parameters and only retrain the classification layer's parameters on new data, and the resulting new model is expected to have discriminative performance on the new data. In this way, we are actually reusing the *knowledge* from the parameters of the learned feature projections (*i.e.* $\{f_1, f_2, \ldots, f_{K-1}\}$) of a given pre-learned vision recognition task.

In some cases, the target training data is far from enough for learning a reliable classification model, some instance based transfer learning approaches then choose to select source samples to enrich the target training data directly. For example, Dai *et al.* in [16] make use of AdaBoost to choose from the source labeled samples the ones which are close to the target probability distribution to help the learning of the target classification model. In this way, the *knowledge* transferred from the source task are source raw data (*i.e.* **X** and **Y**), and they are reused for learning a prediction model for the target task.

Another example is unsupervised Domain Adaptation (DA). In unsupervised DA, the target training data is unlabeled, therefore it is impossible to use the target training data solely for learning a reliable classification model. We thus need to seek for help in labeled source data, which is assumed to share the same label space with the target domain and to own similar but different marginal distribution *w.r.t* the target domain. In this case, one can attempt to align the source and target data distributions (either by projecting the two distribution into a shared feature space in which the two feature distributions are as close to each other as possible; or by projecting one distribution to fit the other one). Then as the two distributions are well aligned in a new feature space, a classification model learned on the source distributions is then considered as applicable for target data. In this way, the *knowledge* reused from source is actually the same as in the previous example, *i.e.*, source labeled data is reused and adapted to target data for learning a new classification model for target task.

In the following sections, we overview different knowledge transfer techniques or approaches for vision recognition tasks. They are categorized according to the origin of *knowledge* being transferred, *i.e. raw data* or *pre-learned model*), as well as the destination of the transferred knowledge, *e.g.* feature extractor parameters or predictor parameters, *etc.*). Table 3 synthesizes the different knowledge transfer techniques that are presented in the subsequent.

Table 3. Categorization of Knowledge Transfer Approaches by where the knowledge come from and where the knowledge go to

From \ **To**	Feature Extractor parameters	Predictor parameters	Both
Raw data	section 3.2	section 3.1	section 3.3
Learned Feature extractor	section 4.1	-	-
Learned Predictor	-	section 4.2	-
Learned model	-	-	section 4.3

3 KNOWLEDGE TRANSFER FROM SOURCE DATA

As we have discussed in section 2, *raw data* is low level information compared to *pre-learned model*. Therefore there exist more possibilities to reuse *raw data* in target task. The main advantage of using *raw data* instead of *pre-learned model* is that *raw data* could be more easily adapted to target task. There are various ways to adapt source data to target data. For example, when the source and target data distributions are not very far from each other, a straight forward way is to re-weight (or select) source samples (or sets) so that the resulting data set could fit the target data distribution.

An alternative way is to learn a shared feature extractor which projects both source and target data into a common feature space, in which the source and target feature distributions are well aligned to each other. When the source and target data distributions are not very close to each other, one could learn a projection which projects source data to the target data space (or the inverse) so that the resulting two data distributions would be close to each other. Once the two data/feature distributions are well aligned, the target task can then benefit from this shared data/feature space in different ways. For example, the target task could benefit from the discriminability of a learned shared feature space, or it could benefit from the source conditional distribution for learning a classifier in the shared data space if the target samples are not enough to support a reliable classification border.

Therefore we further categorize knowledge transfer approaches by two different kinds of knowledge destinations, *i.e.* the *feature extractor parameters* and the *predictor parameters* for the target task. These two kinds of knowledge destinations are not mutually exclusive, some works may focus on transferring source data knowledge to one particular kind of target model parameters, while some works reuse source data for learning all kinds of target model parameters.

In the following we introduce three groups of previous works that transfer source data knowledge for learning target model parameters. Detailed knowledge transfer category for each section can be found in table 3.

3.1 Knowledge transfer from reweighted source data samples/sets to target classifier parameters

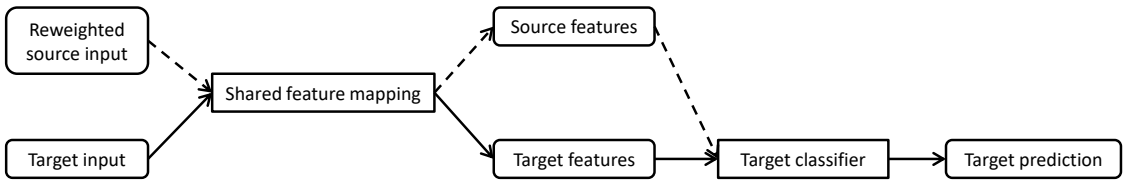

Fig. 4. Illustration of Knowledge Transfer from re-weighted source data to target classifier parameters

A natural way to adapt source data to target is by selecting most related data samples directly from source training data set, or selecting most related source sets when multiple source sets are presented. The selection is usually done by giving weights on source samples/sets. In this subsection we show a group of methods which transfers knowledge from the reweighted source data samples/sets to the learning of target classifier parameters.

In early days before deep convolutional neural networks take over traditional feature extraction skills, research works on knowledge transfer mainly focus on borrowing knowledge from source to build the target predictor (*i.e.* knowledge transfer from source data to target predictor parameters). For feature extraction they use traditional methods (such as SIFT, HOG, *etc.*) on source and target training samples. And they use the extracted features of both source and target training samples as input data to learn a predictive model for target task. In the training process, source samples/sets that are more related to target samples will be given more important weights to enhance the knowledge transferred from them. Different works may use different ways to describe the relatedness between source and target samples, and use different strategies to select related source samples/sets.

In [16] the authors make use of AdaBoost for transfer learning by choosing from the source labeled samples the useful ones for building a classifier for target data. Assume having a few target training samples and a large amount of source training samples, their aim is to select the source samples that follow the same probability distribution as the target samples. To achieve this goal, they build a Transfer AdaBoost framework, namely TrAdaBoost, for learning on

target and source training samples at the same time. In each iteration, AdaBoost works normally on target samples, *i.e.* it increases the weights on misclassified target samples; on the other hand, for source training samples, the misclassified ones are considered as the outliers to the target distribution, therefore the weights on misclassified source samples are decreased. In this way, after several iterations, the source samples that fit the target distribution better will have larger weights, while the source samples that do not fit the target distribution will have lower weights. The source samples with large weights will then intent to help the learning a better classifier for target data.

Since this TrAdaBoost only borrows knowledge from one source task, in [73] the authors extend this method to MultiSource-TrAdaBoost which borrows knowledge from multiple source tasks. Assume having several different source training sample sets, each with abundant labeled samples, and one target training sample set with few labeled samples. In each iteration of AdaBoost, one weak learner is build on each source training set, and the one with the best performance on target set, *i.e.* the one appears to be the most closely related to the target, is chosen as the weak learner for current iteration. In this way, the authors claim that the MultiSource-TrAdaBoost can better avoid negative transfer effect caused by brute-force knowledge transfer from the single source when this source is not closely related to the target.

Beware that, both TrAdaBoost and MultiSource-TrAdaBoost work for binary classification only, *i.e.* the source and target label spaces are the same, which could be defined as $\{+1, -1\}$ where $+1$ indicates positive sample and -1 indicates negative sample. Therefore these two works could make use of selected source samples simply as a part of target training data set for learning classification model. This strategy works when source set and target set are positively correlated, *i.e.* there exist source positive samples which are related to target positive samples and source negative samples which are related to target negative samples. Otherwise, when the two data distributions are not correlated, making use of source samples may harm the performance of the learned model for target task.

An alternative approach would solve this more complicated situation. In [52] the authors propose to use label propagation for knowledge transfer, *i.e.* they propagate labels from samples of selected source sets to each target sample. The resulting method is named Cross-Category Transfer Learning (CCTL). The coefficient for label propagation from a source sample to a target sample is defined by a transfer function, which combines both sample relatedness and domain relatedness between source and target. And the source set selection is also achieved by AdaBoost. Specifically, they define a real-valued transfer function $T_S(\mathbf{x}, \mathbf{x}_{l,i}) = \phi_S(\mathbf{x}, \mathbf{x}_{l,i})k(\mathbf{x}, \mathbf{x}_{l,i})$ (where $\mathbf{x}_{l,i} \in \mathcal{D}_{S,l}$, $\mathbf{x} \in \mathcal{D}_T$) to connect the l-th source set $\mathcal{D}_{S,l}$ and the target training set $\mathcal{D}_T$. In which, the $\phi_S(\mathbf{x}, \mathbf{x}_{l,i}) = \mathbf{x}^\top S \mathbf{x}_{l,i}$ measures the correlation between two different categories, and the kernel function $k(\mathbf{x}, \mathbf{x}_{l,i})$ measures the sample similarity. A cross-category classifier is learned to propagate the labels from the instances in l-th source set $\mathcal{D}_{S,l}$ to the target training set to form a discriminant function $h_l(\mathbf{x})$ as follows:

$$h_l(\mathbf{x}) = \frac{1}{|\mathcal{D}_{S,l}|} \sum_{\mathbf{x}_{l,i} \in \mathcal{D}_{S,l}} y_{l,i} T_S(\mathbf{x}, \mathbf{x}_{l,i}) \tag{1}$$

where $|\mathcal{D}_{S,l}|$ is the cardinality of $\mathcal{D}_{S,l}$, and $y_{l,i}$ is the ground-truth label for $\mathbf{x}_{l,i}$. The parameter matrix S for $h_l(\mathbf{x})$ is learned by minimizing the following objective function:

$$S^* = \arg\min_S \Omega_l(S) \tag{2}$$

where

$$\Omega_l(S) = \sum_{\mathbf{x}_i \in \mathcal{D}_T} \mathbf{w}_i (1 - y_i h_l(\mathbf{x}_i))_+ + \frac{\lambda}{2} \|S\|_F^2 \tag{3}$$

where $(\cdot)_+ = max(0, \cdot)$, $\|S\|_F$ is the Frobenius norm of the matrix S, λ is the balancing parameter, y_i is the ground-truth label for $\mathbf{x}_i$ and $\mathbf{w}_i$ is the sample weight for $\mathbf{x}_i$.

Finally, they define a common AdaBoost process, in each iteration they learn a cross category classifier from each source domain to the target domain, and a same one from target domain to itself, they then pick from these classifiers the one with the minimum training error as the weak classifier for current iteration. the final output is a combination of the weak classifiers learned in all iterations.

Since this CCTL takes into account both category correlations and sample correlations, it shows a better performance than the previously introduced TrAdaBoost and MultiSource-TrAdaBoost. However, when having L different source domains, in each iteration of CCTL one should solve $L + 1$ optimization problems. This makes this method not very efficient, especially when having a lot of source domains.

Although these methods make use of traditional feature extraction, we could easily replace their feature extractors with a state-of-the-art CNN model, which is pre-learned on some related large scale database, to benefit from the better performance of deep features.

In [43] the authors propose a new method, namely Discriminative Transfer Learning (DTL), to reuse source selected data set for learning target classifier. They also show that by combining deep features and knowledge transfer in target classifier one can achieve better results than using traditional features. Unlike previous methods, the authors propose to build sparse reconstruction based discriminative classifiers for target task with selected source sample sets. They use positively correlated source sets as positive dictionaries and negatively correlated source sets as negative dictionaries, the difference between reconstruction errors of target samples on positive dictionary and those on negative dictionary is served as the discriminator. The source data sets are selected through two parallel AdaBoost processes. Therefore the resulting classification model is a combination of multiple selected dictionary pairs. Since this method makes use of both positively correlated source sets and negatively correlated source sets, it shows a better performance than the previously introduced CCTL, and it is also much more efficient than CCTL both on training time and on prediction time.

As can be seen, the methods introduced in this section are all based on boosting framework and all work for binary classification. It is possible to extend these kind of methods to multi-class classification by one-vs-one, one-vs-all or EOOC (Error Correcting Output codes) based approaches, although this extension may increase the time for training and prediction. In table 4 we give a comparison of the detailed setting of these methods along with the original AdaBoost.

3.2 Knowledge transfer from source data to target feature extractor

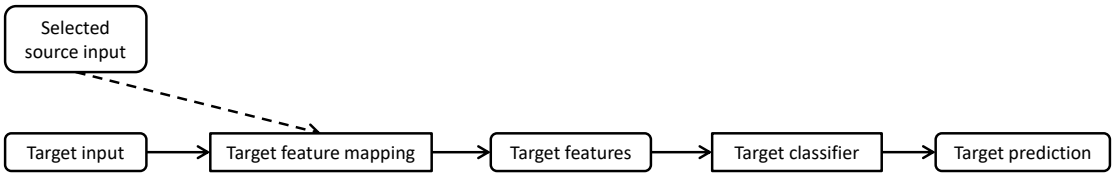

Fig. 5. Illustration of Knowledge Transfer from source data to target feature extractor

In previous subsection we have introduced some transfer learning methods which transfer knowledge from source data to target discriminator parameters. In this section, we show another group of methods, which also transfer knowledge from source data, while they mainly focus on using these knowledge to learn a feature extractor adapted for

Table 4. Comparison of boosting based knowledge transfer methods

Boosting based methods	In each Boosting iteration:	
	Update sample weights (↑: augment weight; ↓: decrease weight)	Choose weak learner
AdaBoost	Wrongly classified samples ↑ Correctly classified samples ↓	Learned with weighted samples
TrAdaBoost	Wrongly classified target samples ↑ Correctly classified target samples ↓ Wrongly classified source samples ↓ Correctly classified source samples ↑	Learned with weighted target and source samples
MultiSourceTrAdaBoost	Wrongly classified target samples ↑ Correctly classified target samples ↓ Wrongly classified source samples ↓ Correctly classified source samples ↑	The one with best performance on target from candidates learned with multiple sources
CCTL	Wrongly classified samples ↑ Correctly classified samples ↓	The one with best performance on target from candidate cross-category classifiers
DTL	Wrongly classified samples ↑ Correctly classified samples ↓	Multiple pairs of source sets which show best performance on target

the target task.

3.2.1 *Knowledge from labeled source data.*

As shown in section 3.1, a straight forward way to adapt source data to target is by re-weighting source data samples/sets. For learning feature extractor parameters adapted for target, we could also use this kind of strategy.

For example, in [25] the authors propose a method which selects related source samples and then learn a deep convolutional neural network for feature extraction through joint fine-tuning with both selected source samples and target training samples. The proposed *Selective Joint Fine-Tuning* is done by two steps:

In step 1, they project both source and target training samples into a low-level feature space by applying either a Gabor filter bank or kernels in the convolutional layers of AlexNet (pre-trained on ImageNet). Then they select nearest source samples for each target training sample. The number of nearest samples is adaptive, *i.e.* hard target samples may get a larger number of source neighbors.

In step 2, they make use of the selected source samples with target training samples to optimize the source and target objective functions at the same time. A 152-layer residual network pre-trained on ImageNet or Places is shared by source and target predictive models as the feature extractor. In this way, the learned DNN for feature extraction benefits from the knowledge of the selected source data.

In a more recent work [45] the authors address a similar problem: how to select an optimal Subset of Classes (SOC) from the source data, subject to a budget constraint, for training a feature extractor which generates good features

for the target task. To achieve this goal, they use a sub-modular set function to model the accuracy achievable on a new task when the features have been learned on a given subset of classes of the source dataset. An optimal subset is identified as the set that maximizes this sub-modular function. The maximization can be accomplished using a greedy algorithm that comes with guarantees on the optimality of the solution.

3.2.2 *Knowledge from unlabeled source data.*

The methods shown in previous section 3.2.1 all make use of labeled source data for training feature extractor. In reality, it is usually more expensive to get labeled data than unlabeled data, therefore transferring knowledge from unlabeled data would be a good choice when it is not easy to get source labels. In this subsection, we show some works that make use of unlabeled source data for learning a feature extractor purposed for target task.

A first group of methods is the so called *self-taught learning* methods. These methods aim to learn a re-constructive dictionary from unlabeled source data, this learned dictionary could then be used for feature extraction by sparse coding for target task.

[54] is the first work that proposed the *self-taught learning* problem. They proposed a sparse coding based approach, which learns high-level feature expressions with unlabeled data using sparse coding. By applying the learned model for feature extraction on target data, the target classification performance could be improved.

Assume having a target training set with m training samples $\{(\mathbf{x}_1^{(l)}, y_1), (\mathbf{x}_2^{(l)}, y_2), \ldots, (\mathbf{x}_m^{(l)}, y_m)\}$. Where "$(l)$" indicates that $\mathbf{x}^{(l)}$ is a labeled sample. A source set with k unlabeled samples $\mathbf{x}_1^{(u)}, \mathbf{x}_2^{(u)}, \ldots, \mathbf{x}_k^{(u)} \in \mathbb{R}^n$ is also given. The unlabeled data are assumed to be related to the target data, without the strong assumption that they should be from the same distribution or the same categories.

For learning the reconstructive dictionary, the optimization problem is defined as follows:

$$\min_{\mathbf{B},\mathbf{A}} \sum_i \|\mathbf{x}_i^{(u)} - \sum_j a_{i,j}\mathbf{b}_j\|_2^2 + \beta\|\mathbf{a}_i\|_1 \tag{4}$$

$$\text{s.t.} \quad \|\mathbf{b}_j\|_2 \leq \mathbf{1}, \forall j \in 1, \ldots, s$$

Where $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_s\}$ are *basis vectors* (*i.e.* dictionary items) with $\mathbf{b}_j \in \mathbb{R}^n$, and $\mathbf{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_k\}$ are *activations* with $\mathbf{a}_i \in \mathbb{R}^s$, $a_{i,j}$ being the activation of $\mathbf{b}_j$ for $\mathbf{x}_i^{(u)}$. Under the condition of sparse reconstruction, $\mathbf{B}$ should be a redundant dictionary, therefore s should be much larger than n. In this optimization objective, the first term encourages each input sample to be well reconstructed by the redundant dictionary, and the second term encourages the activation vectors to be sparse, *i.e.* to have low $l1$ norm. This problem (4) could be solved on optimizing $\mathbf{A}$ and $\mathbf{B}$ alternatively until converge.

Once the reconstructive dictionary $\mathbf{B}$ is learned with source unlabeled data, it could then be applied for feature extraction on target data. To compute features $\hat{\mathbf{a}}_i \in \mathbb{R}^s$ for each target input $\mathbf{x}_i^{(l)} \in \mathbb{R}^n$, the optimization objective is defined as follows:

$$\hat{\mathbf{a}}_i = \arg\min_{\mathbf{a}_i} \|\mathbf{x}_i^{(l)} - \sum_j a_{i,j}\mathbf{b}_j\|_2^2 + \beta\|\mathbf{a}_i\|_1 \tag{5}$$

This is a convex problem and can be solved efficiently. The optimized activations $\hat{\mathbf{a}}_i$ is then considered as the new feature representation for the input sample and is then fed to a standard classification method (*e.,g.* SVM) as input for learning target classifier.

The authors argue that, compared to PCA, this proposed sparse coding process is a better way for unsupervised feature learning in the self-taught learning scenario for two reasons: 1) the sparse coding method learns a non linear representation while PCA learns a linear feature representation; 2) since PCA is a kind of dimension reduction method, the feature vector learned by PCA cannot be longer than the dimension of input sample, while sparse coding can learn a much longer feature vector thanks to the large size of the redundant dictionary. Since the feature vector learned by sparse coding is both long and sparse, it could be considered as a much higher-level representation.

The authors perform experiments of the proposed sparse coding approach with two standard classifiers: a support vector machine(SVM) and a Gaussian discriminant analysis (GDA). In addition, they also proposed a Fisher kernel based classifier specifically designed for sparse coding features. They show results on several different applications including image classification, the results confirmed the effectiveness of the proposed approach.

This sparse coding based approach is widely adopted for self-taught learning scenarios, and is also improved from different aspects by different researchers. For example, in [71] the authors propose to learn the sparse coding basis (*i.e.*, the redundant dictionary) using not only unlabeled samples, but also labeled samples. They also proposed a principled method to seek the optimal dictionary basis vectors for a smaller dictionary which demands less computational cost.

In a recent work [35], the authors propose a new sparse coding based self-taught learning framework for visual learning, which is named "self-taught low-rank (S-Low) coding". In addition to sparse coding, they also add a low-rank constraint into the reconstruction objective function to preserve the subspace structures contained in target data space. This problem is formulated as a dictionary learning problem with rank-minimization constraint, which is a non-convex problem. The authors propose a "majorization-minimization augmented Lagrange multiplier (MM-ALM) algorithm" to solve it.

Another group of the methods are recent deep learning methods. The rapid growth of deep learning techniques shows new ways for unsupervised feature learning. Unsupervised deep learning models, such as auto-encoders (AEs) [69] [70], deep belief networks (DBNs) [34] [33] and generative adversarial networks (GANs) [18], could be used for feature learning with unlabeled data. The resulting feature extraction networks could then be applied for new target tasks.

Although there has not been much work studying unlabeled source data selection when learning a transferable feature extractor for the target task, we believe this should be a research direction worth to explore. Since data selection has already shown its effectiveness when transferring knowledge from labeled source data to target feature extractor, it should show equal importance when transferring knowledge from unlabeled source data to target feature extractor.

3.3 Domain Adaptation: knowledge transfer from source data to target feature extractor and classifier parameters

A special research direction of knowledge transfer, which has been widely explored for a relatively long time, is the Domain Adaptation problem. In Domain Adaptation, we assume that the target data and source data share the same input and output spaces ($\mathcal{X}_T = \mathcal{X}_S$ and $\mathcal{Y}_T = \mathcal{Y}_S$), while having different but related data distributions ($P(X_T) \neq P(X_S)$ and $P(Y_T|X_T) \neq P(Y_S|X_S)$). Depending on whether labeled target training samples are available or not, DA problems could be further categorized into unsupervised Domain Adaptation (data available in training phase including: labeled source data and unlabeled target test data) and supervised Domain Adaptation (data available in training phase

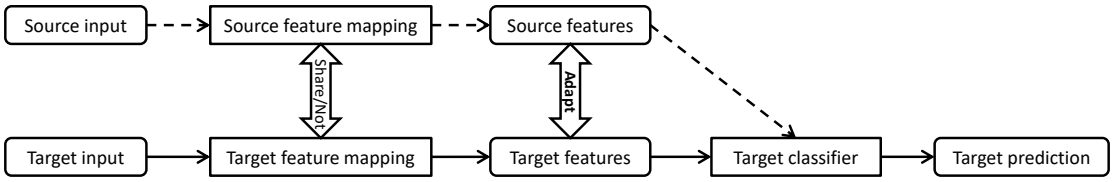

Fig. 6. Illustration of Knowledge Transfer in Domain Adaptation

including: labeled source data, unlabeled target test data and a few labeled target training data). Whether supervised or unsupervised, the main goal of Domain Adaptation is to adapt source data distribution to target data distribution, so that the model parameters learned with both source and target data could show good performance for target task. Therefore we consider Domain Adaptation as a kind of knowledge transfer from source raw data to target model parameters (instead of knowledge transfer from pre-learned model).

It is noteworthy that, in DA, especially in unsupervised DA, the training is usually done in a transductive manner. This means, unlabeled target test data is also involved in the training stage in order to achieve adaptation between source data distribution and target data distribution. This is not a common way, since in machine learning we usually assume test data not accessible during training phase. In reality, we also prefer a model trained with only training data, which allows us to apply the model directly on any future test data that follows the distribution assumption without re-training. However in transfer learning scenarios the situation may be a little different. As in most scenarios which knowledge transfer is needed, labeled target training data must be expensive to be got, that's the reason why we need to transfer knowledge from source task to help the learning of target task. In this case, transductive learning shows its advantage: by making use of the unlabeled target test data, one can get a more accurate estimation about the target data distribution instead of only rely on the rare target training data. The disadvantage of transductive learning is that, it would be more expensive to apply this kind of methods in reality, since it requires accessing test data in training phase, making it impossible to deliver a nicely pre-learned model for direct application on new data.

As we have talked about, the main goal of Domain Adaptation is adaptation between source and target data distribution. There exists various ways to achieve this distribution adaptation, in the following we show several groups of methods. The ultimate goal of adaptation is of course to adapt both marginal distributions ($P(X_S)$ and $P(X_T)$) and conditional distributions ($P(Y_S|X_S)$ and $P(Y_T|X_T)$), while some works may only focus on adapting one of the two, and others may focus on adapting both of them. In table 5 we show a comparison of the DA methods we introduce in this section.

3.3.1 *Learning a shared feature mapping.*

A natural way to achieve distribution adaptation is to learn a shared new feature space for source and target data, in which the distribution discrepancy between source and target feature data is minimized. A lot of DA works follow this way. Here we show a group of methods doing adaptation in this way. They make use of different methods for finding the new feature space (*e.g.* PCA, FLDA, metric learning, *etc.*), and use different methods for measuring the discrepancy between source and target distributions (*e.g.* Bregman divergence, MMD, *etc.*).

Adaptation with Bregman divergence based distance measure:

The first work is [58] where the authors proposed a Bregman Divergence based regularization schema for transfer subspace (representation) learning, which combines Bregman divergence with conventional dimensionality reduction algorithms. This regularized *subspace learning* learns a feature mapping and a classifier at the same time. The regularization term on the feature transformation parameters is based on a *bregman divergence* between the source marginal distribution and the target marginal distribution. Therefore the difference between the two marginal distributions will be explicitly reduced during optimization.

Specifically, assume some feature transformation $v(\mathbf{x}) = \theta\mathbf{x}$ where θ is a z by d matrix that maps the original d dimensional input feature vector into a new z dimensional feature space. In subspace learning framework we learn this matrix θ by minimizing a specific objective function $F(\theta)$:

$$\theta^* = \arg\min_{\theta \in \mathbb{R}^{z \times d}} F(\theta) \tag{6}$$

The objective function $F(\theta)$ could be designed specifically for different applications, in this work it is defined to minimize the classification loss in the shared subspace with different assumptions. For example, the subspace chosen by Fisher's linear discriminant analysis (FLDA) should have minimized trace ratio of the within-class scatter matrix and the between-class scatter matrix.

To reduce the distribution difference between source and target data, the authors propose a Bregman divergence based regularization term $D_\theta(P_S \parallel P_T)$, which measures the difference between source and target distributions in the new subspace θ. Their proposed transfer subspace larning (TSL) framework could then be get by integrating this Bregman divergence based regularization term into the objective function (6):

$$\theta^* = \arg\min_{\theta \in \mathbb{R}^{z \times d}} F(\theta) + \lambda D_\theta(P_S \parallel P_T) \tag{7}$$

with respect to specific constraints, *e.g.*, $\theta^\top \theta = I$. Here λ is the regularization parameter that controls the trade-off between the two terms in (7).

The authors show examples of this transfer subspace learning framework using different $F(\theta)$ (*i.e.* combining with different dimensionality reduction methods), such as transfered principal components analysis (TPCA), transfered Fisher's linear discriminant analysis (TFLDA), transfered locality preserving projections (TLPP) with supervised setting, *etc.* They also give experimental results on face image data sets and text data sets, which show the effectiveness of the proposed framework for transfer learning problems.

Adaptation with Maximum Mean Discrepancy (MMD) as distance measure:

Similar to the previous approach, in [46] the authors proposed a transfer learning algorithm which also combines conventional dimensionality reduction method and a distance measure for measuring the distance between marginal distributions of source data and target data. In this work the authors make use of the Maximum Mean Discrepancy as distribution distance measure, and PCA as the dimensionality reduction method.

The MMD between two sample sets could be considered as first mapping the two sample sets into a RKHS, then calculate the distance between the means of the two sets in the new space (in practice it is calculated with kernel trick which avoids the explicit mapping of the samples). By combining this MMD with common dimension reduction methods, the authors propose the Maximum Mean Discrepancy Embedding (MMDE), which learns a new latent feature space shared by source and target. A classifier is then learned in this latent space with source labeled data, and this

learned classifier is directly used for target classification task (*i.e.*, they assume that in the latent space the conditional distributions of source data and target data are the same).

The authors perform experiments on indoor WiFi localization dataset and text classification dataset, the results showed that using knowledge transfer with the proposed MMDE can effectively improve the model performance compared to the same model learned without knowledge transferred from the source data.

However this method suffers from two limitations: it does not generalize to out-of-sample patterns, and the semi-definite program (SDP) it uses is computationally expensive. To get ride of these limitations, the authors further proposed in [47] a new approach, named *transfer component analysis* (TCA), which learns a set of common *transfer components* for the source and the target domains, at the same time minimizes the difference between the two data distributions in the new subspace and preserves the properties of the source and the target data.

Unlike MMDE, this proposed TCA avoids the use of SDP and does not have the problem for out-of-sample patterns. Furthermore, they propose to use an explicit low-rank representation in a unified kernel learning method, instead of using a two-step approach.

Both MMDE and TCA focus on minimizing the marginal distributions' discrepancy between source and target data, while assuming that the conditional distributions of source and target data in the learned novel feature space are equal so that a classifier learned on source data can be directly applied to target data. However, such equality assumption of conditional distributions is strong and cannot always be respected. In [38], Long *et al.* proposed a *Joint Distribution Adaptation* (JDA), which aims to jointly adapt both the marginal and conditional distributions in a principled dimensionality reduction procedure. Similar to previously introduced MMDE and TCA, JDA also makes use of *Maximum Mean Discrepancy* as the distance measurement between distributions.

Since they consider the problem of *unsupervised* domain adaptation and thereby assume that no labeled sample is provided in target training set. As a result, to reduce the mismatch of conditional distributions, the authors propose to make use of the *pseudo* labels of the target data, which are obtained by applying the classifier learned on source labeled data directly to the target data. Furthermore, Long *et al.* propose to explore the sufficient statistics of class-conditional distributions $P(\mathbf{x}^{(S)}|y^{(S)})$ and $P(\mathbf{x}^{(T)}|y^{(T)})$ instead of the posterior probabilities $P(y^{(S)}|\mathbf{x}^{(S)})$ and $P(y^{(T)}|\mathbf{x}^{(T)})$. With the true labels on the source data and pseudo labels on the target data, they match the distributions $P(\mathbf{x}^{(S)}|y^{(S)} = c)$ and $P(\mathbf{x}^{(T)}|y^{(T)} = c)$ for each class $c \in \{1, \ldots, C\}$ in the label set $\mathcal{Y}$.

The authors proposed an iterative approach where they optimize the feature mapping and pseudo labels alternatively until convergence of the pseudo labels. They performed experiments for image classification problems to evaluate the JDA approach. The results verified the effectiveness of JDA compared to other methods, including in particular TCA, on image classification problems.

In 2010, Ben-David *et al.* proposed a theoretical work [5] to answer the important question in unsupervised domain adaptation: "under what conditions can a classifier trained from source data be expected to perform well on target data?" They address this question by giving a theoretical bound on a classifier's error on target data, which depends on its error on the source data and the divergence between source and target data. The bound is defined as follows (Theorem 2 in [5]): "Let $\mathcal{H}$ be a hypothesis space of VC dimension d. If $\mathcal{U}_S, \mathcal{U}_T$ are unlabeled samples of size m' each, drawn from source domain and target domain respectively, then for any $\delta \in (0, 1)$, with probability at least $1 - \delta$ (over the choice of the samples), for every $h \in \mathcal{H}$":

$$\epsilon_T(h) \leqslant \epsilon_S(h) + \frac{1}{2}\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_S, \mathcal{U}_T) + 4\sqrt{\frac{2d\log(2m') + \log(\frac{2}{\delta})}{m'}} + \lambda \tag{8}$$

Where $\mathcal{H}\Delta\mathcal{H}$ is the "*symmetric difference hypothesis space*" for the hypothesis space $\mathcal{H}$. And $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_S, \mathcal{U}_T)$ is the $\mathcal{H}\Delta\mathcal{H}$-divergence defined for the symmetric hypothesis class $\mathcal{H}\Delta\mathcal{H}$.

From the authors' analysis we could learn that: the performance of a hypothesis h on the target domain is mostly bounded by three terms: (1) the performance of this hypothesis on the source domain (the first term in Eq. 8), (2) the data divergence between the two domains (the second term in Eq. 8) and (3) the coherence of the hypothesis functions across the two domains (the last term in Eq. 8). In previous research works which we have introduced, *e.g.* TCA [47] and JDA [38], the authors only seek to minimize the second term of Eq.8, *i.e.* the difference between data distributions. In a recent work [44], the authors propose to also minimize the last term in Eq.8 by enhancing discriminativeness of the joint feature space and by performing geometric alignment of the underlying data manifold structures across source and target domains. They propose a novel method named *Discriminative and Geometry Aware Domain Adaptation (DGA-DA)*.

Based on the framework of JDA [38], DGA-DA [44] further add a *repulsive force term* to increase the distance of sub-domains with different labels, and two additional consistency constraints, *i.e. label smoothness consistency (LSC)* and *geometric structure consistency (GSC)*, in order to preserve the hidden data geometric structure across different domains.

The authors performed extensive experiments for 49 image classification DA tasks on 8 popular DA benchmarks to verify the effectiveness of the proposed DGA-DA method. They also carried out analysis of DGA-DA *w.r.t.* its hyper-parameters and the convergence speed. In addition, using both synthetic and real data, the authors provide some illustrations for visualizing the effect of data discriminativeness and geometry awareness.

Learning a shared feature space with metric learning:

An alternative way to learn a shared feature space is to cast the representation learning problem into the metric learning scenario. In metric learning, we learn a new metric which defines the distance between two samples in the input sample space. This distance could be used to measure the discrepancy between source and target distributions. For example, Ding and Fu develop a "robust transfer metric learning (RTML)" framework in [17] for unsupervised domain adaptation.

Suppose having a source training set $\mathbf{X}^{(S)} = \{\mathbf{x}_1^{(S)}, \ldots, \mathbf{x}_{n^{(S)}}^{(S)}\}$ of labeled samples from C categories, and a target set $\mathbf{X}^{(T)} = \{\mathbf{x}_1^{(T)}, \ldots, \mathbf{x}_{n^{(T)}}^{(T)}\}$ of unlabeled samples, where $\mathbf{x}_i^{(S)}, \mathbf{x}_i^{(T)} \in \mathbb{R}^d$.

The objective function of RTML is defined as follows:

$$\min_{\mathbf{M}\in\mathbb{S}_+^d} \sum_{c=0}^{C} \operatorname{trace}(\mathbf{\Phi}^{(c)}\mathbf{M}) + \alpha\|\bar{\mathbf{X}} - \mathbf{M}\tilde{\mathbf{X}}\|_F^2 + \lambda \sum_{i=r+1}^{d} (\sigma_i(\mathbf{M}))^2 \tag{9}$$

where $\mathbf{\Phi}^{(c)}$ is the difference between the mean of the source samples labeled to the c-th category and the mean of the target samples with pseudo labels belonging to the c-th category. $\mathbf{M} \in \mathbb{R}^{d\times d}$ is the positive semi-definite ($\mathbf{M} \in \mathbb{S}_+^d$) distance metric to be learned. The second term in Eq.(9) is a denoising term for preserving energy of the two domains. The second term in Eq.(9) is a denoising term for preserving energy of the two domains. $\mathbf{X} = [\mathbf{X}^{(S)}, \mathbf{X}^{(T)}]$, $\bar{\mathbf{X}}$ is the m-times repeated version of $\mathbf{X}$, and $\tilde{\mathbf{X}}$ is the corrupted version of $\bar{\mathbf{X}}$. (See "marginalized Denoising Auto-Encoder (mDAE)" [10] and "Denoising Auto-Encoder (DAE)" [70] for details) The last term is a regularization term which controls the rank of $\mathbf{M}$ to not be larger than r. The optimization is performed as follows: $\mathbf{M}$ and the pseudo labels of the target data are refined alternatively in iterations, *i.e.* optimizing one while fixing the others, until the metric $\mathbf{M}$ converges.

Since $\mathbf{M}$ can be rewritten as $\mathbf{M} = \mathbf{P}\mathbf{P}^\top$, where $\mathbf{P} \in \mathbb{R}^{d\times r}$ and $r \leqslant d$ is the rank of the metric $\mathbf{M}$, the distance defined by this metric $d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)$ can be rewritten as $\|\mathbf{P}^\top(\mathbf{x}_i - \mathbf{x}_j)\|_2$. This shows that the metric learning could actually be

considered as learning an underlying subspace, and the distance defined by this metric equals Euclidean distance in this new subspace.

3.3.2 *Learning separate feature mappings .*

The way to learn a shared feature space for DA demands that the source and target data distributions be enough similar to each other. If not, there might not exist such a common feature transformation that projects two distinct data distributions into two nearby feature distributions. To relax this assumption, one could assume that the source and target data distributions lie in two different subspaces and then find a way to align these two subspaces to achieve adaptation. In the following we show some different ways to make this subspace alignment possible.

Subspace Alignment with dimensionality reduction methods:

Similar to the methods introduced in section 3.3.1, Fernando *et al.* also make use of dimensionality reduction methods for subspace learning [20]. The difference is that, they propose to use two PCAs as dimension reduction on both source and target domain, respectively. Following theoretical recommendations in [5] (see section 3.3.1), this method designs two different subspaces to represent the two different domains, rather than to drag different domains into a common shared subspace. They optimize a mapping function to transform the learned source subspace to the target subspace. In their proposed Subspace Alignment (SA) method, a novel similarity function $Sim(\mathbf{x}^{(S)}, \mathbf{x}^{(T)})$ is defined (with the optimized mapping function) for comparing a source sample $\mathbf{x}^{(S)}$ with a target sample $\mathbf{x}^{(T)}$, this $Sim(\mathbf{x}^{(S)}, \mathbf{x}^{(T)})$ could be used directly in a k-nearest neighbor classification model. An alternative solution is to firstly project the source data into the learned target subspace with the learned mapping between source and target, and project the target original data into the learned target subspace, then learn a SVM classifier in the target new subspace. This work is further improved to a tensor version in [42].

As in SA the source data and the target data are processed separately using their own corresponding feature transformations, and the resulting two different feature spaces are only aligned by their principle components, the variances of the source and target data in the two spaces are not aligned. In this case, the distribution mismatch between the source and target are actually not well minimized with SA. Another problem is that, SA could not handle the situations where the mapping between the two projected spaces is nonlinear. To solve these problems, the method "Subspace Distribution Alignment (SDA)" [60] improves SA by taking into account the variance of the principal components, and "Joint Geometrical and Statistical Alignment (JGSA)" [78] further improves the performance by reducing the mismatch between source and target both statistically and geometrically. To achieve its goal, JGSA also tries to find two feature projections for source data and target data. While instead of aligning the two new data spaces with a third mapping, it minimize the divergence between data distributions in the new spaces using the same way as in JDA [38] and it adapts Fischer discriminant criteria to maximize the variance of target data and preserve the source discriminative information. Similar to JDA, they also make use of pseudo-labels on target data and they also update alternatively the pseudo labels and the learned mappings to improve the final prediction performance until convergence.

Subspace Alignment with Optimal Transportation:

A different way to achieve subspace alignment is to learn the mapping between source and target subspaces as an optimal transportation [13] [14] [50] [12].

Optimal transportation (OT) [68] is a well explored mathematical problem which calculates the distance between two distributions. It considers the distance between two distributions as the minimum effort one should spend to

"transport" the mass in one distribution to fit the other one. Recently, as some fast calculation methods for OT have been proposed (*e.g.* [15]), OT based distance (also called Wasserstein distance, Earth-Mover's distance, *etc.*) has been more and more applied in machine learning and vision problems, especially in solving domain adaptation problems. Most works (*e.g.* [13] [14] [50]) which make use of OT methods for solving domain adaptation problems consider finding the optimal transportation between source and target distributions, and then map the source data to the target distribution (or inversely) by explicitly doing a barycenter mapping. A recent work [12] considers on the contrary that the optimal transportation between source and target distributions is underlying and does not need to be found explicitly. They focus on estimating the target prediction function while at the same time learning the underlying OT based transformation between the source and target distributions.

3.3.3 Deep learning methods for Domain Adaptation.

The last years have seen breakthroughs enabled by deep learning in an increasing number of domains, in particular for various vision recognition tasks. Recent studies show that deep neural networks (DNN) can also learn transferable features which can be well generalized to new tasks. Therefore, more and more works tend to rely upon Deep Neural Networks to solve DA problems. The basic idea is similar to those introduced previously in Sect.3.3.1 and Sect.3.3.2. They make use of DNN as the structure for feature learning, and they add regularization on the new feature space either by using conventional discrepancy measures [66] [41], or by adding an adversarial network as a discrepancy measure [22] [64] on one or multiple intermediate layers' outputs to match the source and target distributions. We will introduce in detail two representative groups of methods in the following part of this section.

Deep Adaptation with MMD criterion:

In [41], Long *et al.* proposed a "Deep Adaptation Network (DAN)" architecture. Similar to previously introduced JDA in Sect.3.3.1, DAN also uses Maximum Mean Discrepancy (MMD) as distance measurement for adapting source and target distributions. Specifically, they used a multi-kernel version of MMD, named MK-MMD as proposed by [27], which gives better performance than MMD and also minimizes the error of rejecting a false null hypothesis.

To learn an optimal feature transformation, the authors propose to extend the AlexNet architecture [30]. Letting Θ denote the set of parameters in the neural network, the objective of learning a CNN is to minimize its empirical risk:

$$\min_{\Theta} \frac{1}{n} \sum_{i=1}^{n} L(h(\mathbf{x}_i), \mathbf{y}_i) \tag{10}$$

where L is the cross-entropy loss function, $h(\mathbf{x}_i)$ is the predicted label vector for input $\mathbf{x}_i$, and $\mathbf{y}_i$ the ground-truth label vector for $\mathbf{x}_i$. According to [75], the convolutional layers in a CNN learn transferable features in the first layers and less transferable features in the middle layers, while the last layers are more domain specific. The authors therefore propose to fix the parameters in the pre-learned *conv1-conv3* layers, fine-tune these in *conv4 – conv5* layers, and retrain the parameters in *fc6 – fc8* layers. At the same time they add an MK-MMD based adapation regularizer to the objective in Eq. (10) on multiple layers' outputs to require the source and target distributions be as close to each other as possible in the hidden feature space. The adaptation regularizer is defined as follows:

$$\min_{\Theta} \frac{1}{n} \sum_{i=1}^{n} L(h(\mathbf{x}_i), \mathbf{y}_i) + \lambda \sum_{l=l_1}^{l_2} d_k^2(\mathbf{X}^{(l)(S)}, \mathbf{X}^{(l)(T)}) \tag{11}$$

where $\lambda > 0$ controls the balance between the CNN loss and the regularization, l_1 and l_2 are layer indexes between which the regularization is effective (in DAN the authors set $l_1 = 6$ and $l_2 = 8$). $d_k^2(\mathbf{X}^{(l)(S)}, \mathbf{X}^{(l)(T)})$ is the MK-MMD between the source and target distributions based on the l-th layer's output.

The authors propose to initialize the parameters in DAN with those of the AlexNet model pre-trained on ImageNet dataset [56]. The training process is therefore a fine-tuning of this pre-trained model on source and target training data.

This work is further improved in [39] and [40]. Since in DAN only the marginal distributions of source and target data are adapted, in [39] the authors proposed a Residual Transfer Network which can adapt both marginal distributions and conditional distributions at the same time. In [40] the authors further propose a joint maximum mean discrepancy (JMMD) criterion in order to adapt the joint distributions (feature+label) of source and target data, the resulting model is named Joint Adaptation Network (JAN). A comparison of these methods to other DA methods is shown in Table 5.

Deep Adaptation with Adversarial Networks:

Another group of methods are those who make use of adversarial networks, instead of MMD used in previously introduced works, to reduce the distribution discrepancy between source and target feature data (as shown in figure 6). These methods are inspired by the recent research trend on generative adversarial networks (GANs) for unsupervised learning [26]. A generative adversarial network is usually build by two parts: one generative part (*i.e.* the generator) and one discriminative part (*i.e.* the discriminator). The generator's objective is to generate samples that are as close as possible to the training samples, while the discriminator's objective is to distinguish between the generated samples and the real training samples. The two parts are adversarial. When the two parts are trained alternatively, the network can get to a balanced situation where both the two parts are well trained. In GANs, the discriminator plays the role as a distance measure, it evaluates the distance between the distribution of the generated samples and that of the real samples. That's why the idea of adversarial networks could be applied to solve domain adaptation problems, simply by adopting the discriminator as a replacement to traditional distance measures (*e.g.* MMD). In [65] the authors propose a general framework to describe these kind of methods, in the following we show that this general framework corresponds to our categorization method for traditional DA methods.

The main goal of adversarial adaptive methods is to regularize the optimization of the source and target feature mappings, $f^{(S)}$ and $f^{(T)}$, so as to minimize the discrepancy between the source and target feature distributions: $\mathbf{X}^{(S)(f^{(S)})} = f^{(S)}(\mathbf{X}^{(S)})$ and $\mathbf{X}^{(T)(f^{(T)})} = f^{(T)}(\mathbf{X}^{(T)})$. When the distance between two distributions is minimized, a classifier $f^{c(S)}$ learned on source data could then be applied for classification on target data.

An adversarial adaptation approach could then be determined by answering the following questions: (1) are the source mapping and target mapping generative or discriminative model? (2) are the weights of source and target mappings shared or not? (3) which kind of adversarial objective is used?

The first question asks about the parameterization of the source and target mappings, it equals the choice of feature extractions methods in traditional DA methods (*e.g.* PCA or FLDA, *etc.*), except for that most DA methods use discriminative mappings since DA problems generally consider discriminative tasks. While generative mappings are also possible to be used for solving DA problems and they are explored in some recent works (*e.g.* [37]), in these works they use random noise as input to the generative network mapping to get output samples, an intermediate output of the mapping is used as feature for training task specific classifier. The second question equals the choice of shared feature space or separate feature spaces with subspace/distribution alignment in DA methods. The third question equals the choice of distance measure for evaluating the discrepancy between source and target distributions. By answering these three questions, we could then categorize these adversarial domain adaptation methods as in the bottom part of Table 5.

Table 5. Comparison of domain adaptation methods (In the last column, 'M' stand for Marginal, 'C' stand for Conditional, 'J' stand for Joint, 'S' stand for Source and 'T' stand for target)

Method	Mapping method	Shared mapping	Discrepancy measure	Minimize Distance between
TSL [58]	dimension reduction	yes	Bregman divergence	M distributions
MMDE [46]	PCA	yes	MMD	M distributions
TCA [47]	PCA	yes	MMD	M distributions
JDA [38]	PCA	yes	MMD	M and C distributions
DGA-DA [44]	PCA	yes	MMD	M and C distributions
RTML [17]	underlying	yes	learned metric	M and C distributions
SA [20]	PCAs	no	distance between Principle components	S and T subspaces
TSA [42]	Tucker Decomposition	no	distance between Principle components	S and T subspaces
SDA [60]	PCAs	no	distance between Principle components	S and T subspaces
JGSA [78]	PCAs	no	distance between Sample means	M and C distributions
DA-ROT [13]	Barycenter mapping	no	Optimal transport	M distributions
OT-DA [14]	Barycenter mapping	no	Optimal transport	M and C distributions
JDOT [12]	underlying	no	Optimal transport	J distributions
ME-DOT [50]	Barycenter mapping	no	Optimal transport	M distributions
DAN [41]	Discriminative NN	yes	MK-MMD	M distributions
RTN [39]	Discriminative NN	yes	MK-MMD	M and C distributions
JAN [40]	Discriminative NN	yes	MK-MMD	J distributions
Gradient reversal [23]	Discriminative NN	yes	Adversarial NN (Mini-max)	M distributions
Domain confusion [64]	Discriminative NN	yes	Adversarial NN (Confusion)	M distributions
CoGAN [37]	Generative NN	no	Adversarial NN (GAN)	M distributions
ADDA [65]	Discriminative NN	no	Adversarial NN (GAN)	M distributions

3.3.4 *Relaxation on shared label space assumption.*

One of the assumptions of DA, that source and target task share a same label space, restricts its application in reality. When outlier classes appear in source or target data, it will make the equal class number adaptation difficult. Therefore, recently several works start to study the case without this shared label space assumption. The main idea of these works is to identify the outliers and only do adaptation on classes shared by source and target.

For example, in [7] the authors proposed the 'open set Domain Adaptation' problem, where only a few categories of interest are shared between source and target (outliers exist in both source and target). To deal with this problem, they learn a mapping from source to target, which maps source samples to be close to target distribution. This is done iteratively: first assign pseudo class labels to a part of the target samples, then minimize the distance between the target

and source samples which have the same label. The assignment problem is defined by a binary linear program that also includes an implicit outlier handling, which will not assign labels to images that are not related to any source domain images.

In [77] the authors deal with a similar problem *partial domain adaptation*, where the source domain has more classes than the target domain. They extend the adversarial neural networks based DA methods for finding the source samples which are from the outlier classes and at the same time reduce the discrepancy between the source and target distributions for the shared classes. In [8] the authors also deal with the partial domain adaptation problem. They propose Selective Adversarial Network (SAN), which selects out the outlier source classes and maximally matches the data distributions in the shared label space.

4 KNOWLEDGE TRANSFER FROM PRE-LEARNED MODEL

In this section we are going to introduce transfer learning works that reuse knowledge from previously learned source tasks. Compared to those introduced in section 3, these methods reuse knowledge from parameters (or outputs) of a model pre-learned for source task. Since they avoid learning from scratch with source raw data, these methods are usually more efficient in training. While as the pre-learned model is more adapted to source task, it makes the result of knowledge transfer more sensitive to distance between target and source.

4.1 Knowledge transfer from feature extractor parameters

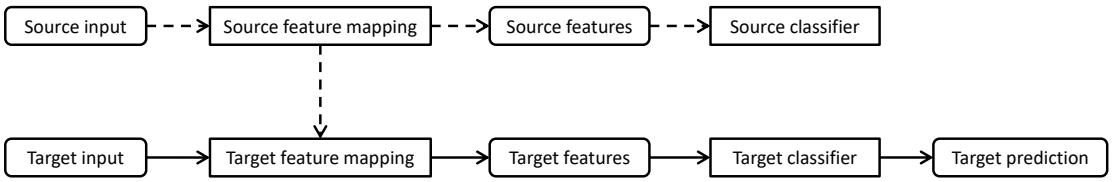

Fig. 7. Illustration of Knowledge Transfer from feature extractor parameters

The very first group of methods are those who reuse a pre-learned feature extractor for new target tasks.

4.1.1 *Transferable feature learning with shallow Neural Networks and linear transformations.*

One of the pioneer works on knowledge transfer is [62], in which Thrun proposed the concept of *lifelong learning*. He proposed an algorithm, which makes use of source data to learn a feature mapping, denoted by $g : I \rightarrow I'$, and then apply this learned mapping for feature extraction on target data to help the classification of the target task. The learning objective of the feature mapping is to make every pair of positive samples stay close to each other and every pair of positive and negative samples stay far from each other in the new feature space. In this way, the learned new feature representation is considered discriminative for source data, and hopefully also be discriminative for target data.

The objective function is defined as follows with $\mathcal{P}_k$ being the set of positive samples and $\mathcal{N}_k$ being the set of negative samples for the k-th task:

$$\min_{g\in\mathcal{G}} \sum_{k=1}^{m} \sum_{\mathbf{x}_i\in\mathcal{P}_k} \Big(\sum_{\mathbf{x}_j\in\mathcal{P}_k} \| g(\mathbf{x}_i) - g(\mathbf{x}_j) \| - \sum_{\mathbf{x}_j\in\mathcal{N}_k} \| g(\mathbf{x}_i) - g(\mathbf{x}_j) \| \Big) \tag{12}$$

In this objective, the candidate feature mapping set $\mathcal{G}$ is the set of two layers neural networks. The learned feature mapping $g(\cdot)$ is then applied for feature extraction on target data. Then a nearest neighbor classifier is learned with the new target representations for target classification task.

Thrun has perform experiments on an object recognition task to show that the use of the transferable feature mapping on target task could improve the performance of the target task when there exist only a small number of labeled target training samples.

This work is further generalized by several authors [2] [3] [1]. The three works can all be considered as special cases of the framework of '*structural learning*' proposed in [2]. And [2] is further applied to image classification in [53].

4.1.2 *Transferable feature learning with metric learning.*

As shown in section 3.3.1, metric learning could provide same effect as subspace learning. Therefore transferable/shared feature learning could also be done with metric learning methods.

One of the first works is [21], which tries to learn a shared feature representation using metric learning disciplines. Similar to the very first transfer learning method [62], this algorithm learns a feature transformation which is later taken as input by a nearest neighbor classifier for the target task. Unlike Thrun's transfer algorithm which deploys a neural network to learn the feature transformation, Fink's transfer algorithm make use of a max-margin approach to directly learn a distance metric.

Like the early works introduced in subsection 4.1.1, the strong assumption which requests the source data to be very close to the target restricts the effectiveness of this method for more general situations. In [49] a new method is introduced which combines the large margin nearest neighbor classification with the multi-task learning paradigm. Unlike the previously introduced method, this method learns a specific metric $d_t(\cdot,\cdot)$ for each of the T tasks. They then model the commonalities between various tasks through a shared Mahalanobis metric with $\mathbf{M}_0 \succeq 0$ and the task-specific characteristics with additional matrices $\mathbf{M}_1, \ldots, \mathbf{M}_T \succeq 0$. The distance for task t is defined as follows:

$$d_t(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{M}_0 + \mathbf{M}_t)(\mathbf{x}_t - \mathbf{x}_j)} \tag{13}$$

Although there is not a specific projection as θ defined in [21], this distance defined in Eq. (13) could still be considered as a distance in an underlying new feature space. The metric defined by $\mathbf{M}_0$ picks up general trends across multiple data sets and $\mathbf{M}_{t>0}$ specialize the metric further for each particular task.

4.1.3 *Transferable feature learning with Deep Neural Networks.*

With the growth of deep learning techniques, deep neural networks have shown great success on learning transferable features, especially for visual data like images. Actually in some of the knowledge transfer methods we have introduced above (*e.g.* the methods introduced in section 3.1) where the focus is on transferring knowledge to classifier parameters,

if we make use of a pre-trained deep neural network for feature extraction, the performance will get a significant improvement comparing to the same method with traditional feature extraction technique (*e.g.* [43]). Nowadays, fine-tuning a deep neural network model pre-trained on some large scale source dataset for a small customized target dataset has already become the most popular way to do knowledge transfer. This kind of deep transferable feature learning is analyzed in [75].

4.2 Knowledge transfer from classifier parameters

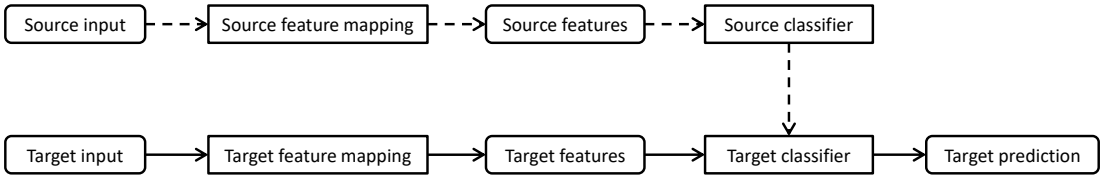

Fig. 8. Illustration of Knowledge Transfer from source classifier parameters to target classifier parameters

Apart from feature extractor parameters, classifier parameters of a pre-learned model could also be reused for learning a new target task. In this section we show two groups of methods, one is based on discriminative classification models (*e.g.* SVM), the second one is based on generative classification models.

4.2.1 *Knowledge transfer from discriminative models.*

Support Vector Machine (SVM) is a supervised discriminative learning method which learns the conditional distribution of labels on knowing input features. Several early works on knowledge transfer from model parameters are constructed based on the SVM classifier. A common form of the objective function of these SVM based transfer learning models could be expressed as follows:

$$\min_{\mathbf{w}^{(T)},b} \Phi(\mathbf{w}^{(T)}) + C \sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}_T} \varepsilon(\mathbf{x}_i, y_i; \mathbf{w}^{(T)}, b) \tag{14}$$

where $\sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}_T} \varepsilon(\mathbf{x}_i, y_i; \mathbf{w}^{(T)}, b)$ is the loss on labeled samples in the target data set $\mathcal{D}_T$, and $\Phi(\mathbf{w}^{(T)})$ is the regularization on model parameter $\mathbf{w}^{(T)}$ which enforces the margin maximization and the knowledge transfer. The knowledge transfer regularization is usually expressed as a minimization of the distance between the pre-learned source parameter $\mathbf{w}^{(S)}$ and the target parameter $\mathbf{w}^{(T)}$.

One of the first SVM based transfer learning works is the Adaptive-SVM (A-SVM) proposed in [72], in which the decision function for the target classification task is formulated as follows:

$$f^{(T)}(\mathbf{x}) = f^{(S)}(\mathbf{x}) + \Delta f(\mathbf{x}) \tag{15}$$

where $f^{(S)}(\cdot)$ is the source decision function and $\Delta f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x})$ is the perturbation function. The perturbation function $\Delta f(\cdot)$ is learned with the target labeled data $\mathcal{D}_T$ and the pre-learned parameters for the source decision function $f^{(S)}(\cdot)$, the objective function is defined as follows:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\varepsilon_i$$
$$\text{s.t. } \varepsilon_i \geqslant 0,\ y_i f^{(S)}(\mathbf{x}_i) + y_i \mathbf{w}^\top \phi(\mathbf{x}_i) \geqslant 1 - \varepsilon_i,\ \forall(\mathbf{x}_i, y_i) \in \mathcal{D}_T \tag{16}$$

where $\sum_i \varepsilon_i$ measures the total classification error of the adapted classifier $f^{(T)}(\cdot)$ in the target domain. The first term in (16) minimizes the deviation between the target decision boundary and the source decision boundary. The cost factor C controls the contribution balance between the source classifier and the target training examples, *i.e.* the larger C is, the smaller the contribution of the source classifier is.

This work was further improved in [4] for object detection. Aytar and Zisserman fistly show a more general form of the objective function for A-SVM(this form was firstly introduced in [36]) as follows:

$$\min_{\mathbf{w}} \|\mathbf{w}^{(T)} - \Gamma\mathbf{w}^{(S)}\|^2 + C\sum_{i=1}^{N}\varepsilon_i \tag{17}$$

where $\mathbf{w}^{(S)}$ and $\mathbf{w}^{(T)}$ are the parameters for source classifier and target classifier respectively. Γ controls the amount of knowledge transfer. The authors have shown that Γ controls the trade-off between margin maximization of the target classifier and the knowledge transfer from source classifier, *i.e.* the larger Γ is, the larger the knowledge transfer is, while the smaller the margin maximization is.

To avoid this trade-off, [4] propose the projective Model Transfer SVM (PMT-SVM), in which they can increase the amount of knowledge transfer without harm on the max-margin objective. The objective function is defined as follows:

$$\min_{\mathbf{w}^{(T)}} \|\mathbf{w}^{(T)}\|^2 + \Gamma\|P\mathbf{w}^{(T)}\|^2 + C\sum_{i=1}^{N}\varepsilon_i,\ \ \text{s.t. } (\mathbf{w}^{(T)})^\top\mathbf{w}^{(S)} \geqslant 0 \tag{18}$$

where $P = I - \frac{\mathbf{w}^{(S)}(\mathbf{w}^{(S)})^\top}{(\mathbf{w}^{(S)})^\top\mathbf{w}^{(S)}}$ is the projection matrix, Γ controls the amount of knowledge transfer, and C controls the weight of the loss function $\sum_i \varepsilon_i$. $\|P\mathbf{w}^{(T)}\|^2 = \|\mathbf{w}^{(T)}\|^2 \sin^2\theta$ is the squared norm of the projection of the $\mathbf{w}^{(T)}$ onto the source hyperplane, θ is the angle between $\mathbf{w}^{(S)}$ and $\mathbf{w}^{(T)}$. $(\mathbf{w}^{(T)})^\top\mathbf{w}^{(S)} \geqslant 0$ constrains $\mathbf{w}^{(T)}$ to the positive half-space defined by $\mathbf{w}^{(S)}$. Experimental results have shown that this PMT-SVM works better compared to A-SVM and SVM when having only a few labeled samples in target domain, especially for one-shot learning when only one labeled sample is available.

In [4] the authors also show a direct generalization of A-SVM to deformable transfer formulation, named Deformable Adaptive SVM (DA-SVM), for object detection with deformable part based models.

In [29] the A-SVM was improved for visual concept classification, where the authors propose the cross-domain SVM (CD-SVM). In CD-SVM they define a weight for each source pattern depending on their distance to the target using the k-nearest neighbors method, and then they train the target SVM classifier with the help of the re-weighted source patterns.

Tommasi *et al.* [63] proposed a multi-model knowledge transfer algorithm based on the Least Square SVM (LS-SVM). The objective function of the multi-model knowledge transfer is defined as follows:

$$\min_{\mathbf{w}, b} \frac{1}{2}\|\mathbf{w} - \sum_{j=1}^{k}\beta_j\mathbf{w}_j\|^2 + C\sum_{i=1}^{N}\varepsilon_i \tag{19}$$

where $\mathbf{w}$ is the parameter of the target model and $\mathbf{w}_j$ are the parameters of the pre-learned source models, the coefficient vector $\boldsymbol{\beta}$ should be chosen in the unitary ball, *i.e.* $\boldsymbol{\beta} \leqslant 1$. The second term in Eq. (19) is the least square loss for target training samples. The optimal solution of Eq. (19) is as follows:

$$\mathbf{w} = \sum_{j=1}^{k} \beta_j \mathbf{w}'_j + \sum_{i=1}^{N} \alpha_i \phi(\mathbf{x}_i) \tag{20}$$

where $\mathbf{w}$ is considered as the sum of a weighted sum of the source models and a new model learned on target training data. The new model parameters α_i could be learned on the target training data. The optimal coefficients β_j could be found by minimizing the LOO (leave one out) error, which estimates the generalization error of classifiers and could be used for model selection [9].

In [32] the authors extend this LSSVM based transfer learning to an incremental transfer learning setting, where the source is a pre-learned multi-class classifier for N classes, denoted as $\mathbf{W}' = [\mathbf{w}'_1, \ldots, \mathbf{w}'_N]$, and the target training set contains samples from the N known classes and a new unknown class. Their aim is to find a new set of hyperplanes $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_N]$ and $\mathbf{w}_{N+1}$, such that: (1) the classification performance on the new target class could be improved by transferring knowledge from the known source class models, and (2) the classification performance on the known N source classes should be maintained or even improved compared to the learned models.

They achieve the first goal by using the regularizer $\|\mathbf{w}_{N+1} - \mathbf{W}'\boldsymbol{\beta}\|^2$, which "enforces the target model $\mathbf{w}_{N+1}$ to be close to a linear combination of the source models." To achieve the second objective, they enforce the new source model parameters $\mathbf{W}$ to be close to the learned source parameters $\mathbf{W}'$ with the term $\|\mathbf{W} - \mathbf{W}'\|_F^2$. The final objective function with the two regularizers is as follows:

$$\min_{\mathbf{W}, \mathbf{w}_{N+1}, \mathbf{b}} \frac{1}{2}\|\mathbf{W} - \mathbf{W}'\|_F^2 + \frac{1}{2}\|\mathbf{w}_{N+1} - \mathbf{W}'\boldsymbol{\beta}\|_F^2 + \frac{C}{2}\sum_{i=1}^{M}\sum_{n=1}^{N+1}(\mathbf{W}_n^\top \mathbf{x}_i + b_n - Y_{i,n})^2 \tag{21}$$

where $\mathbf{Y}$ is the label matrix, $Y_{i,n}$ is equal to 1 if $y_i = n$ and is equal to -1 otherwise.

As can be seen from the above, most SVM based transfer learning methods enforce knowledge transfer simply by adding a regularization term to minimize the distance between the target model parameters and the source model parameters. This brute-force regularization work well for binary-classification when target positive category and source positive category are as close as possible. The extension to multi-class classification could be done in a one-vs-all manner as shown in [32], and the negative transfer is mainly prevented by tuning the parameter $\boldsymbol{\beta}$ (which could be seen as a model selection process).

Another group of methods which transfers knowledge from source models is the so-called *hypothesis transfer learning* [31] [51]. In these methods they assume not having access to source data but only having access to hypothesis induced from source domain. For example in [51] the authors have explored the setting *Metric Hypothesis Transfer Learning*, in which they assume that the source training samples are not accessible so one can only make use of the pre-learned source metric $\mathbf{M}_S$ to help learning the target metric $\mathbf{M}$. They have mainly provided some theoretical analysis and guarantees for transfer metric learning with a biased regularization term. They propose a new stability notion called *on-average-replace-two-stability*, which measures the stability of an algorithm when suffering from a little change in its input. Then based on this, they prove that the metric hypothesis transfer learning can achieve a fast converge rate with a high probability generalization bound under certain conditions. For the weighted biased regularization term they use, *i.e.* $\|\mathbf{M} - \beta\mathbf{M}_S\|$, they propose an approach to set the parameter β rather than tune it with brute-force search. As can be seen, this metric hypothesis transfer learning uses a similar regularization term as what is used in SVM-based transfer

learning methods.

4.2.2 *Knowledge transfer from generative models.*

Another kind of classification methods are generative models, which learn the joint distribution of the labels and input features. Generative models are also adopted for knowledge transfer, especially in the case of zeros-shot or one-shot learning for object recognition, where no target sample or only one target sample is given for training an object recognition model.

A representative work is proposed in [19], which is a Bayesian-based unsupervised one-shot learning framework for object categorization. This work is based on the *constellation model* [6], where an object model consists of several parts and each part is described by its appearance. The shape of the object is described by the relative positions between each parts. The appearances and relative positions are modeled by probability density functions (*e.g.* Gaussians). The objective in training step is to estimate the model distribution parameters conditioned on training samples. This could be done by using the Variational Bayes Procedure, which approximates the desired distribution using an EM like iterative updating strategy. This procedure allows incremental learning, therefore one could take an object model pre-learned on source object samples, and update its parameters with new training samples from target object for learning a new target model.

Another work is [76], where the authors propose a generative attribute model for zero-shot and one-shot learning. In their proposed framework, one category is associated with a list of attributes. They build generative models, which are considered as attribute priors, to describe the probabilistic distributions of image features for all the attributes. Then for zero-shot or one-shot learning, one could classify images from unseen categories just by using their corresponding attribute lists and the pre-learned attribute priors.

As can be seen, knowledge transfer with generative models reuse model parameters pre-learned on source data and usually demand some prior knowledge on target data. For example, in [19] they assume that the target objects could be expressed by a pre-defined constellation model with fixed number of object parts, in [76] they demand the attribute list information is available for target categories.

Nowadays, since deep neural networks have taken over most vision recognition problems, there have not been much recent research works on knowledge transfer from traditional model parameters. However, a new research direction on transferring knowledge from deep neural networks, which could be seen as an extend to model parameter transfer, is growing very fast. In the next section we will introduce this kind of 'knowledge distillation' methods.

4.3 Knowledge distillation for DNN (Knowledge from both feature extractor and classifier parameters)

A special group of works is the knowledge distillation methods for Deep Neural Networks [28] [55] [11] [74] [59]. As its name 'distillation' suggests, these works try to learn a small student network which could give equal performance as a big teacher network. These works not only benefit knowledge transfer from a big pre-learned DNN, but also try to make knowledge more compact by putting the transferred knowledge into a new DNN with smaller amount of parameters.

The concept "knowledge distillation (KD)" is firstly introduced in [28], in which the authors propose the "teacher-student" framework. They make use of a softened version of the output of a big neural network, which is considered as the "teacher network", to teach a smaller neural network, which is considered as the "student network", to give the

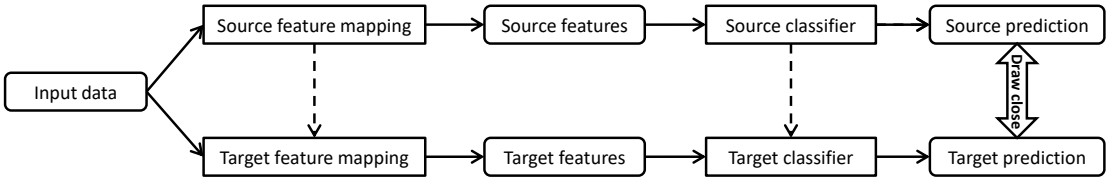

Fig. 9. Illustration of Knowledge Distillation

same output. In this way, the knowledge saved in the big "teacher network" could be compressed/distilled into the smaller "student network". In [55] the authors further make use of the intermediate output of the "teacher network" for training a "deeper and thinner student network". The learned student network can even give better performance than the teacher network and at the same time run faster than the teacher network thanks to the smaller number of parameters. Net2Net [11] make use of a similar "teacher-student" system, while instead of transferring knowledge from a bigger teacher network to a smaller student network, Net2Net focus on transferring knowledge to a "deeper and wider" student network in order to make the initialization and training of the big student network more efficient. Inspired by image style transfer with neural networks [24], which make use of Gram matrix to represent the global style of an image, [74] also tries to make use of Gram matrix to represent the knowledge of how neural nets solve a problem. The Gram matrix is calculated by doing inner products between outputs of two layers, and the student network is trained to generate similar Gram matrices as the teacher network in order to learn the knowledge from the teacher network.

Unlike other methods introduced above, these methods do not focus on adapting a model learned on source dataset to a new model for a different target dataset, they rather consider two different model for a same data distribution. However we still consider these methods as a kind of knowledge transfer since they consider transferring knowledge from one model to another one, and they have the potential to be extended to a normal transfer learning scenario with different source and target data distributions.

5 CONCLUSION

In this paper we have introduced a new methodology to describe and categorize knowledge transfer methods for vision recognition problems. Unlike existing surveys for transfer learning, we focus on where the knowledge come from and where the knowledge go to. Based on this principle, we derive six major categories as shown in table 3, each containing detailed sub categories. For each category we introduce its basic idea by illustration and description and we also introduce some representative works related to this category. Based on our methodology, we could also induce some potential research directions which are not yet explored in this domain. For example, in section 3.2 we show that unlabeled source data selection for learning transferable feature extractor could be a future research direction. And the methods in different major categories shown in table 3 could possibly be combined to benefits more from knowledge transfer. For example, the methods in section 4.1 or section 3.2 could be combined with methods in section 3.1 or section 4.2. We hope this paper could be a good guide for researchers to get an overview about knowledge transfer in vision recognition, to find suitable methods for their applications, or to find directions for their future research.

REFERENCES

[1] Amit, Y., Fink, M., Srebro, N., and Ullman, S. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th international conference on Machine learning* (2007), ACM, pp. 17–24.

[2] Ando, R. K., and Zhang, T. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research 6*, Nov (2005), 1817–1853.

[3] Argyriou, A., Evgeniou, T., and Pontil, M. Multi-task feature learning. In *Advances in neural information processing systems* (2007), pp. 41–48.

[4] Aytar, Y., and Zisserman, A. Tabula rasa: Model transfer for object category detection. In *Proc. 2011 Int. Conf. Computer Vision* (Washington, DC, USA, 2011), IEEE Computer Society, pp. 2252–2259.

[5] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. A theory of learning from different domains. *Machine learning 79*, 1 (2010), 151–175.

[6] Burl, M. C., and Perona, P. Recognition of planar object classes. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on* (1996), IEEE, pp. 223–230.

[7] Busto, P. P., and Gall, J. Open set domain adaptation. In *The IEEE International Conference on Computer Vision (ICCV)* (2017), vol. 1, p. 3.

[8] Cao, Z., Long, M., Wang, J., and Jordan, M. I. Partial transfer learning with selective adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2018).

[9] Cawley, G. C. Leave-one-out cross-validation based model selection criteria for weighted ls-svms. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on* (2006), IEEE, pp. 1661–1668.

[10] Chen, M., Xu, Z., Weinberger, K. Q., and Sha, F. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Coference on International Conference on Machine Learning* (2012), Omnipress, pp. 1627–1634.

[11] Chen, T., Goodfellow, I., and Shlens, J. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641* (2015).

[12] Courty, N., Flamary, R., Habrard, A., and Rakotomamonjy, A. Joint distribution optimal transportation for domain adaptation. In *Advances in Neural Information Processing Systems* (2017), pp. 3733–3742.

[13] Courty, N., Flamary, R., and Tuia, D. Domain adaptation with regularized optimal transport. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2014), Springer, pp. 274–289.

[14] Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. Optimal transport for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2016).

[15] Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems* (2013), pp. 2292–2300.

[16] Dai, W., Yang, Q., Xue, G., and Yu, Y. Boosting for transfer learning. In *Proc. 24th Int. Conf. Machine Learning* (New York, NY, USA, 2007), ACM, pp. 193–200.

[17] Ding, Z., and Fu, Y. Robust transfer metric learning for image classification. *IEEE Transactions on Image Processing 26*, 2 (2017), 660–670.

[18] Donahue, J., Krähenbühl, P., and Darrell, T. Adversarial feature learning. *arXiv preprint arXiv:1605.09782* (2016).

[19] Fei-Fei, L., Fergus, R., and Perona, P. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence 28*, 4 (2006), 594–611.

[20] Fernando, B., Habrard, A., Sebban, M., and Tuytelaars, T. Unsupervised visual domain adaptation using subspace alignment. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013* (2013), pp. 2960–2967.

[21] Fink, M. Object classification from a single example utilizing class relevance metrics. In *Advances in neural information processing systems* (2005), pp. 449–456.

[22] Ganin, Y., and Lempitsky, V. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning* (2015), pp. 1180–1189.

[23] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research 17*, 1 (2016), 2096–2030.

[24] Gatys, L. A., Ecker, A. S., and Bethge, M. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576* (2015).

[25] Ge, W., and Yu, Y. Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI* (2017), vol. 6.

[26] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems* (2014), pp. 2672–2680.

[27] Gretton, A., Sejdinovic, D., Strathmann, H., Balakrishnan, S., Pontil, M., Fukumizu, K., and Sriperumbudur, B. K. Optimal kernel choice for large-scale two-sample tests. In *Advances in neural information processing systems* (2012), pp. 1205–1213.

[28] Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[29] Jiang, W., Zavesky, E., Chang, S.-F., and Loui, A. Cross-domain learning methods for high-level visual concept classification. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on* (2008), IEEE, pp. 161–164.

[30] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105.

[31] Kuzborskij, I., and Orabona, F. Stability and hypothesis transfer learning. In *International Conference on Machine Learning* (2013), pp. 942–950.

[32] Kuzborskij, I., Orabona, F., and Caputo, B. From n to n+1: Multiclass transfer incremental learning. In *Proc. 2013 IEEE Conf. Computer Vision and Pattern Recognition* (June 2013), pp. 3358–3365.

[33] Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning* (2009), ACM, pp. 609–616.

[34] Lee, H., Pham, P., Largman, Y., and Ng, A. Y. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in neural information processing systems* (2009), pp. 1096–1104.

[35] Li, S., Li, K., and Fu, Y. Self-taught low-rank coding for visual learning. *IEEE Trans. Neural Netw. Learn. Syst.* (2017).

[36] Li, X. *Regularized adaptation: Theory, algorithms and applications*, vol. 68. Citeseer, 2007.

[37] Liu, M.-Y., and Tuzel, O. Coupled generative adversarial networks. In *Advances in neural information processing systems* (2016), pp. 469–477.

[38] Long, M., Wang, J., Ding, G., Sun, J., and Yu, P. S. Transfer feature learning with joint distribution adaptation. In *2013 IEEE Int. Conf. Computer Vision* (Dec. 2013), pp. 2200–2207.

[39] Long, M., Zhu, H., Wang, J., and Jordan, M. I. Unsupervised domain adaptation with residual transfer networks. In *Adv. Neural Inf. Process Syst. (NIPS)* (2016).

[40] Long, M., Zhu, H., Wang, J., and Jordan, M. I. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning* (International Convention Centre, Sydney, Australia, 06–11 Aug 2017), D. Precup and Y. W. Teh, Eds., vol. 70 of *Proceedings of Machine Learning Research*, PMLR, pp. 2208–2217.

[41] Long, M.-S., Cao, Y., Wang, J.-M., and Jordan, M. Learning transferable features with deep adaptation networks. In *Proc. 32nd Int. Conf. Machine Learning* (2015), pp. 97–105.

[42] Lu, H., Zhang, L., Cao, Z., Wei, W., Xian, K., Shen, C., and van den Hengel, A. When unsupervised domain adaptation meets tensor representations. In *The IEEE International Conference on Computer Vision (ICCV)* (2017), vol. 2.

[43] Lu, Y., Chen, L., Saidi, A., Dellandrea, E., and Wang, Y. Discriminative transfer learning using similarities and dissimilarities. *IEEE Transactions on Neural Networks and Learning Systems 29*, 7 (July 2018), 3097–3110.

[44] Luo, L., Chen, L., Hu, S., Lu, Y., and Wang, X. Discriminative and geometry aware unsupervised domain adaptation. *CoRR abs/1712.10042* (2017).

[45] Manjunatha, V., Ramalingam, S., Marks, T. K., and Davis, L. Class subset selection for transfer learning using submodularity. *arXiv preprint arXiv:1804.00060* (2018).

[46] Pan, S. J., Kwok, J. T., and Yang, Q. Transfer learning via dimensionality reduction. In *AAAI* (2008), vol. 8, pp. 677–682.

[47] Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks 22*, 2 (2011), 199–210.

[48] Pan, S. J., and Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng. 22* (Oct. 2010), 1345–1359.

[49] Parameswaran, S., and Weinberger, K. Q. Large margin multi-task metric learning. In *Advances in neural information processing systems* (2010), pp. 1867–1875.

[50] Perrot, M., Courty, N., Flamary, R., and Habrard, A. Mapping estimation for discrete optimal transport. In *Advances in Neural Information Processing Systems* (2016), pp. 4197–4205.

[51] Perrot, M., and Habrard, A. A theoretical analysis of metric hypothesis transfer learning. In *International Conference on Machine Learning* (2015), pp. 1708–1717.

[52] Qi, G.-J., Aggarwal, C., Rui, Y., Tian, Q., Chang, S., and Huang, T. Towards cross-category knowledge propagation for learning visual concepts. In *Proc. 2011 IEEE Conf. Computer Vision and Pattern Recognition* (June 2011), pp. 897–904.

[53] Quattoni, A., Collins, M., and Darrell, T. Learning visual representations using images with captions. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on* (2007), IEEE, pp. 1–8.

[54] Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. Self-taught learning: Transfer learning from unlabeled data. In *Proc. 24th Int. Conf. Machine Learning* (2007).

[55] Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550* (2014).

[56] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV) 115*, 3 (2015), 211–252.

[57] Shao, L., Zhu, F., and Li, X. Transfer learning for visual categorization: A survey. *IEEE Trans. Neural Netw. Learn. Syst. PP*, 99 (July 2014), 1–1.

[58] Si, S., Tao, D., and Geng, B. Bregman divergence-based regularization for transfer subspace learning. *IEEE Trans. Knowl. Data Eng. 22* (July 2010), 929–942.

[59] Srinivas, S., and Fleuret, F. Knowledge transfer with Jacobian matching. In *Proceedings of the 35th International Conference on Machine Learning* (Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018), J. Dy and A. Krause, Eds., vol. 80 of *Proceedings of Machine Learning Research*, PMLR, pp. 4730–4738.

[60] Sun, B., and Saenko, K. Subspace distribution alignment for unsupervised domain adaptation. In *BMVC* (2015), pp. 24–1.

[61] Tang, Y., Wang, J., Wang, X., Gao, B., Dellandréa, E., Gaizauskas, R., and Chen, L. Visual and semantic knowledge transfer for large scale semi-supervised object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

[62] Thrun, S. Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems* (1996), The MIT Press, pp. 640–646.

[63] Tommasi, T., Orabona, F., and Caputo, B. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *Proc. 2010 IEEE Conf. Computer Vision and Pattern Recognition* (June 2010), pp. 3081–3088.

[64] Tzeng, E., Hoffman, J., Darrell, T., and Saenko, K. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 4068–4076.

[65] Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)* (2017), vol. 1, p. 4.

[66] Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474* (2014).

[67] Vapnik, V. Principles of risk minimization for learning theory. In *Advances in neural information processing systems* (1992), pp. 831–838.

[68] Villani, C. *Optimal transport: old and new*, vol. 338. Springer Science & Business Media, 2008.

[69] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning* (2008), ACM, pp. 1096–1103.

[70] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research 11*, Dec (2010), 3371–3408.

[71] Wang, H., Nie, F., and Huang, H. Robust and discriminative self-taught learning. In *International Conference on Machine Learning* (2013), pp. 298–306.

[72] Yang, J., Yan, R., and Hauptmann, A. G. Cross-domain video concept detection using adaptive svms. In *Proc. 15th Int. Conf. Multimedia* (New York, NY, USA, 2007), ACM, pp. 188–197.

[73] Yao, Y., and Doretto, G. Boosting for transfer learning with multiple sources. In *Proc. 2010 IEEE Conf. Computer Vision and Pattern Recognition* (June 2010), pp. 1855–1862.

[74] Yim, J., Joo, D., Bae, J., and Kim, J. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[75] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? In *Advances in neural information processing systems* (2014), pp. 3320–3328.

[76] Yu, X., and Aloimonos, Y. Attribute-based transfer learning for object categorization with zero/one training example. *Computer Vision–ECCV 2010* (2010), 127–140.

[77] Zhang, J., Ding, Z., Li, W., and Ogunbona, P. Importance weighted adversarial nets for partial domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2018).

[78] Zhang, J., Li, W., and Ogunbona, P. Joint geometrical and statistical alignment for visual domain adaptation. *arXiv preprint arXiv:1705.05498* (2017).

[79] Zhang, J., Li, W., and Ogunbona, P. Transfer learning for cross-dataset recognition: a survey. *arXiv preprint arXiv:1705.04396* (2017).