Transfer Learning for Image Classification

Ying Lu

N^o d'ordre NNT : 2017LYSEC45

THESE de DOCTORAT DE L'UNIVERSITE DE LYON

opérée au sein de l'Ecole Centrale de Lyon

Ecole Doctorale INFOMATHS ED N° 512

Spécialité: Informatique

Soutenue publiquement le 9/11/2017, par:

Ying LU

Transfer Learning for Image Classification

Devant le jury composé de :

M. Liming Chen	Professeur de l'Ecole Centrale de Lyon	Directeur de thèse
M. Alexandre Saidi	MCF de l'Ecole Centrale de Lyon	Co-directeur de thèse
Mme. Nicole Vincent	Professeur de l'Université Paris Descartes	Examinateur &Président
M. Nicolas Thome	Professeur du Convervatoire national des arts et métier	Rapporteur
M. Krystian Mikolajczyk	Reader de l'Imperial College London	Rapporteur
M. Amaury Habrard	Professeur de l'Université Jean Monnet	Membre invité

Acknowledgements

Firstly I would like to express my gratitude to Professor Liming CHEN, my thesis supervisor. He is visionary and energetic. I would like to thank him for choosing such an interesting thesis topic for me at the beginning of my thesis study. I also thank him for giving me enough independence in pursuing my ideas and for teaching me to iteratively formulate scientific questions and deepen my research work. Then I would like to thank Doctor Alexandre SAIDI, my thesis co-supervisor. He is precise and thoughtful. I would like to thank him for introducing me to machine learning and pattern recognition in my first years and for all his support and help during my thesis study. Especially thank him for his help on administrative procedures and French writings during all these years.

I would also like to thank my colleagues and collaborators for their great support and help. Thanks to Huibin LI for introducing me to the fantastic world of sparse representation and for showing me how to read scientific papers. Thanks to Chao ZHU for introducing me to the advanced image feature extraction techniques and for giving me advices on how to do research. Thanks to Kai WANG for teaching me the useful programming techniques with MATLAB. Thanks to Arnau RAMISA for showing me the collaborative working style with GIT. Thanks to all my colleagues for their help and for all the conversations and debates with them during lunch time, coffee time and laboratory seminars which have potentially given me inspirations for my research.

Special thanks to Professor Stan Z. Li for his precious suggestions on my first journal paper. And special thanks to Professor Xianfeng GU for his great intuitions of mathematician, which have shown me a profound picture of computer vision research from a theoretically higher position, and have inspired me to pursue more effective and more efficient applications of advanced mathematical techniques for solving computer vision problems.

Thanks to the jury members of my thesis defense for the inspirational debates we had during my thesis defense and for their valuable suggestions on my thesis manuscript. And thanks to all the anonymous reviewers I've met through every paper submission for their critiques and comments which have driven me to produce better research work.

I would also like to thank China Scholarship Council (CSC) for the financial support during my first years as a PhD student and the French Research Agency (l'ANR) and the Partner University Fund (PUF) for their financial supports during the last years of my PhD studies through research projects VideoSense, Visen and 4D Vision.

I would like to thank all my friends in Lyon, who have made Lyon as a second home to me. And I would like to thank my husband, Chen WANG, for always supporting me during my hard times. And finally, I would like to thank my parents for their influences since my childhood which have driven me to pursue the Truth in science.

Contents

List of Tables

List of Figures

Abstract

When learning a classification model for a new target domain with only a small amount of training samples, brute force application of machine learning algorithms generally leads to over-fitted classifiers with poor generalization skills. On the other hand, collecting a sufficient number of manually labeled training samples may prove very expensive. Transfer Learning methods aim to solve this kind of problems by transferring knowledge from related source domain which has much more data to help classification in the target domain. Depending on different assumptions about target domain and source domain, transfer learning can be further categorized into three categories: Inductive Transfer Learning, Transductive Transfer Learning (Domain Adaptation) and Unsupervised Transfer Learning. We focus on the first one which assumes that the target task and source task are different but related. More specifically, we assume that both target task and source task are classification tasks, while the target categories and source categories are different but related. We propose two different methods to approach this ITL problem.

In the first work we propose a new discriminative transfer learning method, namely DTL, combining a series of hypotheses made by both the model learned with target training samples, and the additional models learned with source category samples. Specifically, we use the sparse reconstruction residual as a basic discriminant, and enhance its discriminative power by comparing two residuals from a positive and a negative dictionary. On this basis, we make use of similarities and dissimilarities by choosing both positively correlated and negatively correlated source categories to form additional dictionaries. A new Wilcoxon-Mann-Whitney statistic based cost function is proposed to choose the additional dictionaries with unbalanced training data. Also, two parallel boosting processes are applied to both the positive and negative data distributions to further improve classifier performance. On two different image classification databases, the proposed DTL consistently out-

performs other state-of-the-art transfer learning methods, while at the same time maintaining very efficient runtime.

In the second work we combine the power of Optimal Transport and Deep Neural Networks to tackle the ITL problem. Specifically, we propose a novel method to jointly fine-tune a Deep Neural Network with source data and target data. By adding an Optimal Transport loss (OT loss) between source and target classifier predictions as a constraint on the source classifier, the proposed Joint Transfer Learning Network (JTLN) can effectively learn useful knowledge for target classification from source data. Furthermore, by using different kind of metric as cost matrix for the OT loss, JTLN can incorporate different prior knowledge about the relatedness between target categories and source categories. We carried out experiments with JTLN based on Alexnet on image classification datasets and the results verify the effectiveness of the proposed JTLN in comparison with standard consecutive fine-tuning. To the best of our knowledge, the proposed JTLN is the first work to tackle ITL with Deep Neural Networks while incorporating prior knowledge on relatedness between target and source categories. This Joint Transfer Learning with OT loss is general and can also be applied to other kind of Neural Networks.

Keywords: Inductive Transfer Learning, Sparse Representation, Optimal Transport, Computer Vision.

Résumé

Lors de l'apprentissage d'un modèle de classification pour un nouveau domaine cible avec seulement une petite quantité d'échantillons de formation, l'application des algorithmes d'apprentissage automatiques conduit généralement à des classifieurs surdimensionnés avec de mauvaises compétences de généralisation. D'autre part, recueillir un nombre suffisant d'échantillons de formation étiquetés manuellement peut s'avérer très coûteux. Les méthodes de transfert d'apprentissage visent à résoudre ce type de problèmes en transférant des connaissances provenant d'un domain source associé qui contient beaucoup plus de données pour faciliter la classification dans le domaine cible. Selon les différentes hypothèses sur le domaine cible et le domaine source, l'apprentissage par transfert peut être classé en trois catégories: appentissage par transfert inductif, apprentissage par transfert transducteur (adaptation du domaine) et apprentissage par transfert non surveillé. Nous nous concentrons sur le premier qui suppose que la tâche cible et la tâche source sont différentes mais liées. Plus pécisément, nous supposons que la tâche cible et la tâche source sont des tâches de classification, tandis que les catégories cible et les catégories source sont différentes mais liées. Nous proposont deux méthodes différentes pour aborder ce problème.

Dans le premier travail, nous proposons une nouvelle méthode d'apprentissage par transfert discriminatif, à savoir DTL(Discriminative Transfer Learning), combinant une série d'hypothèses faites à la fois par le modèle appris avec les échantillons de cible et les modèles supplémentaires appris avec des échantillons des catégories sources. Plus précisément, nous utilisons le résidu de reconstruction creuse comme discriminant de base et améliore son pouvoir discriminatif en comparant deux résidus d'un dictionnaire positif et d'un dictionnaire négatif. Sur cette base, nous utilisons des similitudes et des dissemblances en choisissant des catégories sources positivement corrélées et négativement corrélées pour former des dictio-

nnaires supplémentaires. Une nouvelle fonction de coût basée sur la statistique de Wilcoxon-Mann-Whitney est proposée pour choisir les dictionnaires supplémentaires avec des données non équilibrées. En outre, deux processus de Boosting parallèles sont appliqués à la fois aux distributions de données positives et négatives pour améliorer encore les performances du classificateur. Sur deux bases de données de classification d'images différentes, la DTL proposée surpasse de manière constante les autres méthodes de l'état de l'art du transfert de connaissances, tout en maintenant un temps d'exécution très efficace.

Dans le deuxième travail, nous combinons le pouvoir du transport optimal (OT) et des réseaux de neurones profond (DNN) pour résoudre le problème ITL. Plus précisément, nous proposons une nouvelle méthode pour affiner conjointement un réseau de neurones avec des données source et des données cibles. En ajoutant une fonction de perte du transfert optimal (OT loss) entre les prédictions du classificateur source et cible comme une contrainte sur le classificateur source, le réseau JTLN (Joint Transfer Learning Network) proposé peut effectivement apprendre des connaissances utiles pour la classification cible à partir des données source. En outre, en utilisant différents métriques comme matrice de coût pour la fonction de perte du transfert optimal, JTLN peut intégrer différentes connaissances antérieures sur la relation entre les catégories cibles et les catégories sources. Nous avons effectué des expérimentations avec JTLN basées sur Alexnet sur les jeux de données de classification d'image et les résultats vérifient l'efficacité du JTLN proposé. A notre connaissances, ce JTLN proposé est le premier travail à aborder ITL avec des réseaux de neurones profond (DNN) tout en intégrant des connaissances antérieures sur la relation entre les catégories cible et source.

Mots clés: Inductive Transfer Learning, Sparse Representation, Optimal Transport, Computer Vision.

Chapter 1

Introduction

Making machines that can learn and solve problems as humans is one of the most exciting and even controversial dreams of mankind. The corresponding research topic is called Artificial Intelligence (AI) and is defined as the study of "Intelligent agents": any device that perceives its environment and takes actions that maximize its chance of success at some goal [Russell *et al.* 1995]. Among many of the sub-topics of AI, one important research direction, which is commonly known as Machine Learning (ML), is to study the construction of algorithms that can learn from and make predictions on data. Arthur Samuel firstly defined Machine Learning as "the field of study that gives computers the ability to learn without being explicitly programmed" [Samuel 1959]. The earliest theoretical foundations of Machine Learning are built by Valiant, who introduced the framework of Probably Approximately Correct (PAC) learning [Valiant 1984], and Vapnik, who casts the problem of 'learning' as an optimization problem [Vapnik & Vapnik 1998]. Nowadays machine learning is a combination of several disciplines such as statistics, information theory, measure theory and functional analysis.

Depending on the nature of the learning "signal" or "feedback" available to a learning system, Machine Learning tasks are typically classified into three broad categories: Supervised Learning (where the computer is presented with example inputs and their desired outputs, the goal is to learn a general rule that maps inputs to outputs), Unsupervised Learning (where only example inputs are given without corresponding outputs, the goal is to find structure in the given inputs) and Reinforcement Learning (where the computer program interacts with a dynamic environment in which it must perform a certain goal, the program is provided feedback in terms of rewards and punishments as it navigates its problem space).

Depending on the desired output of a learning system, Machine Learning tasks can be categorized differently: in *classification*, inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more of these classes; in *regression*, the outputs are continuous rather than discrete; in *clustering*, a set of inputs is to be divided into groups; in *density estimation*, the program finds the distribution of inputs in some space; and *dimensionality reduction* simplifies inputs by mapping them into a lower-dimensional space, etc.

Nowadays, technology is in constant evolution and the amount of data is everyday dramatically increasing. In particular, we are witnessing a spectacular growth in image and video data due to the rapid spread of electronic devices capable of recording and sharing pictures and videos (*e.g.* smart-phones, tablets, digital cameras, surveillance video recorders, *etc.*) all around the world. Consequently, billions of raw images and videos are diffused on the Internet. For instance, 612 million of pictures are uploaded on Flickr during the year 2016[1], and approximately 400 hours of new videos are uploaded on Youtube every minute according to a recent report in 2017[2]. However, most of these images and video content are difficult to exploit because they have not been properly labeled or edited. Therefore, automatic classification of images becomes a quite urgent need. **In this thesis, we mainly focus on the *supervised classification* of images**.

Ideally, when enough labeled training samples are given, the supervised classification problem could be formalized as an Empirical Risk Minimization (ERM) problem where we search in the hypothesis space for a hypothesis that can minimize the empirical risk on training samples. Thanks to Hoeffding's inequality, when the hypothesis space is properly chosen and the training set is large enough, the learned hypothesis could have a bounded generalization error (*i.e., the difference between the empirical risk and the expected risk*) which guarantees its good performance on same distributed test samples.

However, in reality this is not always the case. A common problem is that collecting a sufficient number of manually labeled training samples is very expen-

[1] https://www.flickr.com/photos/franckmichel/6855169886
[2] https://expandedramblings.com/index.php/youtube-statistics/

sive. Especially with the rapid increase of Web data, most of the Web collected images are unlabeled or with very noisy labels. When the number of images is enormous, manually labelling them would be expensive and time-consuming. Furthermore, when dealing with visual data, a frequently encountered problem is that even for a same semantic concept, the images obtained can be surprisingly different by using different sensors, under different lighting conditions, or having different backgrounds, *etc.*. These kind of problems give birth to a pressing need for algorithms that can learn efficiently from a small amount of labeled training data by leveraging knowledge from related unlabeled or noisy labeled data or differently distributed data. The research direction that deals with these kind of problems is called 'Transfer Learning'.

The study of Transfer Learning is motivated by the fact that human, even a child, can intelligently apply knowledge learned previously to solve new problems efficiently. An example in [Quattoni *et al.* 2009] gives an evidence on this point: when a child learns to recognize a new letter of the alphabet he will use examples provided by people with different hand-writing styles using pens of different colors and thicknesses. Without any prior knowledge a child would need to consider a large set of features as potentially relevant for learning the new concept, so we would expect the child to need a large number of examples. But if the child has previously learnt to recognize other letters, he can probably discern the relevant attributes (*e.g.* number of lines, line curvatures) from irrelevant ones (*e.g.* the color of the lines) and learn the new concept with a few examples. The fundamental motivation for Transfer Learning in the field of Machine Learning was discussed in a NIPS-95 workshop on "Learning to Learn", which focused on the need for lifelong machine learning methods that retain and reuse knowledge which are learned previously. Research on Transfer Learning has attracted more and more attention since 1995. Compared to traditional machine learning techniques which try to learn each task from scratch, transfer learning techniques try to transfer the knowledge from some previous tasks to a target task when the latter has fewer high-quality training data.

The goal of this thesis is to develop efficient transfer learning algorithms for images classification. In the following we will firstly give a formal

description of this problem, and then introduce our contributions.

1.1 Problem definition

1.1.1 Image classification

In an image classification task our goal is to learn a mapping from images to class labels. The input images could be represented by pre-extracted feature vectors (as in chapter 3) or image pixels directly (as in chapter 4), we can therefore assume a vector $\mathbf{x} \in \mathbb{R}^d$ as notation for an image, with d the number of feature dimensions or number of pixels. For the output class labels, we can either consider binary classification (as in chapter 3) or multi-class classification (as in chapter 4). In both cases we can either represent the label of an image with a scalar y ($y \in \{+1, -1\}$ for binary classification or y a discrete value as class index for multi-class classification) or a vector $\mathbf{y} \in \{0, 1\}^n$, with n the number of classes.

To build an efficient image classification model, there are two key problems that need to be solved. The first one is to find a discriminative feature space in which the class distributions can be easily distinguished from each other, this can either be done by feature selection (*i.e.*, selecting most discriminative features), or by mapping the samples into a new feature space (*e.g.*, traditional feature extraction techniques such as SIFT (Scale-Invariant Feature Transform) or HOG (Histogram of Oriented Gradient), or the recent representation learning techniques such as Dictionary Learning or Convolutional Neural Networks (CNNs)). The second one is to build a proper classifier which maps samples from the feature space to the class label space. The classifier can either be a generative model which learns the joint distribution $p(\mathbf{x}, y)$ (*e.g.*,mixture models) or a discriminative model which learns the conditional distribution $p(y|\mathbf{x})$ (*e.g.*, Support Vector Machines (SVMs)).

In the computer vision community, the first problem is usually the most concerned one, especially with the rapid evolution of Deep Neural Networks, a good feature representation learned with Convolutional Neural Networks can give excellent classification performance even with a simple classifier (*e.g.*, softmax classifier or K-Nearest Neighbors classifier). However, the second problem is also important,

especially for transfer learning problems in which we only have a few target training samples. Depending on the techniques used, these two problems can sometimes be treated in a unified model (*e.g.*, in CNNs the feature extraction layers and the softmax classifier are integrated in a unified Deep Neural Network).

1.1.2 Transfer Learning

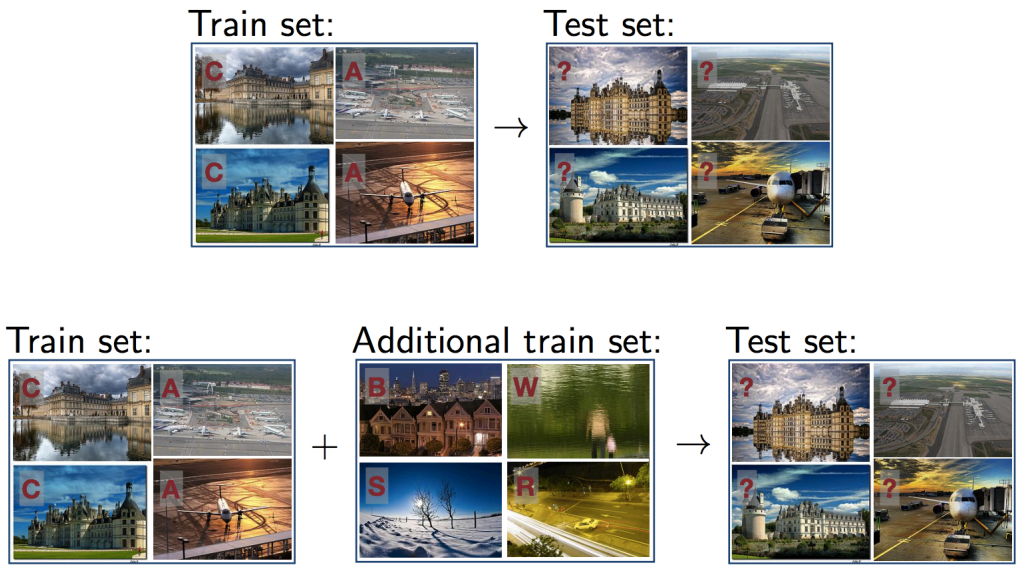

Figure 1.1: Comparison of traditional machine learning setting with transfer learning setting: in traditional machine learning setting, training set and test set should be formed with images from same categories and follow the same probability distribution; while in transfer learning setting, an additional training set is also given, which is allowed to have images from different data distribution, or even from different kind of categories. (The labels 'C' stands for 'Castle', 'A' stands for 'Airport', 'B' stands for 'Building', 'W' stands for 'Water', 'S' stands for 'Snow', 'R' stands for 'Road' and '?' stands for unknown.)

In this section, we follow the notations introduced in [Pan & Yang 2010a] to describe the problem statement of transfer learning. A *domain* $\mathcal{D}$ consists of two components: a feature space $\mathcal{X}$ and a marginal probability distribution $P(\mathbf{x})$, where $\mathbf{x} \in \mathcal{X}$. In general, if two domains are different, then they may have different feature spaces or different marginal probability distributions. Given a specific domain, $\mathcal{D} = \{\mathcal{X}, P(\mathbf{x})\}$, a *task* $\mathcal{T}$ consists of two components: a label space $\mathcal{Y}$ and a predictive function $f(\cdot)$, denoted by $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$. The function $f(\cdot)$ is a predictive

function (*i.e.*, a mapping from the feature space to the label space) that can be used to make predictions on unseen instances. From a probabilistic viewpoint, $f(\mathbf{x})$ can also be written as the conditional distribution $P(y|\mathbf{x})$.

Based on the notations defined above, the definition of transfer learning can be defined as follows [Pan & Yang 2010a],

Definition 1. *Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, transfer learning aims to help improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$.*

In the above definition, a domain is a pair $\mathcal{D} = \{\mathcal{X}, P(\mathbf{x})\}$, thus the condition $\mathcal{D}_S \neq \mathcal{D}_T$ implies that either $\mathcal{X}_S \neq \mathcal{X}_T$ or $P(\mathbf{x}_S) \neq P(\mathbf{x}_T)$. Similarly, a task is defined as a pair $\mathcal{T} = \{\mathcal{Y}, P(y|\mathbf{x})\}$, thus the condition $\mathcal{T}_S \neq \mathcal{T}_T$ implies that either $\mathcal{Y}_S \neq \mathcal{Y}_T$ or $P(y_S|\mathbf{x}_S) \neq P(y_T|\mathbf{x}_T)$. When the target and the source domains are the same, *i.e.* $\mathcal{D}_S = \mathcal{D}_T$, and their learning tasks are the same, *i.e.* $\mathcal{T}_S = \mathcal{T}_T$, the learning problem becomes a traditional machine learning problem. An illustration which compares the traditional machine learning setting and the transfer learning setting is given in Figure 1.1.

Based on different conditions for differences between source domain and target domain and differences between source task and target task, transfer learning scenarios can be categorized differently. For example, based on whether the feature spaces or label spaces are identical or not, transfer learning is categorized into two settings [Pan 2014]: 1) *homogeneous transfer learning* (where the intersection between source and target feature spaces is not empty ($\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$) and source and target label spaces are the same ($\mathcal{Y}_S = \mathcal{Y}_T$), while source and target marginal distributions and conditional distributions are different ($P(\mathbf{x}_S) \neq P(\mathbf{x}_T)$ or $P(y_S|\mathbf{x}_S) \neq P(y_T|\mathbf{x}_T)$)), and 2) *heterogeneous transfer learning* (where the two feature spaces have empty intersection or the two label spaces are different ($\mathcal{X}_S \cap \mathcal{X}_T = \emptyset$ or $\mathcal{Y}_S \neq \mathcal{Y}_T$)).

Another way to categorize transfer learning is based on whether the two domains or two tasks are identical or not [Pan & Yang 2010a], we have: 1) *inductive transfer*

learning (where source and target domains are the same, while source and target tasks are different but related), 2) *unsupervised transfer learning* (where source and target domains are different but related, and source and target tasks are also different but related), and 3) *transductive transfer learning* (where source and target domains are different but related, while source and target tasks are the same).

In this thesis, we mainly focus on the transfer learning scenario where the intersection between source and target feature spaces is not empty ($\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$) while the source and target label spaces are different ($\mathcal{Y}_S \neq \mathcal{Y}_T$). According to the two categorization methods shown above, this scenario can be categorized as *heterogeneous transfer learning* or *inductive transfer learning*.

Specifically, in chapter 3 we consider binary classification problem. We assume having a target domain $\mathcal{D}_T$ with a binary classification task $\mathcal{T}_T$, which can be accessed through a small set of target training data. We also assume having a source domain $\mathcal{D}_S$ with multiple source binary classification tasks $\mathcal{T}_{S,1}, \ldots, \mathcal{T}_{S,L}$, which can be accessed through a large set of source training data. As mentioned in the previous paragraph, here $\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$ and $\mathcal{Y}_{S,i} \neq \mathcal{Y}_T$, $\forall i \in [1, L]$. The aim is to learn a discriminative predictive function for the target task using both the target training data and the source training data.

In chapter 4 we consider multi-class classification problem. We assume having a target domain $\mathcal{D}_T$ with a multi-class classification task $\mathcal{T}_T$, along with a source domain $\mathcal{D}_S$ with a multi-class classification task $\mathcal{T}_S$. Similarly we assume $\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$ and $\mathcal{Y}_S \neq \mathcal{Y}_T$. The aim is also to learn a discriminative predictive function for the target task using both the target training data and the source training data.

1.2 Thesis Contributions

As mentioned above, in this thesis we study the *heterogeneous inductive transfer learning* scenario for image classification. Specifically, we propose two different approaches to tackle this problem:

- In chapter 3 we propose a novel discriminative knowledge transfer method, which leverages relatedness of various source categories with the target cate-

gory to enhance learning of the target classifier. The proposed Discriminative Transfer Learning (DTL) explicitly makes use of both positively and negatively correlated source categories to help classification of the target category. Specifically, DTL chooses from source categories the most positively and negatively correlated categories to act as positive and negative dictionaries for discriminative classification of target samples using reconstruction residuals on these dictionaries. We further enhance the performance of DTL by concurrently running two AdaBoost processes on the positive and negative distributions of the target training set. A novel Wilcoxon-Mann-Whitney(WMW)-based cost function is also introduced in DTL to deal with the unbalanced nature of data distribution.

The main contributions of this work are fourfold:

1. We highlight the importance of learning both similarity and dissimilarity in a transfer learning algorithm through joint use of positively correlated and negatively correlated source categories, and introduce a Bi-SRC classifier as the building block of the proposed DTL.
2. We propose a novel cost function based on the Wilcoxon-Mann-Whitney (WMW) statistic, and apply two parallel boosting processes, both on positive data and on negative data distribution, thus successfully avoiding the effect of unbalanced data distribution.
3. We conduct theoretical analyses on the proposed DTL algorithm and provide theoretical guarantees both in terms of error bound and time complexity.
4. Using different features and evaluating on two different performance metrics, we benchmark the proposed DTL on two different databases for the task of image categorization. We also consistently demonstrate the effectiveness of the proposed DTL: it displays the best performance with a large margin in comparison to several state-of-the-art TL methods, with a runtime that can prove 80 times faster in training and 66 times faster in testing than the other state-of-the-art TL methods that it has been

compared with.

- In chapter 4 we propose a novel method to jointly fine-tune a Deep Neural Network with both source data and target data. In contrast to naive joint fine-tuning, we propose to explicitly account for the relatedness between source and target tasks and explore such prior knowledge through the design of a novel loss function, namely Optimal Transport loss (OT loss), which is minimized during joint training of the underlying neural network, in order to bridge the gap between the source and target classifiers. This results in a Joint Transfer Learning Network (JTLN) which can be built upon common Deep Neural Network structure. In JTLN, the source data and target data go through same feature extraction layers simultaneously, and then separate into two different classification layers. The Optimal Transport loss is added between the two classification layers' outputs, in order to minimize the distance between two classifiers' predictions. As the Optimal Transport loss is calculated with a pre-defined cost matrix, this JTLN can therefore incorporate different prior knowledge about the relations between source and target tasks by using different kind of cost metric. We show two examples of using the distances between category distributions as cost metric.

  The contributions of this work are threefold:

  1. We propose a Joint Transfer Learning framework built upon existing Deep Neural Networks for Inductive Transfer Learning.
  2. We extend the Wasserstein loss proposed in [Frogner *et al.* 2015] to a more general Optimal Transport loss for comparing probability measures with different length, and use it as a soft penalty in our JTLN.
  3. We show two different ways of using the distance between category distributions as cost metric for OT loss. Experimental results on two ITL image classification datasets show that JTLN with these two cost metrics can achieve better performance than consecutive fine-tuning or simple joint fine-tuning without extra constraint.

1.3 Outline of the thesis

The structure of this thesis is as follows: chapter 2 reviews existing related work on general transfer learning algorithms and transfer learning algorithms for vision recognition problems; chapter 3 describes in detail our work on Discriminative Transfer Learning using both similarities and dissimilarities; chapter 4 describes in detail our work on Joint Transfer Learning Network. Finally, in chapter 5 we draw conclusions and discuss future lines of research.

Chapter 2

Literature Review

In this chapter we review related works on general transfer learning algorithms as well as previous literature on transfer learning algorithms for vision recognition problems.

Following the two main problems for image classification (which are also the key problems for vision recognition tasks) introduced in section 1.1.1, we can approximately categorize related transfer learning algorithms into two categories: 1) feature representation level knowledge transfer; and 2) classifier level knowledge transfer. We will introduce these two categories in detail in the following sections.

2.1 Feature representation level knowledge transfer

Feature representation level knowledge transfer algorithms mainly focus on learning a feature representation which is discriminative for target task by leveraging knowledge from both target training data and source data. Normally these methods assume that there exists a mapping from the original input space to an underlying shared feature representation. This latent representation captures the information necessary for training classifiers for source and target tasks. The goal of these algorithms is therefore to uncover the underlying shared representation and the parameters of the classifier for target task.

Grouped by the feature mapping techniques used by different algorithms, we will mainly introduce five lines of research works in this section:

In section 2.1.1 we introduce some early transfer learning algorithms which learns a shared representation with shallow Neural Networks or linear transformations. Most of these works are proposed for the multi-task learning scenario, which

considers source tasks and target tasks equally and aim to share knowledge across all related tasks.

In section 2.1.2 we introduce some transfer learning algorithms which learns a shared representation using conventional dimensionality reduction methods. Most of these works are proposed for the domain adaption scenario, which is also a sub-topic of transfer learning and aim at adapting distributions between source data and target data.

In section 2.1.3 we introduce some transfer learning algorithms which learn underlying feature space through the metric learning framework.

In section 2.1.4 we introduce some recent transfer learning algorithms which make use of the Deep Neural Networks as feature mapping. Similar to the methods in section 2.1.2, these algorithms are also proposed for domain adaptation scenarios. Since nowadays deep neural networks are the state-of-the-art feature learning architecture, these DNN based transfer learning algorithms are also the current state-of-the-art methods for domain adaptation.

In section 2.1.5 we introduce some knowledge transfer algorithms based on dictionary learning (also known as *sparse coding*) which are designed for the *self-taught learning* scenario. Since in *self-taught learning* we perform knowledge transfer from unlabeled source samples which are easy to collect, this is a hard but promising sub-problem of transfer learning.

2.1.1 Representation Learning with Shallow Neural Networks and linear transformations

2.1.1.1 Transfer learning with shallow Neural Networks

One of the earliest works on transfer learning was [Thrun 1996] which introduced the concept of *lifelong learning*. Thrun proposed a transfer algorithm that uses source training data to learn a function, denoted by $g : I \rightarrow I'$, which maps input samples in feature space I to a new feature space I'. The main idea is to find a new representation where every pair of positive examples for a task will lie close to each other while every pair of positive and negative examples will lie far from each

other.

Let $\mathcal{P}_k$ be the set of positive samples for the k-th task and $\mathcal{N}_k$ the set of negative samples, Thrun's transfer algorithm minimizes the following objective:

$$\min_{g \in \mathcal{G}} \sum_{k=1}^{m} \sum_{\mathbf{x}_i \in \mathcal{P}_k} \Big(\sum_{\mathbf{x}_j \in \mathcal{P}_k} \| \, g(\mathbf{x}_i) - g(\mathbf{x}_j) \, \| - \sum_{\mathbf{x}_j \in \mathcal{N}_k} \| \, g(\mathbf{x}_i) - g(\mathbf{x}_j) \, \| \Big) \tag{2.1}$$

where $\mathcal{G}$ is the set of transformations encoded by a two layer neural network. The transformation $g(\cdot)$ learned from the source data is then used to project the samples of the target task into the new feature space. Classification for the target task is performed by running a nearest neighbor classifier in the new space.

The paper presented experiments on a small object recognition task. The results showed that when labeled data for the target task is scarce, the representation obtained by running their transfer algorithms on source training data could improve the classification performance of a target task.

This work is further generalized by several authors [Ando & Zhang 2005] [Argyriou *et al.* 2007] [Amit *et al.* 2007]. The three works can all be casted under the framework of '*structural learning*' proposed in [Ando & Zhang 2005].

2.1.1.2 Structural learning methods

In a *structural learning* framework we assume the existence of task-specific parameters $\mathbf{w}_k$ for each task and shared parameters θ that parameterize a family of underlying transformations. Both the structural parameters and the task-specific parameters are learned together via joint risk minimization on some supervised training data for m related tasks.

Consider learning linear predictors of the form $h_k(\mathbf{x}) = \mathbf{w}_k^T v(\mathbf{x})$ for some $\mathbf{w} \in \mathbb{R}^z$ and some transformation $v \in \mathcal{V} : \mathbb{R}^d \to \mathbb{R}^z$. In particular, let $\mathcal{V}$ be the family of linear transformations: $v^\theta(\mathbf{x}) = \theta\mathbf{x}$ where θ is a z by d matrix that maps a d dimensional input vector to a z dimensional space.

Define the task-specific parameters matrix: $W = [\mathbf{w}_1, \ldots, \mathbf{w}_m]$ where $\mathbf{w}_k \in \mathbb{R}^z$ are the parameters for the k-th task and $w_{j,k}$ is the parameter value for the j-th

hidden feature and the k-th task. A *structural learning* algorithm finds the optimal task-specific parameters W^* and structural parameters θ^* by minimizing a jointly regularized empirical risk:

$$\arg\min_{W,\theta} \sum_{k=1}^{m} \frac{1}{n_k} \sum_{i=1}^{n_k} \text{Loss}(\mathbf{w}_k^\top \theta \mathbf{x}_i^k, y_i^k) + \gamma \Phi(W) + \lambda \Psi(\theta) \tag{2.2}$$

The first term in equation (2.2) measures the mean error of the m classifiers by means of some loss function $\text{Loss}(\cdot)$. The second term is a regularization penalty on the task-specific parameters W and the last term is a regularization penalty on the structural parameters θ. Different choices of regularization functions $\Phi(W)$ and $\Psi(\theta)$ result in different *structural learning* algorithms.

[Ando & Zhang 2005] combine a l_2 regularization penalty on the task-specific parameters with an orthonormal constraint on the structural parameters, resulting in the following objective:

$$\arg\min_{W,\theta} \sum_{k=1}^{m} \frac{1}{n_k} \sum_{i=1}^{n_k} \text{Loss}(\mathbf{w}_k^\top \theta \mathbf{x}_i^k, y_i^k) + \gamma \sum_{k=1}^{m} \parallel \mathbf{w}_k \parallel_2^2, \quad s.t.\ \theta\theta^T = I \tag{2.3}$$

where θ is a z by d matrix, z is assumed to be smaller than d and its optimal value is found using a validation set. Therefore knowledge transfer is realized by mapping the high dimensional feature vector $\mathbf{x}$ to a new feature vector $\theta \cdot \mathbf{x}$ in a shared low dimensional feature space. The authors propose to solve (2.3) using an alternating minimization procedure. This algorithm is applied in the context of *asymmetric transfer* where *auxiliary* (*i.e.* source) training sets are utilized to learn the structural parameter θ. The learned structural parameter is then used to project the samples of the target task and train a classifier on the new feature space. The paper presented experiments on text categorization where the source training sets were automatically derived from unlabeled data (this algorithm can therefore be regarded as a semi-supervised training algorithm). their results showed that the proposed algorithm gave significant improvements over a baseline method that trained on the labeled data ignoring the source training sets.

This work is further applied to image classification in [Quattoni *et al.* 2007] in which they consider having a large set of images with associated captions, while among them only a few images are annotated with story news labels. They take the prediction of content words from the captions to be the source tasks and the prediction of a story label to be the target tasks. The goal is to leverage the source tasks to derive a lower dimensional representation that captures the relevant information necessary to discriminate between different stories. They perform experiments with this method both on synthetic data and real news image data. Results show that when source data labels are suitably related to a target task, the structural learning method can discover feature groupings that speed up learning of the target task.

[Argyriou *et al.* 2007] proposed an alternative model to learn shared representations. In their approach the structural parameter θ is assumed to be a d by d matrix, *i.e.* the linear transformation does not map the inputs $\mathbf{x}$ to a lower dimensional space. Instead, sharing of hidden features across tasks is realized by a regularization penalty imposed on the task-specific parameters W, which requires only a few hidden features to be used by any task (*i.e.* requires the matrix W to be row-sparse). This regularization is achieved by using the following matrix norm: $l_{1,2}(W) = \sum_{j=1}^{z} \parallel \mathbf{w}^j \parallel_2$, which is known to promote row sparsity in W. Therefore the objective can be written as:

$$\arg\min_{W,\theta} \sum_{k=1}^{m} \frac{1}{n_k} \sum_{i=1}^{n_k} \text{Loss}(\mathbf{w}_k^\top \theta \mathbf{x}_i^k, y_i^k) + \gamma l_{1,2}(W) \tag{2.4}$$

The authors showed that this problem is equivalent to a convex problem for which they developed an alternating minimization algorithm. The paper presented experiments on a product rating problem where the goal is to predict ratings given by different subjects. In the context of multi-task learning predicting the ratings for a single subject can be regarded as a task. The transfer learning assumption is that predictions made by different subjects are related. The results showed that their algorithm gave better performance than a baseline model where each task was trained independently with an l_1 penalty.

[Amit *et al.* 2007] proposed a regularization scheme for transfer learning based on a trace norm regularization penalty. Consider the following m by d parameter matrix $W = [\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^m]$, where each row corresponds to the parameters of one task. The transfer algorithm minimizes the following jointly regularized objective:

$$\arg\min_{W} \sum_{k=1}^{m} \frac{1}{n_k} \sum_{i=1}^{n_k} \text{Loss}(\mathbf{w}_k^\top \mathbf{x}_i^k, y_i^k) + \gamma \Omega(W) \tag{2.5}$$

where $\Omega(W) = \sum_i |\gamma_i|$ and γ_i is the i-th eigenvalue of W. This norm is used because it is known to induce low rank on solution matrices W [Srebro & Jaakkola 2003]. Recall that the rank of a d by m matrix W is the minimum z such that W can be factored as $W = \theta^\top W'$, for a z by m matrix W' and a z by d matrix θ.

Notice that θ is no longer in equation (2.5), this is because in this formulation we do not search explicitly for a transformation θ. Instead, we utilize the regularization penalty $\Omega(W)$ to encourage solutions where the task-specific parameters W can be expressed as the combination of a few basis shared across tasks. This optimization problem can be expressed as a semi-definite program and can be solved with an interior-point method. However, the authors argue that interior point methods scale poorly with the size of the training set and proposed a gradient based method to solve this problem by minimizing a smoothed approximation of (2.5). The authors conducted experiments on a multi-class classification task where the goal is to distinguish between 72 classes of mammals. The performance of their transfer learning algorithm is compared to that of a baseline multi-class SVM classifier. Their results show that the trace-norm penalty can improve multi-class accuracy when only a few samples are available for training.

2.1.2 Representation Learning with dimensionality reduction methods

In the previous section 2.1.1, we have introduced some early works on representation learning using shallow neural networks or linear transformations. A common assumption of these works is that they assume a representation mapping learned

with the source data can be directly applied to target data. This is a strong assumption which requests the source data to be very close to the target data, if not the knowledge transfer with the learned representation mapping will fail.

In this section and the next section 2.1.4 we introduce some transfer learning algorithms with distribution adaptation which relax this assumption and learn the shared representation by explicitly minimize predefined distance measures to reduce the differences between source and target in the marginal distribution [Si *et al.* 2010] [Pan *et al.* 2008] [Pan *et al.* 2011], or in the conditional distribution [Satpal & Sarawagi 2007], or both [Long *et al.* 2013b] [Long *et al.* 2015] [Long *et al.* 2016]. These works are mostly proposed for the *Domain Adaptation* scenario, which is a subproblem of transfer learning.

Recall the notations defined in section 1.1.2, assume having a target domain $\mathcal{D}_T = \{\mathcal{X}_T, P(\mathbf{x}_T)\}$ with a target task $\mathcal{T}_T = \{\mathcal{Y}_T, P(y_T|\mathbf{x}_T)\}$ (here conditional probability $P(y_T|\mathbf{x}_T)$ is equivalent to the prediction function $f_T(\cdot)$), and a source domain $\mathcal{D}_S = \{\mathcal{X}_S, P(\mathbf{x}_S)\}$ with a source task $\mathcal{T}_S = \{\mathcal{Y}_S, P(y_S|\mathbf{x}_S)\}$. Given the assumptions: $\mathcal{X}_S = \mathcal{X}_T$, $\mathcal{Y}_S = \mathcal{Y}_T$, $P(\mathbf{x}_S) \neq P(\mathbf{x}_T)$ and $P(y_S|\mathbf{x}_S) \neq P(y_T|\mathbf{x}_T)$, the transfer learning algorithms by distribution adaptation aim to learn a new feature representation in which the distribution differences between $P(\mathbf{x}_S)$ and $P(\mathbf{x}_T)$, or between $P(y_S|\mathbf{x}_S)$ and $P(y_T|\mathbf{x}_T)$, or both of them are explicitly reduced.

2.1.2.1 Adaptation with Bregman divergence based distance measure

In [Si *et al.* 2010] the authors proposed a Bregman Divergence based regularization schema for transfer subspace (representation) learning, which combines Bregman divergence with conventional dimensionality reduction algorithms. Similar to the *structural learning* schema introduced in section 2.1.1, this regularized *subspace learning* also learns a feature mapping and a classifier at the same time. The difference between this work and the *structural learning* framework is that 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 $v^{\theta}(\mathbf{x}) = \theta\mathbf{x}$ 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 $F(\theta)$:

$$\theta^* = \arg\min_{\theta \in \mathbb{R}^{z \times d}} F(\theta) \tag{2.6}$$

The objective function $F(\theta)$ is designed for specific applications, here it minimizes the data classification loss in the selected subspace according to different assumptions or intuitions. For example, Fisher's linear discriminant analysis (FLDA) selects a subspace, where the trace ratio of the within-class scatter matrix and the between-class scatter matrix is minimized.

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 measure the distribution difference of samples drawn from different domains in the projected subspace θ. By integrating this regularization into (2.6), we obtain a new framework for transfer subspace learning (TSL):

$$\theta^* = \arg\min_{\theta \in \mathbb{R}^{z \times d}} F(\theta) + \lambda D_{\theta}(P_S \parallel P_T) \tag{2.7}$$

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 (2.7).

Let $v : I \rightarrow I'$ be a C^1 convex function defined on a closed convex set $I \subset \mathbb{R}^+$. We denote the first order derivative of v by v', denote its inverse function by $\xi = (v')^{-1}$. The probability density for the source and target samples in the projected subspace I' is $P_S(v(\mathbf{x}))$ and $P_T(v(\mathbf{x}))$ respectively. The regularization term is defined as follows:

$$D_{\theta}(P_S \parallel P_T) = \int d(\xi(P_S(v(\mathbf{x}))), \xi(P_T(v(\mathbf{x})))) d\mu \tag{2.8}$$

where $d(\xi(P_S(v(\mathbf{x}))), \xi(P_T(v(\mathbf{x})))$ is the difference at $\xi(P_T(v(\mathbf{x})))$ between the function v and the tangent line to v at point $(\xi(P_S(v(\mathbf{x}))), v(\xi(P_S(v(\mathbf{x})))))$, $d\mu$ (*i.e.*,

$du(v(\mathbf{x})))$ is the Lebesgue measure. The right hand side of (2.8) is also called the U-divergence on the subspace $\mathbb{R}^d$.

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 present experimental evidence on both face image data sets and text data sets, suggesting that the proposed framework is effective to deal with cross-domain learning problems.

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

Similar to the previous approach, in [Pan *et al.* 2008] 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.

Maximum Mean Discrepancy (MMD) is a two samples test criterion for comparing distributions based on Reproducing Kernel Hilbert Space (RKHS). Let $X = \{x_1, \ldots, x_{n_1}\}$ and $Y = \{y_1, \ldots, y_{n_2}\}$ be two random variable sets from distributions $\mathcal{P}$ and $\mathcal{Q}$, respectively, and $\mathcal{H}$ be a universal RKHS with the reproducing kernel mapping ϕ: $f(x) = \langle \phi(x), f \rangle$, $\phi : \mathcal{X} \to \mathcal{H}$. The empirical estimate of distance between $\mathcal{P}$ and $\mathcal{Q}$ defined by MMD is as follows:

$$Dist(X, Y) = \| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi(x_i) - \frac{1}{n_2} \sum_{i=1}^{n_2} \phi(y_i) \|_{\mathcal{H}} \tag{2.9}$$

As can be seen, the MMD between two sample sets is equivalent to the distance between the means of the two sample sets mapped into a RKHS. Based on this, the authors proposed a new dimensionality reduction method, denoted as MMDE (Maximum Mean Discrepancy Embedding), to learn a low-dimensional latent space

F common to source and target domains. 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).

Denote the source domain data as $\mathcal{D}_{src} = \{(x_1^{src}, y_1^{src}), \ldots, (x_{n_1}^{src}, y_{n_1}^{src})\}$, where $x_i^{src} \in \mathbb{R}^m$ is the input sample feature and y_i^{src} the corresponding label. Similarly, denote the target domain data as $\mathcal{D}_{tar} = \{(x_1^{tar}, y_1^{tar}), \ldots, (x_{n_2}^{tar}, y_{n_2}^{tar})\}$ with $x_i^{tar} \in \mathbb{R}^m$. Let the feature projection map be ψ. Then learning a common low-dimensional latent space in which the distributions of the source and target data (*i.e.*, X'_{src} and X'_{tar}) can be close to each other is equivalent to minimizing the MMD between X'_{src} and X'_{tar}:

$$\begin{aligned} Dist(X'_{src}, X'_{tar}) &= Dist(\psi(X_{src}), \psi(X_{tar})) \\ &= \| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi \circ \psi(x_i^{src}) - \frac{1}{n_2} \sum_{i=1}^{n_2} \phi \circ \psi(x_i^{tar}) \| \end{aligned} \tag{2.10}$$

Denote the corresponding kernel of $\phi \circ \psi$ by k, then equation (2.10) can be written in terms of the kernel matrices defined by k as:

$$Dist(X'_{src}, X'_{tar}) = \text{trace}(KL) \tag{2.11}$$

where

$$K = \begin{bmatrix} K_{src,src} & K_{src,tar} \\ K_{tar,src} & K_{tar,tar} \end{bmatrix} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$$

is a composite kernel matrix, and $L = [L_{ij}] \succeq 0$ with

$$L_{ij} = \begin{cases} \frac{1}{n_1^2} & x_i, x_j \in X_{src}, \\ \frac{1}{n_2^2} & x_i, x_j \in X_{tar}, \\ -\frac{1}{n_1 n_2} & \text{otherwise.} \end{cases}$$

The authors proved that this kernel matrix K correspond to an universal kernel,

so we can learn this kernel matrix instead of learning the universal kernel k. Thus, the embedding problem can be formulated as the following optimization problem:

$$\begin{aligned} \min_{K=\widetilde{K}+\varepsilon I} \quad & \text{trace}(KL) - \lambda \text{trace}(K) \\ \text{s.t.} \quad & K_{ii} + K_{jj} - 2K_{ij} = d_{ij}^2, \ \forall (i,j) \in \mathcal{N}, \\ & K\mathbf{1} = \mathbf{0}, \ \widetilde{K} \succeq 0. \end{aligned} \tag{2.12}$$

where ε is a small positive constant and $\mathbf{1}$ and $\mathbf{0}$ are the vectors of ones and zeros, respectively. The second term is added to unfold the high dimensional data by maximizing the trace of K. This optimization problem can be further rewritten as a semidefinite program (SDP) which learns $\widetilde{K}$ and can be solved by standard SDP solvers. After obtaining $\widetilde{K}$, the authors then apply PCA and select the leading eigen vectors to construct low-dimensional representations X'_{src} and X'_{tar}. A classifier is learned with X'_{src} and Y_{src} and is applied directly for classification in target domain.

The authors perform experiments on indoor WiFi localization dataset and text classification dataset, the results showed that the proposed MMDE can effectively improve the performance compared to traditional machine learning algorithms.

2.1.2.3 Transfer Component Analysis

The previous MMDE suffers from two major limitations: (1) it is transductive, and does not generalize to out-of-sample patterns; (2) it learns the latent space by solving a semi-definite program (SDP), which is a very expensive optimization problem. Furthermore, in order to construct low-dimensional representations, in MMDE the obtained K has to be further post-processed by PCA, this step may discard potentially useful information in K. To get ride of these limitations, the authors further proposed in [Pan *et al.* 2011] a new approach, named *transfer component analysis* (TCA), which tries to learn a set of common *transfer components* underlying both domains such that the difference in data distributions in the new subspace of two domains can be reduced, and data properties can be preserved.

Unlike MMDE, this proposed TCA avoids the use of SDP and can be gener-

alized to out-of-sample patterns. Besides, instead of using a two-step approach, they propose a unified kernel learning method which utilizes an explicit low-rank representation.

First note that the kernel matrix K defined in (2.11) can be decomposed as $K = (KK^{-1/2})(K^{-1/2}K)$, which is often known as the empirical kernel map. Consider the use of a matrix $\widetilde{W} \in \mathbb{R}^{(n_1+n_2)\times m}$ that transforms the empirical kernel map features to an m-dimensional space (where $m \ll n_1 + n_2$). The resultant kernel matrix is then

$$\widetilde{K} = (KK^{-1/2}\widetilde{W})(\widetilde{W}^\top K^{-1/2}K) = KWW^\top K \tag{2.13}$$

where $W = K^{-1/2}\widetilde{W}$. This kernel $\widetilde{K}$ facilitates a readily parametric form for out-of-sample kernel evaluations. On using the definition of $\widetilde{K}$ in (2.13), the MMD distance between the two domains X'_{src} and X'_{tar} can be written as:

$$Dist(X'_{src}, X'_{tar}) = \text{trace}((KWW^\top K)L) = \text{trace}(W^\top KLKW) \tag{2.14}$$

In minimizing (2.14), a regularization term $\text{trace}(W^\top W)$ is usually needed to control the complexity of W. This regularization term can also avoid the rank deficiency of the denominator in the generalized eigenvalue decomposition.

Besides reducing the distance between the two marginal distributions, the projection ϕ should also preserve data properties that are useful for the target supervised learning task. As in PCA or KPCA, this can be done by preserving the data variance. Therefore by combining the minimization of distribution difference and preserving of the data variance, the kernel learning problem becomes:

$$\begin{aligned} \min_{W} \quad & \text{trace}(W^\top KLKW) + \mu \ \text{trace}(W^\top W) \\ \text{s.t.} \quad & W^\top KHKW = I_m \end{aligned} \tag{2.15}$$

where $\mu > 0$ is a trade-off parameter, $W^\top KHKW$ is the variance of the pro-

jected samples with $H = I_{n_1+n_2} - (1/n_1+n_2)\mathbf{1}\mathbf{1}^\top$ the centering matrix, $\mathbf{1} \in \mathbb{R}^{n_1+n_2}$ is the column vector with all 1's, and $I_{n1+n2} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$ $I_m \in \mathbb{R}^{m\times m}$ are identity matrix. Though this optimization problem involves a non-convex norm constraint $W^\top KHKW = I_m$, the authors proved that it can still be solved efficiently by the following trace optimization problem:

$$\max_W \ \text{trace}((W^\top(KLK + \mu I_m)W)^{-1}W^\top KHKW) \tag{2.16}$$

Similar to kernel Fisher discriminant analysis [Muller *et al.* 2001], the W solutions in (2.16) are the m leading eigenvectors of $(KLK + \mu I)^{-1}KHK$, where $m \leqslant n_1 + n_2 - 1$. This unsupervised approach is named TCA (Transfer Component Analysis), based on this, the authors also proposed a semi-supervised version SSTCA. The effectiveness and efficiency of TCA and SSTCA are verified by experiments on five toy datasets and two real-world applications: cross-domain indoor WiFi localization and cross-domain text classification.

2.1.2.4 Joint Distribution Adaptation

The previous three works all focus on adapting the marginal distribution difference between source and target data, while assuming that the conditional distributions of source and target data in the learned new feature space are equal so that a classifier learned on source data can be directly applied to target data. However in reality the equality assumption of conditional distributions is strong and cannot always be respected. In [Long *et al.* 2013b] the authors proposed a transfer learning approach, referred to as *Joint Distribution Adaptation* (JDA), which aims to jointly adapt both the marginal distribution and conditional distribution in a principled dimensionality reduction procedure. Similar to MMDE and TCA, which are introduced previously, JDA also make use of *Maximum Mean Discrepancy* as the distance measure between distributions.

Assume $(\mathbf{x}_s, y_s)$ represent a labeled sample from source training set, $\mathbf{x}_t$ represent an unlabeled sample from target training set, $P(\mathbf{x}_s)$ and $P(\mathbf{x}_t)$ represent the marginal distributions of source and target data respectively, $P(y_s|\mathbf{x}_s)$ and $P(y_t|\mathbf{x}_t)$

represent the conditional distributions of source and target data respectively. The authors propose to adapt the joint distributions by a feature transformation T so that the joint expectations of the features $\mathbf{x}$ and labels y are matched between domains:

$$\begin{aligned} \min_{T} \ & \| \mathbb{E}_{P(\mathbf{x}_s, y_s)}[T(\mathbf{x}_s), y_s] - \mathbb{E}_{P(\mathbf{x}_t, y_t)}[T(\mathbf{x}_t), y_t] \|^2 \\ \approx \ & \| \mathbb{E}_{P(\mathbf{x}_s)}[T(\mathbf{x}_s)] - \mathbb{E}_{P(\mathbf{x}_t)}[T(\mathbf{x}_t)] \|^2 \\ + \ & \| \mathbb{E}_{P(y_s|\mathbf{x}_s)}[y_s|T(\mathbf{x}_s)] - \mathbb{E}_{P(y_t|\mathbf{x}_t)}[y_t|T(\mathbf{x}_t)] \|^2 \end{aligned} \tag{2.17}$$

Similar to TCA, the authors first use PCA and MMD to reduce the marginal distributions. Denote $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{m \times n}$ the input data matrix, and $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}$ the centering matrix, where $n = n_s + n_t$ and $\mathbf{1}$ the $n \times n$ matrix of ones, then the co-variance matrix can be computed as $\mathbf{XHX}^\top$. The learning goal of PCA is to find an orthogonal transformation matrix $\mathbf{A} \in \mathbb{R}^{m \times k}$ such that the embedded data variance is maximized:

$$\max_{\mathbf{A}^\top \mathbf{A} = \mathbf{I}} \ \text{trace}(\mathbf{A}^\top \mathbf{XHX}^\top \mathbf{A}) \tag{2.18}$$

To reduce the difference between marginal distributions, the empirical MMD, which computes the distance between the sample means of the source and target data in the k-dimensional embeddings, should be minimized:

$$\begin{aligned} \text{Dist}(P(\mathbf{x}_s), P(\mathbf{x}_t)) &= \| \frac{1}{n_s} \sum_{i=1}^{n_s} \mathbf{A}^\top \mathbf{x}_i - \frac{1}{n_t} \sum_{j=n_s+1}^{n_s+n_t} \mathbf{A}^\top \mathbf{x}_j \|^2 \\ &= \text{trace}(\mathbf{A}^\top \mathbf{X} \mathbf{M}_0 \mathbf{X}^\top \mathbf{A}) \end{aligned} \tag{2.19}$$

where $\mathbf{M}_0$ is the MMD matrix and is computed as:

$$(M_0)_{ij} = \begin{cases} \frac{1}{n_s n_s}, & \text{if } \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_s \\ \frac{1}{n_t n_t}, & \text{if } \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_t \\ \frac{-1}{n_s n_t}, & \text{otherwise.} \end{cases} \tag{2.20}$$

By minimizing equation (2.19) such that equation (2.18) is maximized, the marginal distributions between domains are drawn close under the new representation $\mathbf{Z} = \mathbf{A}^\top \mathbf{X}$.

Since in this work the authors assume that no labeled sample is provided in target training set, to reduce the conditional distributions, the authors propose to explore the *pseudo* labels of the target data, which can be predicted by applying some base classifiers trained on the labeled source data to the unlabeled target data. Furthermore, they authors 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 source labels and pseudo target labels, we can match the class-conditional distributions $P(\mathbf{x}_s|y_s = c)$ and $P(\mathbf{x}_t|y_t = c)$ with respect to each class $c \in \{1, \ldots, C\}$ in the label set $\mathcal{Y}$. The authors modify MMD to measure the distance between the class-conditional distributions:

$$\begin{aligned} &\text{Dist}(P(\mathbf{x}_s|y_s = c), P(\mathbf{x}_t|y_t = c)) \\ &\quad = \| \frac{1}{n_s^{(c)}} \sum_{\mathbf{x}_i \in \mathcal{D}_s^{(c)}} \mathbf{A}^\top \mathbf{x}_i - \frac{1}{n_t^{(c)}} \sum_{\mathbf{x}_j \in \mathcal{D}_t^{(c)}} \mathbf{A}^\top \mathbf{x}_j \|^2 \\ &\quad = \text{trace}(\mathbf{A}^\top \mathbf{X} \mathbf{M}_c \mathbf{X}^\top \mathbf{A}) \end{aligned} \tag{2.21}$$

where $\mathcal{D}_s^{(c)} = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathcal{D}_s \wedge y(\mathbf{x}_i) = c\}$ is the set of examples belonging to class c in the source data, $y(\mathbf{x}_i)$ is the true label of $\mathbf{x}_i$, and $n_s^{(c)} = |\mathcal{D}_s^{(c)}|$. Correspondingly, $\mathcal{D}_t^{(c)} = \{\mathbf{x}_j \mid \mathbf{x}_j \in \mathcal{D}_t \wedge \widehat{y}(\mathbf{x}_j) = c\}$ is the set of examples belongs to class c in the target data, $\widehat{y}(\mathbf{x}_j)$ is the pseudo (predicted) label of $\mathbf{x}_j$, and $n_t^{(c)} = |\mathcal{D}_t^{(c)}|$. Thus the MMD matrices $\mathbf{M}_c$ involving class labels are computed as follows:

$$(M_c)_{ij} = \begin{cases} \frac{1}{n_s^{(c)} n_s^{(c)}}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_s^{(c)} \\ \frac{1}{n_t^{(c)} n_t^{(c)}}, & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_t^{(c)} \\ \frac{-1}{n_s^{(c)} n_t^{(c)}}, & \begin{cases} \mathbf{x}_i \in \mathcal{D}_s^{(c)}, \mathbf{x}_j \in \mathcal{D}_t^{(c)} \\ \mathbf{x}_i \in \mathcal{D}_t^{(c)}, \mathbf{x}_j \in \mathcal{D}_s^{(c)} \end{cases} \\ 0, & \text{otherwise.} \end{cases} \tag{2.22}$$

By minimizing equation (2.21) such that equation (2.18) is maximized, the conditional distribution between domains are drawn close under the new representation $\mathbf{Z} = \mathbf{A}^\top \mathbf{X}$. To achieve effective and robust transfer learning, JDA aim to simultaneously minimize the differences in both the marginal distributions and conditional distributions across domains. By incorporating equations (2.19) and (2.21) into equation (2.18), we can get the JDA optimization problem:

$$\min_{\mathbf{A}^\top \mathbf{X}\mathbf{H}\mathbf{X}^\top \mathbf{A} = \mathbf{I}} \sum_{c=0}^{C} \text{trace}(\mathbf{A}^\top \mathbf{X} \mathbf{M}_c \mathbf{X}^\top \mathbf{A}) + \lambda \|\mathbf{A}\|_F^2 \tag{2.23}$$

where λ is the regularization parameter. Based on the generalized Rayleigh quotient, minimizing equations (2.19) and (2.21) such that equation (2.18) is maximized is equivalent to minimizing equation (2.19) and (2.21) such that equation (2.18) is fixed. The previously introduced TCA can be viewed as a special case of JDA with $C = 0$.

For nonlinear problems, consider kernel mapping $\psi : \mathbf{x} \to \psi(\mathbf{x})$, and kernel matrix $\mathbf{K} = \psi(\mathbf{X})^\top \psi(\mathbf{X}) \in \mathbb{R}^{n \times n}$, They use the Representer theorem to formulate the Kernel-JDA as:

$$\min_{\mathbf{A}^\top \mathbf{K}\mathbf{H}\mathbf{K}^\top \mathbf{A} = \mathbf{I}} \sum_{c=0}^{C} \text{trace}(\mathbf{A}^\top \mathbf{K} \mathbf{M}_c \mathbf{K}^\top \mathbf{A}) + \lambda \|\mathbf{A}\|_F^2 \tag{2.24}$$

The problem of finding the optimal adaptation matrix $\mathbf{A}$ as defined in (2.23) can be reformulated as a generalized eigen decomposition problem. The authors proposed an iterative approach where in each iterate they first solve the generalized eigen decomposition problem, then use the new adaptation matrix $\mathbf{A}$ to get the

new features and train a standard classifier, the pseudo target labels can then be updated with this new classifier to form a new eigen decomposition problem which can be solved in the next iterate. This EM-like pseudo label refinement procedure continues until convergence of the pseudo labels.

The authors performed experiments for image classification problems to evaluate the JDA approach. The results verified that JDA can outperform other methods (including TCA) on four types of cross-domain image classification problems.

2.1.2.5 Adaptation with Subspace Alignment

In [Fernando *et al.* 2013] the authors propose a different method compared with the previously introduced ones. They propose to use two PCAs as dimension reduction on source domain and target domain respectively. Following theoretical recommendations of Shai-Ben David's research, this method designs two different subspaces to represent the two different domains, rather than to drag different domains into a common shared subspace. This goal is achieved via optimizing a mapping function that transforms the source subspace into the target one. The authors design a new domain adaptation approach based on subspace alignment.

Firstly, they transform every source and target sample in the form of a D-dimensional z-normalized vector (*i.e.* a vector of zeros mean and unit standard deviation). Then, by using PCA, they select for each domain d eigenvectors corresponding to the d largest eigenvalues. These eigenvectors are selected as bases of the source and target subspaces, which are denoted by X_S and X_T respectively ($X_S, X_T \in \mathbb{R}^{D\times d}$). Note that X'_S and X'_T are orthonormal, which means $X'_S X_S = \mathbf{I}_d$, $X'_T X_T = \mathbf{I}_d$ ($\mathbf{I}_d$ is the identity matrix of size d).

As projecting two different domains into an intermediate common shared subspace may lead to information loss in both source and target domains. The authors suggest to project source and target data to their corresponding subspaces X_S and X_T respectively. Suppose a source sample $\mathbf{y}_S \in \mathbb{R}^{1\times D}$ and a target sample $\mathbf{y}_T \in \mathbb{R}^{1\times D}$, the projection is done by $\mathbf{y}_S X_S$ and $\mathbf{y}_T X_T$. They then provide a transformation matrix M from X_S to X_T, which is supposed to connect the two domains and reduce the divergence between the two domains. The transformation

matrix M is learned via minimizing the following Bregman matrix divergence:

$$F(M) = \|X_S M - X_T\|_F^2 \tag{2.25}$$

$$M^* = \arg\min(F(M)) \tag{2.26}$$

where $\|\cdot\|_F^2$ is the Frobenius norm. Since X_S and X_T are generated from eigenvectors, they are intrinsically regularized. The Frobenius norm is invariant to orthonormal operations, therefore equation (2.25) could be rewritten as follows:

$$F(M) = \left\|X'_S X_S M - X'_S X_T\right\|_F^2 = \left\|M - X'_S X_T\right\|_F^2 \tag{2.27}$$

The optimal M^* could therefore be obtained as $M^* = X'_S X_T$, which implies that the new coordinate system is equivalent to $X_a = X'_S X_S X_T$. This X_a is called the *target aligned source coordinate system.* When source and target domains are the same, then $X_S = X_T$ and M^* is the identity matrix.

A novel similarity function $Sim(\mathbf{y}_S, \mathbf{y}_T)$ is defined as follows to compare a source sample $\mathbf{y}_S$ with a target sample $\mathbf{y}_T$:

$$Sim(\mathbf{y}_S, \mathbf{y}_T) = (\mathbf{y}_S X_S M^*)(\mathbf{y}_T X_T)' = \mathbf{y}_S A \mathbf{y}'_T \tag{2.28}$$

where $A = X_S M^* X'_T$. This $Sim(\mathbf{y}_S, \mathbf{y}_T)$ could be used directly to perform a k-nearest neighbor classification task. An alternative solution is to firstly project the source data via X_a into the target aligned source subspace and project the target data into the target subspace using X_T, then learn a SVM from this d-dimensional space.

In this method, the unique hyper parameter is d, the number of eigenvectors. The authors have derived an upper bound on the similarity function $Sim(\mathbf{y}_S, \mathbf{y}_T)$, which corresponds to d. And they show that d could be efficiently tuned with this bound to guarantee the solution M^* being stable and not over-fitting.

2.1.2.6 Joint Geometrical and Statistical Alignment

The Subspace Alignment (SA) method which is introduced in the previous subsection does not assume that there exist a unified transformation to reduce the domain shifts. The variance of projected source domain data will be different from that of target domain after mapping the source subspace using a linear map because of the domain shift. Therefore, SA fails to minimize the distribution mismatch between domains after aligning the subspaces. In addition, SA cannot deal with situations where the shift between two subspaces are nonlinear. Subspace Distribution Alignment (SDA) [Sun & Saenko 2015] improves SA by considering the variance of the orthogonal principal components. However, the variances are considered based on the aligned subspaces. Hence, only the magnitude of each eigen direction is changed which may still fail when the domain shift is large. To solve this problem, in [Zhang *et al.* 2017] a unified framework that reduces the shift between domains both statistically and geometrically is proposed, which is referred to as Joint Geometrical and Statistical Alignment (JGSA).

JGSA reduces the domain divergence both statistically and geometrically by exploiting both shared and domain specific features of two domains. The JGSA is formulated by finding two coupled projections (A for source domain, and B for target domain) to obtain new representations of respective domains, such that (1) the variance of target domain is maximized, (2) the discriminative information of source domain is preserved, (3) the divergence of source and target distributions is small, and (4) the divergence between source and target subspaces is small.

The overall objective function of JGSA is defined as follows:

$$\max \frac{\mu\{\text{Target Var.}\} + \beta\{\text{Between Class Var.}\}}{\{\text{Distribution shift}\} + \lambda\{\text{Subspace shift}\} + \beta\{\text{Within Class Var.}\}} \tag{2.29}$$

where λ, μ and β are trade-off parameters to balance the importance of each quantity, and Var. indicates variance.

Denote the source domain data as $X_s \in \mathbb{R}^{D \times n_s}$ and the target domain data as $X_t \in \mathbb{R}^{D \times n_t}$, where D is the dimension of the data instance, n_s and n_t are

number of samples in source and target domain respectively. The target variance maximization is achieved as follows:

$$\max_{B} Tr(B^\top S_t B) \tag{2.30}$$

where $S_t = X_t H_t X_t^\top$ is the target domain scatter matrix, $H_t = I_t - \frac{1}{n_t} 1_t 1_t^\top$ is the centering matrix, $1_t \in \mathbb{R}^{n_t}$ is the column vector with all ones.

The source discriminative information is preserved by:

$$\max_{A} Tr(A^\top S_b A) \tag{2.31}$$

$$\max_{A} Tr(A^\top S_w A) \tag{2.32}$$

where S_w is the within class scatter matrix, and S_b is the between class scatter matrix of the source domain data.

They employ the MMD criteria to compare the distributions between domains, which computes the distance between the sample means of the source and target data in the k-dimensional embeddings. Then they follow the idea of JDA (which is introduced previously in subsection 2.1.2.4) to minimize the conditional distribution shift between domains. By combining the marginal and conditional distribution shift minimization terms, the final distribution divergence minimization term is defined as:

$$\min_{A,B} Tr\left(\begin{bmatrix} A^\top & B^\top \end{bmatrix} \begin{bmatrix} M_s & M_{st} \\ M_{ts} & M_t \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} \right) \tag{2.33}$$

where M_s, M_t, M_{st} and M_{ts} construct the relationships between source domain data and target domain data.

Similar to SA, they also reduce the discrepancy between domains by moving closer the source and target subspaces. The subspace divergence minimization is achieved by:

$$\min_{A,B} \|A - B\|_F^2 \tag{2.34}$$

By using term (2.34) together with (2.33), both shared and domain specific features are exploited such that the two domains are well aligned geometrically and statistically.

Finally, by incorporating the above five quantities (2.30), (2.31), (2.32), (2.33) and (2.34) together, the objective function (2.29) could be rewritten as follows:

$$\max_{A,B} \frac{Tr\left(\begin{bmatrix} A^\top & B^\top \end{bmatrix} \begin{bmatrix} \beta S_b & \mathbf{0} \\ \mathbf{0} & \mu S_t \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix}\right)}{Tr\left(\begin{bmatrix} A^\top & B^\top \end{bmatrix} \begin{bmatrix} M_s + \lambda I + \beta S_w & M_{st} - \lambda I \\ M_{ts} - \lambda I & M_t + (\lambda + \mu) I \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix}\right)} \tag{2.35}$$

where $I \in \mathbb{R}^{d \times d}$ is the identity matrix. Minimizing the denominator of (2.35) encourages small marginal and conditional distribution shifts, and small within class variance in the source domain. Maximizing the numerator of (2.35) encourages large target domain variance, and large between class variance in the source domain. Similar to JDA, they also iteratively update the pseudo labels of target domain data using the learned transformations to improve the labeling quality until convergence.

2.1.3 Representation learning as metric learning

Another group of transfer learning methods is to cast the representation learning (or subspace learning) problem into the metric learning scenario.

One of the first works is [Fink 2005], which tries to learn a shared feature representation using metric learning disciplines. Similar to the very first transfer learning method [Thrun 1996], this algorithm learns a feature transformation which is later utilized 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 follows a max-margin approach to directly learn a distance metric.

Consider learning a function $d : X \times X \to \mathbb{R}$ which has the following properties: (i) $d(x, x') \geqslant 0$, (ii) $d(x, x') = d(x', x)$ and (iii) $d(x, x') + d(x', x'') \geqslant d(x, x'')$. A function satisfying these three properties is called a pseudo-metric. Ideally, we would like to learn a function d that assigns a smaller distance to pairs having the same label than to pairs with different labels. More precisely, for any positive samples $\mathbf{x}_i$, $\mathbf{x}_j$, and negative sample $\mathbf{x}_k$, we would require the difference in distance to be at least γ, :

$$d(\mathbf{x}_i, \mathbf{x}_j) \leqslant d(\mathbf{x}_i, \mathbf{x}_k) - \gamma \tag{2.36}$$

Fink's algorithm make use of a pseudo-metric of the form: $d(\mathbf{x}_i, \mathbf{x}_j)^2 = \| \theta\mathbf{x}_i - \theta\mathbf{x}_j \|_2^2$, the problem is therefore to learn a linear projection θ that achieves γ separation as defined in (2.36). This projection is learned with source data, and then is deployed for classification of target data. The underlying transfer learning assumption is that a projection θ that achieves γ separation on the source tasks will most likely achieve γ separation on the target task. Therefore, if we project the target samples using θ and run a nearest neighbor classifier in the new space we are likely to get a good performance.

Like the early works introduced in subsection 2.1.1, 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 [Parameswaran & Weinberger 2010] 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 idiosyncrasies with additional matrices $\mathbf{M}_1, \ldots, \mathbf{M}_T \succeq 0$. The distance for task t is defined as follows:

$$d_t(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{M}_0 + \mathbf{M}_t)(\mathbf{x}_t - \mathbf{x}_j)} \tag{2.37}$$

Although there is not a specific projection as θ defined in [Fink 2005], this distance defined in Eq. (2.37) 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. An illustration is shown in Figure 2.1. They authors have proved theoretically that the Eq. 2.37 is a well defined pseudo-metric when the matrices $\mathbf{M}_t$ are constrained to be positive semi-definite (*i.e.* $\mathbf{M}_t \succeq 0$).

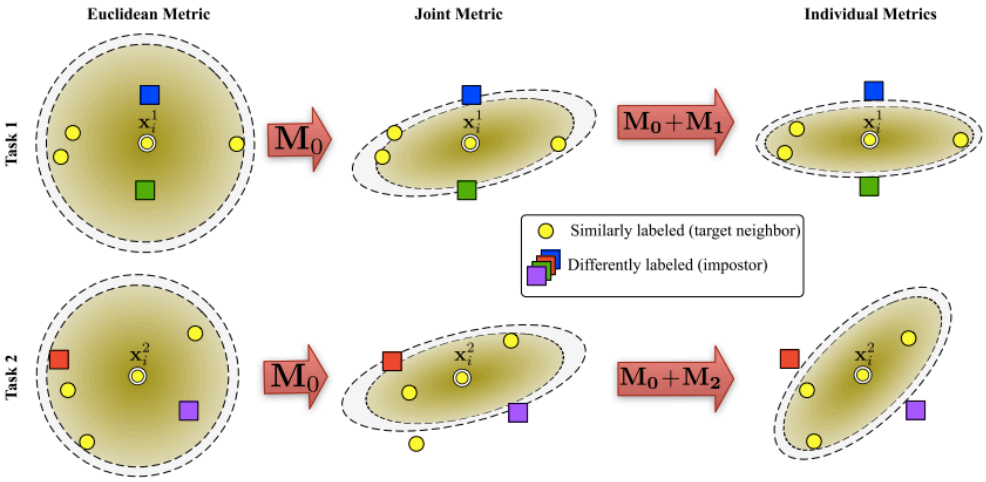

Figure 2.1: Figure from [Parameswaran & Weinberger 2010]: An illustration of mt-lmnn(Multi task large margin nearest neighbors). The matrix $\mathbf{M}_0$ captures the communality between the several tasks, whereas $\mathbf{M}_t$ for $t > 0$ adds the task specific distance transformation.

To ensure that the learning algorithm does not put too much emphasis onto the shared parameters $\mathbf{M}_0$ or the individual parameters $\mathbf{M}_1, \ldots, \mathbf{M}_T$, they use the regularization term as follows:

$$\min_{\mathbf{M}_0,\ldots,\mathbf{M}_T} \gamma_0 \|\mathbf{M}_0 - \mathbf{I}\|_F^2 + \sum_{t=1}^{T} \gamma_t \|\mathbf{M_t}\|_F^2 \tag{2.38}$$

The trade-off parameter γ_t controls the regularization of $\mathbf{M}_t$ for all $t = 0, 1, \ldots, T$. If $\gamma_0 \to \infty$, the shared metric $\mathbf{M}_0$ reduces to the plain Euclidean metric and if $\gamma_{t>0} \to \infty$, the task-specific metrics $\mathbf{M}_{t>0}$ become irrelevant zero matrices. Therefore, if $\gamma_{t>0} \to \infty$ and γ_0 is small, a single metric $\mathbf{M}_0$ across all tasks will be learned, the result is equivalent to applying lmnn (large margin nearest neighbors) on the union of all data sets. In the other extreme case, when $\gamma_0 = 0$ and $\gamma_{t>0} \to \infty$,

the formulation will reduce to T independent lmnn algorithms.

Let S_t be the set of triples restricted to only vectors for task t, *i.e.*, $S_t = \{(i,j,k) \in \mathcal{J}_t^3 : j \rightsquigarrow i, y_k \neq y_i\}$. Where $j \rightsquigarrow i$ indicates that $\mathbf{x}_j$ is a target neighbor of $\mathbf{x}_i$, and $\mathcal{J}_t$ is the set of indexes such that $i \in \mathcal{J}_t$ if and only if the input-label pair $(\mathbf{x}_i, y_i)$ belongs to task t. Combine the regularizer in Eq. 2.38 with the objective of lmnn applied to each of the T tasks. To ensure well-defined metrics, the authors also add constraints that each matrix is positive semi-definite, *i.e.* $\mathbf{M}_t \succeq 0$. The resulting algorithm is called *multi-task large margin nearest neighbor*(mt-lmnn). The optimization problem is shown in Eq (2.39) and can be solved after some modifications to the special-purpose solver presented in [Weinberger & Saul 2009].

$$\min_{\mathbf{M}_0,\ldots,\mathbf{M}_T} \gamma_0 \|\mathbf{M}_0 - \mathbf{I}\|_F^2 + \sum_{t=1}^{T} \left[\gamma_t \|\mathbf{M}_t\|_F^2 + \sum_{(i,j)\in\mathcal{J}_{t,j\rightsquigarrow i}} d_t^2(\mathbf{x}_i, \mathbf{x}_j) + \sum_{(i,j,k)\in S_t} \xi_{ijk} \right]$$

subject to: $\forall t, \forall (i,j,k) \in S_t$:

(1) $d_t^2(\mathbf{x}_i, \mathbf{x}_k) - d_t^2(\mathbf{x}_i, \mathbf{x}_j) \geqslant 1 - \xi_{ijk}$

(2) $\xi_{ijk} \geqslant 0$

(3) $\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_T \succeq 0.$

(2.39)

This regularization term $\gamma_0 \|\mathbf{M}_0 - \mathbf{I}\|_F^2$ could be interpreted as learning $\mathbf{M}_0$ while trying to stay close to the Euclidean distance. Another kind of metric regularization for transfer learning is to replace $\mathbf{I}$ with the auxiliary metric $\mathbf{M}_S$ learned from source task. The regularization $\|\mathbf{M} - \mathbf{M}_S\|$ could be interpreted as transferring knowledge brought by $\mathbf{M}_S$ for learning $\mathbf{M}$. This setting is similar to some domain adaptation methods introduced in the previous section 2.1.2. In domain adaptation methods the source metric and target metric are usually learned simultaneously by using the source and target training samples. However it is sometimes impossible to have access to all the training samples. In [Perrot & Habrard 2015] 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 showing that supervised regularized metric learning approaches using a biased regularization are well-founded. Their analysis is based on algorithmic stability arguments allowing one to derive generalization guarantees when a learning algorithm does not suffer too much from a little change in the training sample. Firstly they introduced a new notion of stability called *on-average-replace-two-stability* that is well-suited to regularized metric learning formulations. This notion allows one to prove a high probability generalization bound for metric hypothesis transfer learning achieving a fast converge rate in $O(\frac{1}{n})$ in the context of admissible, lipschitz and convex losses. Secondly, they provided a consistency result from which they justify the interest of *weighted biased regularization* of the form $\|\mathbf{M} - \beta\mathbf{M}_S\|$ where β is a parameter to set. They derive an approach for assessing this parameter without resorting to a costly parameter tuning procedure. They also provided an experimental study showing the effectiveness of transfer metric learning with weighted biased regularization in the presence of few labeled data both on standard metric learning an transfer learning tasks.

2.1.4 Representation Learning with Deep Neural Networks

In the previous sections 2.1.1 and 2.1.2 we've introduced some transfer learning algorithms using traditional shallow feature learning methods for transferable feature learning. Recently Deep Neural Networks have gained great success on feature learning, which outperform shallow models for various applications including image classification. Furthermore, recent studies reveal that a deep neural network can learn transferable features which generalize well to novel tasks. In this section, we introduce some recent works on transferable feature learning with deep neural networks. Similarly to previous two sections, the methods in this section are proposed for domain adaptation problem. They extend deep convolutional neural networks (CNNs) to domain adaptation either by adding one or multiple adaptation layers through which the mean embeddings of distributions are matched [Tzeng *et al.* 2014] [Long *et al.* 2015], or by adding a fully connected subnetwork as a domain discriminator whilst the deep features are learned to confuse the domain

discriminator in a domain-adversarial training paradigm [Ganin & Lempitsky 2015] [Tzeng *et al.* 2015]. In the following we will show details of two representative methods.

2.1.4.1 Adaptation with MMD: Deep Adaptation Networks

In [Long *et al.* 2015] the authors proposed a Deep Adaptation Network (DAN) architecture, which generalizes deep Convolutional Neural Networks (CNN) to the domain adaptation scenario. Similar to JDA, which is introduced in previous section 2.1.2, DAN also use Maximum Mean Discrepancy as distance measure for adapting source and target distributions. In this work they use a multi-kernel version of MMD, named MK-MMD and proposed by [Gretton *et al.* 2012], which is formalized to jointly maximize the two-sample test power and minimize the Type II error, *i.e.*, the failure of rejecting a false null hypothesis.

Assume having a set of labeled source training samples $\{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ and a set of unlabeled target training samples $\{\mathbf{x}_j^t\}_{j=1}^{n_t}$. Denote by $\mathcal{H}_k$ be the reproducing kernel Hilbert space (RKHS) endowed with a characteristic kernel k. The *mean embedding* of distribution p in $\mathcal{H}_k$ is a unique element $\mu_k(p)$ such that $\mathbf{E}_{\mathbf{x}\sim p} f(\mathbf{x}) = \langle f(\mathbf{x}), \mu_k(p)\rangle_{\mathcal{H}_k}$ for all $f \in \mathcal{H}_k$. The MK-MMD $d_k(p, q)$ between probability distributions p and q is defined as the RKHS distance between the mean embeddings of p and q. The squared formulation of MK-MMD is defined as:

$$d_k^2(p, q) \triangleq \|\mathbf{E}_p[\phi(\mathbf{x}^s)] - \mathbf{E}_q[\phi(\mathbf{x}^t)]\|_{\mathcal{H}_k}^2 \tag{2.40}$$

A property of this distance is that $p = q$ if and only if $d_k^2(p, q) = 0$. The characteristic kernel k associated with the feature map ϕ is defined as the convex combination of m PSD (Positive Semi-Definite) kernels $\{k_u\}_{u=1}^m$. As studied theoretically in [Gretton *et al.* 2012], the kernel adopted for the mean embeddings of p and q is critical to ensure the test power and low test error. The multi-kernel k can leverage different kernels to enhance MK-MMD test, leading to a principled method for optimal kernel selection.

Using the kernel trick, MK-MMD (2.40) can be computed as the expectation of

kernel functions:

$$d_k^2(p,q) = \mathbf{E}_{\mathbf{x}^s\mathbf{x}'^s} k(\mathbf{x}^s, \mathbf{x}'^s) + \mathbf{E}_{\mathbf{x}^t\mathbf{x}'^t} k(\mathbf{x}^t, \mathbf{x}'^t) - 2\mathbf{E}_{\mathbf{x}^s\mathbf{x}^t} k(\mathbf{x}^s, \mathbf{x}^t) \tag{2.41}$$

where $\mathbf{x}^s, \mathbf{x}'^s \sim p$, $\mathbf{x}^t, \mathbf{x}'^t \sim q$, and $k \in \mathcal{K}$. However, this computation incurs a complexity of $O(n^2)$ which is undesirable for deep CNNs. Therefore the authors propose to adopt the unbiased estimate of MK-MMD [Gretton *et al.* 2012] which can be computed with linear complexity $O(n)$. Specifically:

$$d_k^2(p,q) = \frac{2}{n_s} \sum_{i=1}^{n_s/2} g_k(\mathbf{z}_i) \tag{2.42}$$

where $\mathbf{z}_i$ is a quad-tuple defined as: $\mathbf{z}_i \triangleq (\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s, \mathbf{x}_{2i-1}^t, \mathbf{x}_{2i}^t)$, and the multi-kernel function k on each quad-tuple $\mathbf{z}_i$ is evaluated by:

$$g_k(\mathbf{z}_i) \triangleq k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s) + k(\mathbf{x}_{2i-1}^t, \mathbf{x}_{2i}^t) - k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^t) - k(\mathbf{x}_{2i}^s, \mathbf{x}_{2i-1}^t) \tag{2.43}$$

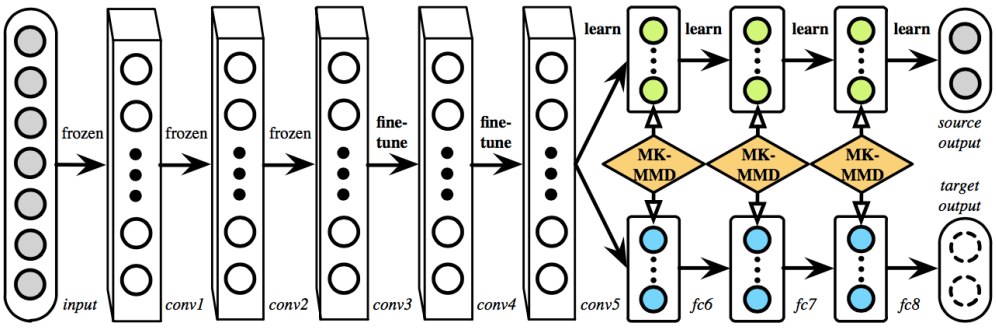

Figure 2.2: Figure from [Long *et al.* 2015]: The DAN architecture for learning transferable features. Since deep features eventually transition from general to specific along the network, (1) the features extracted by convolutional layers *conv1* – *conv3* are general, hence these layers are frozen, (2) the features extracted by layers *conv4* – *conv5* are slightly less transferable, hence these layers are learned via fine-tuning, and (3) fully connected layers *fc6* – *fc8* are tailored to fit specific tasks, hence they are not transferable and should be adapted with MK-MMD.

To learn an optimal feature transformation, the authors propose to extend the AlexNet architecture [Krizhevsky *et al.* 2012], which is comprised of five convolutional layers (*conv1* – *conv5*) and three fully connected layers (*fc6* – *fc8*). Each *fc* layer ℓ learns a nonlinear mapping $\mathbf{h}_i^\ell = f^\ell(\mathbf{W}^\ell \mathbf{h}_i^{\ell-1} + \mathbf{b}^\ell)$, where $\mathbf{h}_i^\ell$ is the ℓ-th layer hidden representation of point $\mathbf{x}_i$, $\mathbf{W}^\ell$ and $\mathbf{b}^\ell$ are the weights and bias of the ℓ-th layer, and f^ℓ is the activation, taking as rectifier units $f^\ell(\mathbf{x}) = max(\mathbf{0}, \mathbf{x})$ for hidden layers or softmax units $f^\ell(\mathbf{x}) = \frac{e^{\mathbf{x}}}{\sum_{j=1}^{|\mathbf{x}|} e^{x_j}}$ for the output layer. Letting $\Theta = \{\mathbf{W}^\ell, \mathbf{b}^\ell\}_{\ell=1}^{l}$ denote the set of all CNN parameters, the objective of learning a CNN is to minimize its empirical risk:

$$\min_{\Theta} \frac{1}{n_a} \sum_{i=1}^{n_a} J(\theta(\mathbf{x}_i^a), y_i^a) \tag{2.44}$$

where J is the cross-entropy loss function, and $\theta(\mathbf{x}_i^a)$ is the conditional probability that the CNN assigns $\mathbf{x}_i^a$ to label y_i^a. Given that the convolutional layers can learn generic features that tend to be transferable in layers *conv1* – *conv3* and are slightly domain-biased in *conv4* – *conv5*, while the higher layers *fc6* – *fc8* are more domain specific which cannot be directly transferred to the target domain via fine-tuning with limited target supervision [Yosinski *et al.* 2014]. The authors therefore propose to freeze *conv1-conv3* and fine-tune *conv4* – *conv5* to preserve the efficacy of fragile co-adaptation, while retrain the *fc6* – *fc8* layers' parameters with requiring the distributions of the source and target to become similar under the hidden representations of these fully connected layers. This can be realized by adding an MK-MMD-based multi-layer adaptation regularizer (2.40) to the CNN risk in (2.44):

$$\min_{\Theta} \frac{1}{n_a} \sum_{i=1}^{n_a} J(\theta(\mathbf{x}_i^a), y_i^a) + \lambda \sum_{\ell=l_1}^{l_2} d_k^2(\mathcal{D}_s^\ell, \mathcal{D}_t^\ell) \tag{2.45}$$

where $\lambda > 0$ is a penalty parameter, l_1 and l_2 are layer indexes between which the regularizer is effective, in DAN the authors set $l_1 = 6$ and $l_2 = 8$. $\mathcal{D}_*^\ell = \{\mathbf{h}_i^{*\ell}\}$ is the ℓ-th layer hidden representation for the source and target examples, and $d_k^2(\mathcal{D}_s^\ell, \mathcal{D}_t^\ell)$ is the MK-MMD between the source and target evaluated on the ℓ-th

layer representation. An illustration of the whole DAN architecture could be found in Figure 2.2.

The authors propose to initialize the parameters in DAN with the parameters of an AlexNet model pre-trained on ImageNet 2012. The training process is therefore a fine-tuning of this pre-trained model on source and target training data.

They perform experiments of DAN compared to other transfer learning methods and deep learning methods on both unsupervised and semi-supervised adaptation problems. The results verified the efficacy of the proposed DAN against previous methods.

This work is further improved in [Long *et al.* 2016], in which the authors proposed a Residual Transfer Network which can adapt both marginal distributions and conditional distributions at the same time.

2.1.4.2 Adaptation with Adversarial Networks

In [Ganin & Lempitsky 2015] the authors proposed a new approach to domain adaptation in deep architectures. As the training progresses, the approach promotes the emergence of "deep" features that are (i) discriminative for the main learning task on the source domain and (ii) invariant with respect to the shift between the domains. This adaptation behavior is achieved by augmenting a feed-forward model with few standard layers and a new *gradient reversal* layer. The resulting augmented architecture can be trained using standard back-propagation.

Assume that the model works with input samples $\mathbf{x} \in X$ where X is some input space and certain labels (output) y from the label space Y. They consider classification problems where Y is a finite set $Y = \{1, 2, \ldots, L\}$ in the paper, although they claim that their approach is generic and can handle any output label space that other deep feed-forward models can handle. They further assume that there exist two distributions $\mathcal{S}(x, y)$ and $\mathcal{T}(x, y)$ on $X \otimes Y$, which will be referred to as the source distribution and the target distribution. Both distributions are assumed complex and unknown, and furthermore similar but different (in other words, $\mathcal{S}$ is "shifted" from $\mathcal{T}$ by some *domain shift*). Their ultimate goal is to be able to predict labels y given the input $\mathbf{x}$ for the target distribution. At training time, the model

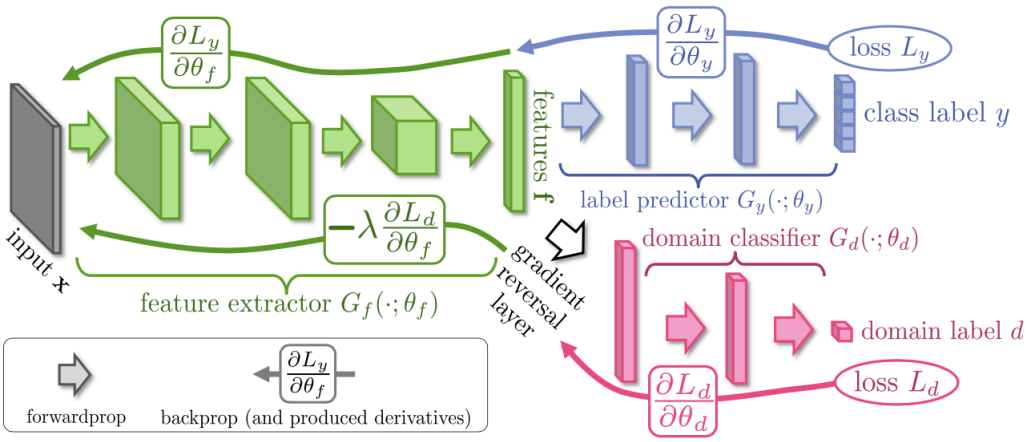

Figure 2.3: Figure from [Ganin & Lempitsky 2015]: The proposed architecture includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the back-propagation-based training. Otherwise, the training proceeds in a standard way and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.

have an access to a large set of training samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ from both the source and the target domains distributed according to the marginal distributions $\mathcal{S}(\mathbf{X})$ and $\mathcal{T}(\mathbf{x})$. They denote with d_i, which is a binary variable, the *domain label* for the i-th sample, which indicates whether x_i come from the source distribution ($d_i = 0$) or from the target distribution ($d_i = 1$). For the samples from the source distributions they also have categories labels $y_i \in Y$.

They define a deep feed-forward architecture that for each input $\mathbf{x}$ predicts its label $y \in Y$ and its domain label $d \in \{0, 1\}$. They decompose such mapping into three parts. They assume that the input $\mathbf{x}$ is first mapped by a mapping G_f (a *feature extractor*) to a D-dimensional feature vector $\mathbf{f} \in \mathbb{R}^D$. The feature mapping may also include several feed-forward layers and we denote the vector of parameters of all layers in this mapping as θ_f, *i.e.* $\mathbf{f} = G_f(\mathbf{x}; \theta_f)$. Then, the feature vector $\mathbf{f}$ is mapped by a mapping G_y (*label predictor*) to the label y, and they denote the parameters of this mapping with θ_y. Finally, the same feature vector $\mathbf{f}$ is mapped

to the domain label d by a mapping G_d (*domain classifier*) with the parameters θ_d (see Figure 2.3).

During the learning stage, they aim to minimize the label prediction loss on the annotated part (*i.e.* the source part) of the training set, and the parameters of both the feature extractor and the label predictor are thus optimized in order to minimize the empirical loss for the source domain samples. This ensures the discriminativeness of the features $\mathbf{f}$ and the overall good prediction performance of the combination of the feature extractor and the label predictor on the source domain.

At the same time, they want to make the features $\mathbf{f}$ domain-variant. That is, they want to make the distributions $S(\mathbf{f}) = \{G_f(\mathbf{x};\theta_f)|\mathbf{x} \sim S(\mathbf{x})\}$ and $T(\mathbf{f}) = \{G_f(\mathbf{x};\theta_f)|\mathbf{x} \sim T(\mathbf{x})\}$ to be similar. To achieve this, they seek the parameters θ_f of the feature mapping that *maximize* the loss of the domain classifier (by making the two feature distributions as similar as possible), while simultaneously seeking the parameters θ_d of the domain classifier that *minimize* the loss of the domain classifier. In addition, they seek to minimize the loss of the label predictor. More formally, they consider the functional:

$$\begin{aligned} E(\theta_f, \theta_y, \theta_d) &= \sum_{\substack{i=1,\dots,N \\ d_i=0}} L_y(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y), y_i) \\ &\quad - \lambda \sum_{i=1,\dots,N} L_d(G_d(G_f(\mathbf{x}_i;\theta_f);\theta_d), y_i) \\ &= \sum_{\substack{i=1,\dots,N \\ d_i=0}} L_y^i(\theta_f, \theta_y) - \lambda \sum_{i=1,\dots,N} L_d^i(\theta_f, \theta_d) \end{aligned} \tag{2.46}$$

where $L_y(\cdot,\cdot)$ is the loss for label prediction (*e.g.* multinomial), $L_d(\cdot,\cdot)$ is the loss for the domain classification (*e.g.*, logistic), while L_y^i and L_d^i denote the corresponding loss functions evaluated at the i-th training example. Based on this, they seek the parameters $\hat{\theta}_f$, $\hat{\theta}_y$, $\hat{\theta}_d$ that deliver a saddle point of the functional (2.46):

$$
\begin{aligned}
(\hat{\theta}_f, \hat{\theta}_y) &= \arg\min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \hat{\theta}_d) \\
\hat{\theta}_d &= \arg\min_{\theta_d} E(\hat{\theta}_f, \hat{\theta}_y, \theta_d)
\end{aligned}
\tag{2.47}
$$

The authors demonstrate that standard stochastic gradient solvers (SGD) can be adapted for the search of this saddle point. They show that a saddle point for (2.46) (2.47) can be found as a stationary point of the following stochastic updates:

$$
\theta_f \leftarrow \theta_f - \mu \left(\frac{\partial L_y^i}{\partial \theta_f} - \lambda \frac{\partial L_d^i}{\partial \theta_f} \right) \tag{2.48}
$$

$$
\theta_y \leftarrow \theta_y - \mu \frac{\partial L_y^i}{\partial \theta_y} \tag{2.49}
$$

$$
\theta_d \leftarrow \theta_d - \mu \frac{\partial L_d^i}{\partial \theta_d} \tag{2.50}
$$

where μ is the learning rate. As direct implementation of (2.48) – (2.50) is not possible, they authors propose to reduce the updates to some form of SGD by introducing a special *gradient reversal layer* (GRL). During the forward propagation, GRL acts as an identity transform. During the back-propagation though, GRL takes the gradient from the subsequent level, multiplies it by $-\lambda$ and passes it to the preceding layer. This GRL is inserted between the feature extractor and the domain classifier, resulting in the architecture depicted in Figure 2.3. Running SGD in this model implements the updates (2.48) – (2.50) and converges to a saddle point of (2.46).

To evaluate the proposed approach, the authors perform experiments on a number of image datasets and their modifications. Results show that their approach outperforms baseline methods and some previous domain adaptation methods.

2.1.5 Representation Learning with Dictionary Learning

Another group of research works on transferable feature learning is based on *dictionary learning* (also known as *sparse coding*). Due to the fact that learning reconstructive dictionary usually don't need labeled data, many of these works are proposed for the self-taught learning scenario in which we are provide with a small set of labeled target training samples and a large set of unlabeled source training samples.

2.1.5.1 Self-taught Learning and a Sparse coding based approach

[Raina *et al.* 2007] is the first work that proposed the *self-taught learning* problem. They proposed an approach to self-taught learning that uses sparse coding to construct higher-level features using the unlabeled data. These features form a succinct input representation and can improve classification performance for the target task.

In self-taught learning, one is given a labeled training set of m samples $\{(x_l^{(1)}, y^{(1)}), (x_l^{(2)}, y^{(2)}), \ldots, (x_l^{(m)}, y^{(m)})\}$ drawn i.i.d. from some distribution $\mathcal{D}$. Here, each $x_l^{(i)} \in \mathbb{R}^n$ is an input feature vector (the "l" subscript indicates that it is a labeled example), and $y^{(i)} \in \{1, \ldots, C\}$ is the corresponding class label. In addition, a set of k unlabeled examples $x_u^{(1)}, x_u^{(2)}, \ldots, x_u^{(k)} \in \mathbb{R}^n$ is also given. The unlabeled data are not assumed to be drawn from the same distribution as the labeled data, nor that it can be associated with the same class labels as the labeled data. While the labeled and unlabeled data should not be completely irrelevant to each other if unlabeled data is to help the target task.

To learn the higher-level representations, the authors proposed a modified version of the sparse coding algorithm [Olshausen & Field 1996]. Specifically, given the unlabeled data $\{x_u^{(1)}, \ldots, x_u^{(k)}\}$ with each $x_u^{(i)} \in \mathbb{R}^n$, the propose the following optimization problem:

$$\min_{b,a} \sum_i \|x_u^{(i)} - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1 \tag{2.51}$$

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

The optimization variables in this problem are the *basis vectors* $b = \{b_1, b_2, \ldots, b_s\}$ with each $b_j \in \mathbb{R}^n$, and the *activations* $a = \{a^{(1)}, \ldots, a^{(k)}\}$ with each $a^{(i)} \in \mathbb{R}^s$, $a_j^{(i)}$ is the activation of basis b_j for input $x_u^{(i)}$. The number of bases s can be much larger than the input dimension n. This optimization objective balances two terms: (1) The first quadratic term encourages each input $x_u^{(i)}$ to be reconstructed well as a weighted linear combination of the bases b_j (with corresponding weights given by the activations $a_j^{(i)}$); (2) The second term encourages the activations to have low L_1 norm, *i.e.*, it encourages the activations a to be *sparse* – in other words, for most of its elements to be zero. The problem (2.51) is convex over each subset of variables a and b (though not jointly convex); in particular, the optimization over activations a is an l_1-regularized least square problem, and the optimization over basis vectors b is an $l2$-constrained least square problem. These two convex sub-problems can be solved efficiently, and the objective in (2.51) can be iteratively optimized over a and b alternatively while holding the other set of variables fixed.

It is often easy to obtain large amounts of unlabeled data that shares several salient features with the labeled data from the target classification task. Building on this observation, the authors propose the following approach to *self-taught learning*: they first apply sparse coding to the unlabeled data $x_u^{(i)} \in \mathbb{R}^n$ to learn a set of bases b. Then for each training input $x_l^{(i)} \in \mathbb{R}^n$ from the target task, they compute features $\hat{a}(x_l^{(i)}) \in \mathbb{R}^s$ by solving the following optimization problem:

$$\hat{a}(x_l^{(i)}) = \arg\min_{a^{(i)}} \|x_l^{(i)} - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1 \tag{2.52}$$

This is a convex $l1$-regularized least square problem and can be solved efficiently. It approximately expresses the input $x_l^{(i)}$ as a sparse linear combination of the bases b_j. The sparse vector $\hat{a}(x_l^{(i)})$ is the new representation for $x_l^{(i)}$. These new features are taken as input to standard supervised classification algorithms (such as SVMs) for target task.

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: first, PCA results in *linear* feature extraction while the sparse coding method learns a nonlinear representation $\hat{a}(x)$ due to the presence of the $l1$ term in equation (2.51); second, since PCA assumes the bases to be orthogonal, the number of PCA features cannot be greater than the dimension n of the input, while sparse coding can use more basis vectors than the input dimension. By learning a large number of basis vectors but using only a small number of them for any particular input, sparse coding gives a higher-level representation in terms of the many possible "basic patterns".

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 [Wang *et al.* 2013] 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.

2.1.5.2 Self-taught Low-rank coding

In a recent work [Li *et al.* 2017] the authors propose a new sparse coding based self-taught learning framework for visual learning, which can utilize the rich low-level pattern information abstracted from the auxiliary domain, in order to characterize the high-level structural information in the target domain. Since many types of visual data have been proven to contain subspace structures, a low-rank constraint is introduced into the coding objective to better characterize the structure of the given target set. This proposed representation learning framework is called self-taught low-rank (S-Low) coding, which can be formulated as a non-convex rank-minimization and dictionary learning problem.

Consider having a set of abundant unlabeled samples $\mathbf{X}_S = \{\mathbf{x}_1^{(S)}, \ldots, x_m^{(S)}\} \in \mathbb{R}^{d\times m}$ in the source domain, and a limited number of samples $\mathbf{X}_T = \{x_1^{(T)}, \ldots, x_n^{(T)}\} \in \mathbb{R}^{d\times n}$ in the target domain. Here d is the feature dimension, m is the number of samples in source training set, and n is the number of samples in target training set. Unlike the previous approach [Raina *et al.* 2007], in this work the authors do not require that the samples in the target domain are labeled. Therefore their framework can be adapted to either unsupervised or supervised situations according to the availability of labels on target training samples.

Conventionally, the sparse coding, dictionary learning or low-rank learning methods approximately represent the samples in a single domain (here we take the target domain as an example) as:

$$\mathbf{X}_T \approx \mathbf{D}_T \mathbf{Z}_T \tag{2.53}$$

where $\mathbf{Z}_T \in \mathbb{R}^{r\times n}$ is the representation coefficient matrix and $\mathbf{D}_T \in \mathbb{R}^{d\times r}$ is a dictionary with r the size of this dictionary. $\mathbf{Z}_T$ is usually expected to be sparse or low-rank, according to the application scenario. To make use of samples in both domains, the authors propose to learn the dictionary from all available samples in two domains (source and target). The whole sample set is denoted as $\mathbf{X} = [\mathbf{X}_S\ \mathbf{X}_T]$. Therefore they propose the following constraint:

$$[\mathbf{X}_S\ \mathbf{X}_T] = \mathbf{D}[\mathbf{Z}_S\ \mathbf{Z}_T] + [\mathbf{E}_S\ \mathbf{E}_T] \tag{2.54}$$

where $\mathbf{Z}_S \in \mathbb{R}^{r\times m}$ and $\mathbf{Z}_T \in \mathbb{R}^{r\times n}$ are the coefficient matrices corresponding to source domain and target domain, respectively. $\mathbf{E}_S \in \mathbb{R}^{d\times m}$ and $\mathbf{E}_T \in \mathbb{R}^{d\times n}$ are the sparse noise matrices that model the reconstruction errors in auxiliary and target domains. The noise matrices $\mathbf{E}_S$ and $\mathbf{E}_T$ are often constrained using the surrogate of $l0$ norm which enables the model to learn a robust dictionary.

To discover the underlying subspace structure in training data, the authors further propose to impose a low-rank constraint on the coefficient matrix $\mathbf{Z}_T$ in the target domain where the learning tasks are performed. The objective function can therefore be formulated as:

$$\begin{aligned} \min_{\mathbf{D},\mathbf{Z}_S,\mathbf{Z}_T,\mathbf{E}_S,\mathbf{E}_T} \quad & \text{rank}(\mathbf{Z}_T) + \lambda_1 \|\mathbf{E}_S\|_0 + \lambda_2 \|\mathbf{E}_T\|_0 \\ \text{s.t.} \quad & \mathbf{X}_S = \mathbf{D}\mathbf{Z}_S + \mathbf{E}_S \\ & \mathbf{X}_T = \mathbf{D}\mathbf{Z}_T + \mathbf{E}_T \end{aligned} \tag{2.55}$$

where rank$(\cdot)$ denotes the rank function, $\|\cdot\|_0$ is the l_0 norm, and λ_1 and λ_2 are two trade-off parameters. In (2.55) the first term characterizes the low rankness of $\mathbf{Z}_T$ in the target domain, and the last two terms model the reconstruction errors. This is a variant of rank minimization problem that is NP-hard in general. Therefore, it cannot be solved directly, normal solution would be to relax the l_0 norm and the rank function with l_1 norm and nuclear norm respectively. However, the authors argue that the l_1 norm and the nuclear norm are biased estimators, as they over penalize large entries and large singular values. Therefore, the authors propose to employ the non-convex surrogates of l_0 norm and rank function, which are MCP norm and matrix γ-norm, respectively.

The definition of matrix MCP norm for a matrix $\mathbf{B} \in \mathbb{R}^{p\times q}$ is:

$$M_{\lambda,\gamma}(\mathbf{B}) = \sum_{i,j} \phi_{\lambda,\gamma}(B_{i,j}) \tag{2.56}$$

$$\phi_{\lambda,\gamma}(t) = \lambda \int_0^t \left[1 - \frac{x}{\gamma\lambda}\right] dx = \begin{cases} \gamma\frac{\lambda^2}{2}, & \text{if } |t| \geq \gamma\lambda \\ \gamma|t| - \frac{t^2}{2\gamma}, & \text{otherwise.} \end{cases}$$

where $[z]_+ = max(z, 0)$, λ is set to 1, and for simplicity denote $M_\gamma(\mathbf{B}) = M_{1,\gamma}(\mathbf{B})$.

The matrix γ-norm is defined as:

$$\begin{aligned} \|\mathbf{B}\|_\gamma &= \sum_{i=1}^{s} \int_0^{\sigma_i(\mathbf{B})} \left(1 - \frac{u}{\gamma}\right)_+ du \\ &= \sum_{i=1}^{s} \phi_{1,\gamma}(\sigma_i(\mathbf{B})) = M_\gamma(\sigma(\mathbf{B})),\ \gamma > 1 \end{aligned} \tag{2.57}$$

where $\sigma(\mathbf{B}) = (\sigma_1(\mathbf{B}), \ldots, \sigma_s(\mathbf{B}))^\top$ denotes a function from $\mathbb{R}^{p\times q}$ to $\mathbb{R}^s_{+,s} = \min(p, q)$. The matrix γ-norm is non-convex with respect to $\mathbf{B}$.

Furthermore, the dictionary is jointly learned from both auxiliary and target domains, in order to transfer useful knowledge from the auxiliary domain. As the source data set usually contains much more samples than target data set $\mathbf{X}_T$, the learning of dictionary is easily dominated by the source data. To emphasize the reconstruction power of $\mathbf{D}$ in the target domain, the authors propose to introduce an $l_{2,1}$ norm constraint on the source coefficient matrix $\mathbf{Z}_S$. In this way, some rows in $\mathbf{Z}_S$ are encouraged to be zero, which enables $\mathbf{X}_S$ to adaptively select bases from $\mathbf{D}$.

By replacing the rank function and l_0 norm with matrix γ-norm and MCP norm, and adding the $l_{2,1}$ norm constraint on source coefficient matrix, the objective function (2.55) can then be rewritten as:

$$\begin{aligned} \min_{\mathbf{D},\mathbf{Z}_S,\mathbf{Z}_T,\mathbf{E}_S,\mathbf{E}_T} \quad & \|\mathbf{Z}_T\|_{\gamma 1} + \lambda_1 M_{\gamma 2}(\mathbf{E}_S) + \lambda_2 M_{\gamma 2}(\mathbf{E}_T) + \lambda_3 \|\mathbf{Z}_S\|_{2,1} \\ \text{s.t. } & \mathbf{X}_S = \mathbf{D}\mathbf{Z}_S + \mathbf{E}_S, \quad \mathbf{X}_T = \mathbf{D}\mathbf{Z}_T + \mathbf{E}_T \end{aligned} \tag{2.58}$$

where λ_3 is a trade-off parameter, and $\|\mathbf{Z}_S\|_{2,1} = \sum_{j=1}^{n}(\sum_{i=1}^{d}([\mathbf{Z}_S]_{ij})^2)^{1/2}$ is the $l_{2,1}$ norm. Each column in the learned coefficient matrix $\mathbf{Z}_T$ corresponds to one sample in the target domain, which is named *low-rank coding* of the corresponding sample.

The authors proposed a majorization-minimization augmented Lagrange multiplier (MM-ALM) algorithm to solve the problem (2.58). They presented two applications of this S-Low coding, one for clustering and the other for classification. Experimental results on five benchmark data sets demonstrated the effectiveness of the proposed algorithms compared with the state-of-the-art self-taught learning methods.

2.2 Classifier level knowledge transfer

Unlike the Representation level knowledge transfer methods, which focus on learning a shared representation space for source and target data, classifier level knowledge transfer focus on transferring knowledge from source classifiers. The objective of these kind of methods is usually selecting and adapting some learned classifiers to a new classification task which has only few labeled samples.

2.2.1 SVM based methods

Support Vector Machine (SVM) is a supervised learning method for solving classification and regression problems, and several early works on classifier level knowledge transfer are constructed based on the original SVM classifier. A common form of the objective function of these SVM based transfer learning models could be expressed as follows:

$$\min_{\mathbf{w}^t, b} \Phi(\mathbf{w}^t) + C \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_T} \varepsilon(\mathbf{x}_i, y_i; \mathbf{w}^t, b) \tag{2.59}$$

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 domain $\mathcal{D}_T$, 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 [Yang *et al.* 2007], in which Yang *et al.* assume that the decision function $f_T(\cdot)$ for the target classification task can be formulated as:

$$f_T(\mathbf{x}) = f_S(\mathbf{x}) + \Delta f(\mathbf{x}) \tag{2.60}$$

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 using the labeled data from the target domain $\mathcal{D}_T$ and the pre-learned parameters for the source decision function $f_S(\cdot)$, the objective function is as follows:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } \xi_i \geqslant 0,\ y_i f_S(\mathbf{x}_i) + y_i \mathbf{w}^\top \phi(\mathbf{x}_i) \geqslant 1 - \xi_i,\ \forall(\mathbf{x}_i, y_i) \in \mathcal{D}_T \tag{2.61}$$

where $\sum_i \xi_i$ measures the total classification error of the adapted classifier $f_T(\cdot)$ in the target domain. The first term in (2.61) minimizes the deviation between the target decision boundary and the source decision boundary. The cost factor C balances the contribution between the source classifier and the training examples, *i.e.* the larger C is, the smaller the influence of the source classifier is.

This work was improved in [Aytar & Zisserman 2011] 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 [Li 2007]) as follows:

$$\min_{\mathbf{w}} \|\mathbf{w}^t - \Gamma\mathbf{w}^s\|^2 + C\sum_{i=1}^{N}\xi_i \tag{2.62}$$

where $\mathbf{w}^s$ and $\mathbf{w}^t$ are the parameters for source classifier and target classifier respectively. Γ controls the amount of transfer regularization. The authors have shown that Γ is actually a trade-off parameter 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, [Aytar & Zisserman 2011] propose the projective Model Transfer SVM (PMT-SVM), in which they can increase the amount of transfer without penalizing margin maximization. The objective function for PMT-SVM is as follows:

$$\min_{\mathbf{w}^t} \|\mathbf{w}^t\|^2 + \Gamma\|P\mathbf{w}^t\|^2 + C\sum_{i=1}^{N}\xi_i, \quad \text{s.t. } (\mathbf{w}^t)^\top\mathbf{w}^s \geqslant 0 \tag{2.63}$$

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 transfer regularization, and C controls the weight of the loss function $\sum_i \xi_i$.

$\|P\mathbf{w}^t\|^2 = \|\mathbf{w}^t\|^2 \sin^2\theta$ 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 [Aytar & Zisserman 2011] 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 [Jiang *et al.* 2008] the A-SVM was improved for visual concept classification, where the authors propose the cross-domain SVM (CD-SVM) which makes use of k-nearest neighbors from the target domain to define a weight for each auxiliary pattern, and then the SVM classifier was trained with re-weighted patterns.

Tommasi *et al.* [Tommasi *et al.* 2010] 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:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w} - \sum_{j=1}^{k} \beta_j \mathbf{w}_j\|^2 + \frac{C}{2}\sum_{i=1}^{N} \zeta_i (y_i - \mathbf{w}\cdot\phi(\mathbf{x}_i) - b)^2 \tag{2.64}$$

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. (2.64) is the least square loss for training samples in the target domain. The optimal solution of Eq. (2.64) is as follows:

$$\mathbf{w} = \sum_{j=1}^{k} \beta_j \mathbf{w}'_j + \sum_{i=1}^{N} \alpha_i \phi(\mathbf{x}_i) \tag{2.65}$$

where $\mathbf{w}$ is expressed as a weighted sum of the pre-trained models scaled by the parameters β_j, plus the new model built on the incoming 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 is an unbiased estimator of the classifier generalization error and can be used for

model selection [Cawley 2006].

In [Kuzborskij *et al.* 2013] 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 is a small training set composed from samples belonging to the N known classes and a new 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) performance on the target $(N+1)$-th class improves by transferring from the source models, and (2) performance on the source N classes should not deteriorate or even improve compared to the former.

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. Negative transfer is prevented by weighting the amount of transfer of each source model using the coefficient vector $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_N]^\top$. To achieve the second objective, they enforce the new hyperplanes $\mathbf{W}$ to remain close to the source hyperplanes $\mathbf{W}'$ using the term $\|\mathbf{W} - \mathbf{W}'\|_F^2$. The final objective function with the two regularizers is as follows:

$$\min_{\mathbf{W},\mathbf{w}_{N+1},\mathbf{b}} \frac{1}{2}\|\mathbf{W}-\mathbf{W}'\|_F^2+\frac{1}{2}\|\mathbf{w}_{N+1}-\mathbf{W}'\boldsymbol{\beta}\|_F^2+\frac{C}{2}\sum_{i=1}^{M}\sum_{n=1}^{N+1}(\mathbf{W}_n^\top\mathbf{x}_i+b_n-Y_{i,n})^2 \quad (2.66)$$

where $\mathbf{Y}$ is the label matrix, $Y_{i,n}$ is equal to 1 if $y_i = n$ and is equal to -1 otherwise. The solution to this problem is given by:

$$\begin{aligned}
\mathbf{W}_n &= \mathbf{W}'_n + \sum_{i=1}^{M} A_{i,n}\mathbf{x}_i, \;\; n = 1, \ldots, N \\
\mathbf{w}_{N+1} &= \sum_{n=1}^{N} \beta_n \mathbf{W}'_n + \sum_{i=1}^{M} A_{i,(N+1)}\mathbf{x}_i \\
\mathbf{b} &= \mathbf{b}' - \begin{bmatrix}\mathbf{b}'' & \mathbf{b}''^\top & \boldsymbol{\beta}\end{bmatrix}
\end{aligned} \quad (2.67)$$

where

$$\mathbf{A} = \mathbf{A}' - [\mathbf{A}'' \;\; \mathbf{A}''\boldsymbol{\beta}]$$

,

$$\begin{bmatrix} \mathbf{A}' \\ \mathbf{b}'^{\top} \end{bmatrix} = \mathbf{M} \begin{bmatrix} \mathbf{Y} \\ \mathbf{0} \end{bmatrix}$$

,

$$\begin{bmatrix} \mathbf{A}'' \\ \mathbf{b}''^{\top} \end{bmatrix} = \mathbf{M} \begin{bmatrix} \mathbf{X}^{\top}\mathbf{W}' \\ \mathbf{0} \end{bmatrix}$$

,

$$\mathbf{M} = \begin{bmatrix} \mathbf{X}^{\top}\mathbf{X} + \frac{1}{C}\mathbf{I} & \mathbf{1} \\ \mathbf{1}^{\top} & 0 \end{bmatrix}^{-1}$$

To tune the parameter $\boldsymbol{\beta}$, they extend the method shown in [Tommasi *et al.* 2010]. They cast the optimization of $\boldsymbol{\beta}$ as the minimization of a convex upper bound of the LOO error and they propose a projected subgradient descent algorithm to find the optimal $\boldsymbol{\beta}$.

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 [Kuzborskij *et al.* 2013], and the negative transfer is mainly prevented by tuning the parameter $\boldsymbol{\beta}$ (which could be seen as a model selection process).

2.2.2 Boosting based methods

Adaptive boosting (AdaBoost) [Freund & Schapire 1997] is a widely used boosting algorithm. It can be used in conjunction with many other types of learning algorithms to improve performance. The basic learning algorithm, also known as 'weak learner', is trained in each iteration with reweighted training instances and the final output of AdaBoost is a weighted combination of the weak learners trained in each iteration. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. The ability of turning 'weak learners' to a 'strong learner' makes AdaBoost a natural choice

for transfer learning. The models trained on source data are always 'weak' for target data due to the distribution discrepancy between source and target, therefore one can make a combination of these 'weak models' with AdaBoost to make the combination a 'strong' model on target data.

One of the first works using AdaBoost for transfer learning is the Transfer AdaBoost(TrAdaBoost) [Dai *et al.* 2007]. Dai et *al.* 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 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 instances with large weights will intent to help the learning algorithm to train better classifiers.

Since the original TrAdaBoost only works for one source domain, in [Yao & Doretto 2010] the authors extend the TrAdaBoost to transfer learning from multiple source domains. A new algorithm MultiSource-TrAdaBoost is proposed as a direct extension of TrAdaBoost. 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 the imposition to transfer knowledge from a single source, which is potentially loosely

related to the target.

The TrAdaBoost and the MultiSource-TrAdaBoost are all instance based transfer learning algorithms, *i.e.* they try to identify which training instances, coming from the source domains, can be reused, together with the target training instances, to boost the target classifier. In [Yao & Doretto 2010] the authors further propose another parameter based transfer learning algorithm, namely TaskTrAdaBoost, in which the target classifier model shares some parameters with the most closely related sources. The TaskTrAdaBoost tries to identify which parameters, coming from various sources, can be reused, together with the target training data, to improve the target classifier learning. More precisely, TaskTrAdaBoost is a two phase learning approach. Phase-I deploys traditional machine learning to extract suitable parameters that summarize the knowledge from the sources. In this case they learn a set of weak classifiers on each source task with AdaBoost. Phase-II is a parameter-transfer approach for boosting the target classifier. It is again an AdaBoost loop over the target training data. At each iteration, they pick from the set of weak classifiers learned in Phase-I the one with the lowest classification error on the target training data. As all weak classifiers could be pre-learned in Phase-I, the phase-II transfer learning approach could be much faster than the previously introduced MultiSource-TrAdaBoost algorithm.

Theoretical analysis and experimental results in [Yao & Doretto 2010] have shown the advantage of MultiSourceTrAdaBoost and TaskTrAdaBoost over the original TrAdaBoost. Since the TaskTrAdaBoost limits the freedom in picking the weak classifiers, which leads to a smaller VC-dimension of the candidate hypothesis space, the prediction error in individual iteration would be greater for TaskTrAdaBoost, while the generalization error could be reduced, because this effect also avoids over-fitting. Therefore, TaskTrAdaBoost works better when having a large number of source tasks and a small number of target training samples. Furthermore, the convergence rate of TaskTrAdaBoost also have a reduced upper bound, which means it requires fewer iterations to converge, and therefore is more efficient than MultiSourceTrAdaBoost.

Another boosting based transfer learning algorithm is the Cross-Category Trans-

fer Learning (CCTL) proposed in [Qi *et al.* 2011]. Instead of directly transferring knowledge from source instances or pre-learned source task parameters, the authors propose to use a label propagation approach to transform source classifier predictions to target task. Specifically, they define a real-valued transfer function $T_S(\mathbf{x}, \mathbf{x}_{l,i}) = \phi_S(\mathbf{x}, \mathbf{x}_{l,i})k(\mathbf{x}, \mathbf{x}_{l,i})$ to connect the l-th source domain and the target category. 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 domain $\mathcal{D}_{S,l}$ to the target category to form a discriminant function $h_l(\mathbf{x})$ as follows:

$$h_l(\mathbf{x}) = \frac{1}{|\mathcal{D}_{S,l}|} \sum_{\mathbf{x}_{l,i} \in \mathcal{D}_{S,l}} y_{l,i} T_S(\mathbf{x}, \mathbf{x}_{l,i}) \tag{2.68}$$

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

$$S^* = \arg\min_S \Omega_l(S) \tag{2.69}$$

where

$$\Omega_l(S) = \sum_{i=1}^{N} \mathbf{w}_i (1 - y_i h_l(\mathbf{x}_i))_+ + \frac{\lambda}{2} \|S\|_F^2 \tag{2.70}$$

where $(\cdot)_+ = max(0, \cdot)$, $\|S\|_F$ is the Frobenius norm of the matrix S, λ is the balancing parameter, and $\mathbf{w}_i$ is the sample weight for i-th sample in target domain.

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 TaskTrAdaBoost. However, CCTL only works for binary classification problems. Further-

more, 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.

Table 2.1 summarizes the boosting based transfer learning methods introduced in this section compared with the original AdaBoost algorithm.

Table 2.1: Boosting based transfer learning 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
TaskTrAdaBoost	Wrongly classified samples ↑ Correctly classified samples ↓	The one with best performance on target from pre-learned weak classifiers
CCTL	Wrongly classified samples ↑ Correctly classified samples ↓	The one with best performance on target from candidate cross- category classifiers

2.2.3 Generative models

The two groups of methods introduced previously are all discriminative models which learn the conditional distribution of labels on knowing input features. 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 [Fei-Fei *et al.* 2006], which is a Bayesian-based unsupervised one-shot learning object categorization framework that learns a new object category using a single example (or just a few).

Firstly, to formalize the task of object classification with a generative model, they start with a learned object class model and its corresponding model distribution $p(\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a set of model parameters for the distribution. Given a new image and the objective is to decide if it contains an instance of the target object class or not. In this query image they have identified N interesting features with locations $\mathcal{X}$, and appearances $\mathcal{A}$. They now make a Bayesian decision R. For clarity, they express training images through the detected feature locations $\mathcal{X}_t$ and appearances $\mathcal{A}_t$.

$$\begin{aligned} R &= \frac{p(\text{Object}|\mathcal{X}, \mathcal{A}, \mathcal{X}_t, \mathcal{A}_t)}{p(\text{No Object}|\mathcal{X}, \mathcal{A}, \mathcal{X}_t, \mathcal{A}_t)} \\ &= \frac{p(\mathcal{X}, \mathcal{A}|\mathcal{X}_t, \mathcal{A}_t, \text{Object})p(\text{Object})}{p(\mathcal{X}, \mathcal{A}|\mathcal{X}_t, \mathcal{A}_t, \text{No Object})p(\text{No object})} \\ &\approx \frac{\int p(\mathcal{X}, \mathcal{A}|\boldsymbol{\theta}, \text{Object})p(\boldsymbol{\theta}|\mathcal{X}_t, \mathcal{A}_t, \text{Object})d\boldsymbol{\theta}}{\int p(\mathcal{X}, \mathcal{A}|\boldsymbol{\theta}_{bg}, \text{No object})p(\boldsymbol{\theta}_{bg}|\mathcal{X}_t, \mathcal{A}_t, \text{No object})d\boldsymbol{\theta}_{bg}} \end{aligned} \tag{2.71}$$

Note that the ratio of $\frac{p(\text{Object})}{p(Noobject)}$ in the second line is usually set manually to 1, hence it is omitted in the third line of Equation (2.71). The goal of learning in this formulation is to estimate the density of the object model $p(\boldsymbol{\theta}|\mathcal{X}_t, \mathcal{A}_t, \text{Object})$. In other words, in the high dimensional space that characterize the objects, the goal is to find the appropriate distribution that defines the extent of where and how the models occupy this space. This goal is achieved through the usage of prior knowledge.

The representation of the object class model is chosen based on the *constellation model* [Burl & Perona 1996]. A constellation model consists of a number of parts, each encoding information on both the shape and the appearance. The appearance of each part is modeled and the shape of the object is represented by the mutual position of the parts [Fergus *et al.* 2003]. The entire model is generative and probabilistic, so appearance and shape are all modeled by probability density functions,

which are Gaussians. Assume a generative object model is learned, with P parts and a posterior distribution on the parameters $\boldsymbol{\theta} : p(\boldsymbol{\theta}|\mathcal{X}_t, \mathcal{A}_t)$ where $\mathcal{X}_t$ and $\mathcal{A}_t$ are the location and appearances of interesting features found in the training data. Recall the Bayesian decision rule in Equation (2.71). Assume that all non-object images can also be modeled by a background with a single set of parameters $\boldsymbol{\theta}_{bg}$ which are fixed. The ratio of the priors may be estimated from the training set or set manually (usually to 1). The decision then requires the calculation of the ratio of the two likelihood functions, which may be factored as follows:

$$p(\mathcal{X}, \mathcal{A}|\boldsymbol{\theta}) = \sum_{\mathbf{h}\in H} p(\mathcal{X}, \mathcal{A}, \mathbf{h}|\boldsymbol{\theta}) = \sum_{\mathbf{h}\in H} \underbrace{p(\mathcal{A}|\mathbf{h}, \boldsymbol{\theta})}_{\text{Appearance}} \underbrace{p(\mathcal{X}|\mathbf{h}, \boldsymbol{\theta})}_{\text{Shape}} \tag{2.72}$$

Since the model only has P (typically 3-7) parts while there are N (up to 100) features in the image, the authors introduce an indexing variable $\mathbf{h}$ which they call a *hypothesis* which allocates each image feature either to an object or to the background.

The task in learning is to estimate the density $p(\boldsymbol{\theta}|\mathcal{X}_t, \mathcal{A}_t)$. This is done using the Variational Bayes Procedure. It approximates the posterior distribution $p(\boldsymbol{\theta}|\mathcal{X}_t, \mathcal{A}_t)$ by $q(\boldsymbol{\theta}, \boldsymbol{\omega}, \mathbf{h})$. ω is the mixture component label and $\mathbf{h}$ is the hypothesis. Using Bayes' rule: $q(\boldsymbol{\theta}, \boldsymbol{\omega}, \mathbf{h}) \approx p(\boldsymbol{\theta}|\mathcal{X}_t, \mathcal{A}_t) \propto p(\mathcal{X}_t, \mathcal{A}_t|\boldsymbol{\theta})p(\boldsymbol{\theta})$. The likelihood terms use Gaussian densities and by assuming priors of a conjugate form, int his case a Normal-Wishart, the posterior q-function is also a Normal-Wishart density. The variational Bayes procedure is a variant of EM which iteratively updates the hyper-parameters and latent variables to monotonically reduce the Kullback-Liebler distance between $p(\boldsymbol{\theta}|\mathcal{X}_t, \mathcal{A}_t)$ and $q(\boldsymbol{\theta}, \boldsymbol{\omega}, \mathbf{h})$. Using this approach one is allowed to incorporate prior information in a systematic way. There are two stages to learning: an E-step where the responsibilities of the hidden variables are calculated and an M-step to update the hyper-parameters of $q(\boldsymbol{\theta}, \boldsymbol{\omega}, \mathbf{h})$. This learning approach could be done in an incremental way when doing knowledge transfer. Assume having a model with hyper-parameters $\boldsymbol{\Theta}$, estimated using M previous images, and having N new images for updating the model. From the M previous images, one have retained sufficient statistics for each mixture component ω. Then compute the re-

sponsibilities and the sufficient statistics for the new images. In the incremental M-step combine the sufficient statistics from these new images with the existing set of sufficient statistics from the previous images, then compute the overall sufficient statistics. From these one can update the model hyper-parameters. When the model converges, the final value of the sufficient statistics from the new images are combined with the existing set.

This approach allows one to incorporate priors from unrelated object categories. For example, when learning the category motorbikes, priors can be obtained by averaging the learned model parameters from other three categories spotted cats, faces and airplanes.

Another work is [Yu & Aloimonos 2010], where the authors propose an attribute-based transfer learning framework for zero-shot and one-shot learning. They firstly build a generative attribute model to learn the probabilistic distributions of image features for each attribute, which they consider as attribute priors. These attribute priors can then be used to (1) classify unseen images of target categories (for zero-shot learning), or (2) facilitate learning classifiers for target categories when there is only one training example per target category (for one-shot learning).

Specifically, the category-attribute relationship is represented by a category-attribute matrix $\mathcal{M}$, where the entry at the m-th row and the l-th column is a binary value indicating whether category m has the l-th attribute. Each object category thus has a list of attributes whose corresponding values in $\mathcal{M}$ equal to "yes". Given an object category, the list of associated attributes $\mathbf{a}$ is deterministic. For example, for category *cow*, the attribute list is $\mathbf{a} = \{black, white, brown, spots, furry, domestic\}$. This information is supposed to be available for both source categories and target categories.

The attribute model and the target classifier belong to an extension of topic models. In a topic model, a document $\mathbf{w}$ is modeled by a mixture of topics, and each topic z is represented by a probability distribution of words (noted as w). In computer vision, a quantized image feature is often analogous to a word, a group of co-occurred image features to a topic, and an image to a document. The authors

choose the bag-of-features as image representation, *i.e.* the spatial information of image features is discarded, and an image is represented as a collection of orderless visual words.

They employ the Author-Topic (AT) model as the attribute model. The AT model is a generative model which is originally designed to model the interests of authors from a given document corpus. They extend the AT model to describe the distribution of images features related to attributes. An image j has a list of attributes, denoted by $\mathbf{a}_j$. An attribute l in $\mathbf{a}_j$ is modeled by a discrete distribution of K topics, which parameterized by a K-dim vector $\boldsymbol{\theta}_l = [\theta_{l1}, \ldots, \theta_{lK}]$ with topic k receiving weight θ_{lk}. The topic k is modeled by a discrete distribution of W code-words in the lexicon, which is parameterized by a W-dim vector $\boldsymbol{\phi}_k = [\phi_{k1}, \ldots, \phi_{kW}]$ with code word v receiving weight ϕ_{kv}. Symmetric Dirichlet priors are placed on θ and ϕ, with $\theta_l \sim \text{Dirichlet}(\alpha)$, and $\phi_k \sim \text{Dirichlet}(\lambda)$, where α and λ are hyper-parameters that affect the sparsity of these distributions. Given an attribute list $\mathbf{a}_j$ and a desired number N_j of visual words in image j. To generate each visual word, they firstly condition on $\mathbf{a}_j$, choose an attribute $x_{ji} \sim \text{Uniform}(\mathbf{a}_j)$; then they condition on x_{ji}, choose a topic $z_{ji} \sim \text{Discrete}(\theta_{x_{ji}})$, where θ_l defines the distribution of topics for attribute $x = l$; finally, they condition on z_{ji} and choose a visual word $w_{ji} sim \text{Discrete}(\phi_{z_{ji}})$, where ϕ_k defines the distribution of visual words for topic $z = k$.

Given a training corpus, the goal of inference in an AT model is to identify the values of ϕ and θ. To achieve this goal, the authors make use of a collapsed block Gibbs sampling method [Rosen-Zvi *et al.* 2010]. Here "collapse" means that the parameters ϕ and θ are analytically integrated out, and the "block" means that the pair of (x_{ji}, z_{ji}) are drawn together. To run the Gibbs sampling algorithm, they firstly initialize $\mathbf{x}$ and $\mathbf{z}$ with random assignments. In each Gibbs sampling iteration, they draw samples of x_{ji} and z_{ji} for all visual words in the training corpus in a randomly permuted order of i and j. The samples of $\mathbf{x}$ and $\mathbf{z}$ are recorded after the burn-in period. According to experimental results, 200 iterations are sufficient for the sampler to be stable. The posterior means of θ and ϕ can then be estimated using the recorded samples.

If there is only one attribute in each image and the attribute is the object category label, the AT model can be used in object categorization problems, they call this approach Category-Topic (CT) model and use it as the target classifier. (It is also possible to employ other types of target classifier, for example, the authors also evaluate SVM as a target classifier. Although experiment show that the CT model outperforms SVM by a large margin.) After learning a CT model, it can be used to classify a test image by choosing the target classifier that yields the highest likelihood.

If the attribute list is unique in each category, an AT model can also be used to classify a new image by the maximum likelihood criterion in the case of zero-shot learning problem. An approximate likelihood could be calculated with a pseudo weight for the category-specified topic distribution of a new category, this pseudo weight can be viewed as a prior before seeing the real training examples of the new category.

To deal with the one-shot learning, the main problem is that the number of training samples in source is higher than the one of target by several orders, therefore one need to control the balance between the prior information from source categories and the new information in target categories. The authors propose two approaches to achieve this goal.

The first one is knowledge transfer by synthesis of training samples. Firstly, they learn the attribute model from the training samples of the source categories; then, for each target category, they run the generative process to produce S synthesized training samples using the estimated $\hat{\theta}$ and $\hat{\phi}$ as well as the attribute list associated to this target category. The number of visual words for each generated training sample is chosen as the average number of visual words per image in the source categories. In this procedure, the number of synthesized training samples S represent the confidence about the attribute priors. It can be used to adjust the balance between the attribute priors and new observations from the training images of target categories.

The second approach is to give parameters of the CT model in the target classifiers informative priors. These priors could be estimated with the source data, the

scaling factors of these priors represent the confidence on them, which can be used to control the balance between attribute priors and the new observations.

2.3 Summary

In this chapter we have shown two groups of related research works on feature representation level knowledge transfer and on classifier level knowledge transfer. Actually, the two groups of methods correspond to techniques in two importance steps for solving vision recognition problems, they are not conflicting but complementary. Therefore, more and more recent transfer learning works are combining the two together to give a comprehensive framework for knowledge transfer.

For example, in [Kuzborskij *et al.* 2015] the authors propose a scalable greedy subset selection algorithm which selects relevant source hypotheses and feature dimensions simultaneously. The feature level knowledge transfer is done by feature dimension selection and the classifier level knowledge transfer is done by source hypotheses selection. In [Long *et al.* 2016] the authors proposed an end-to-end deep convolutional neural network for domain adaptation, which is called the *Residual Transfer Network* (RTN). In the first layers of RTN they try to learn a shared feature space for source and target by minimizing the criteria MMD between the two feature distributions. In the last layers they add some extra residual blocks on the source classification network to adjust the difference between parameters of source classifier and that of target classifier (*i.e.* to minimize the conditional distribution discrepancy between source and target).

As most works in the transfer learning domain have specific assumptions, we give a summary of the common transfer learning scenarios with their assumptions in table 2.2. The scenario we consider in our contributions is the Inductive Transfer Learning scenario.

The discriminative transfer learning (DTL) algorithm we propose in chapter 3 falls under the boosting based methods which is introduced in section 2.2.2. Unlike previous works, in the proposed DTL the weak learners are not simply the learned source classifiers, but rather discriminative sparse representation based classifiers

Table 2.2: Some common transfer learning scenarios and their corresponding related works appeared in this chapter. The detailed explanation of the notations appeared in columns 'Assumption' and 'Objective' could be found in section 1.1.2.

TL scenarios	Source Data	Target Data	Assumption	Objective	Related works
Multi-task learning	labeled	labeled	$\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$ $\mathcal{T}_S \neq \mathcal{T}_T$	learn $f_S(\cdot)$, $f_T(\cdot)$	[Ando & Zhang 2005] [Argyriou *et al.* 2007] [Amit *et al.* 2007] [Parameswaran & Weinberger 2010]
Domain Adaptation	labeled	unlabeled	$\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$ $\mathcal{Y}_S = \mathcal{Y}_T$	learn $f_T(\cdot)$	[Pan *et al.* 2008] [Pan *et al.* 2011] [Satpal & Sarawagi 2007] [Long *et al.* 2013b] [Long *et al.* 2015] [Long *et al.* 2016] [Fernando *et al.* 2013] [Sun & Saenko 2015] [Zhang *et al.* 2017] [Tzeng *et al.* 2014] [Tzeng *et al.* 2015] [Ganin & Lempitsky 2015]
Self-taught Learning	unlabeled	labeled	$\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$	learn $f_T(\cdot)$	[Quattoni *et al.* 2007] [Raina *et al.* 2007] [Wang *et al.* 2013] [Li *et al.* 2017]
Inductive Transfer learning	labeled	labeled	$\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$ $\mathcal{Y}_S \neq \mathcal{Y}_T$	learn $f_T(\cdot)$	[Thrun 1996] [Si *et al.* 2010] [Perrot & Habrard 2015] [Aytar & Zisserman 2011] [Jiang *et al.* 2008] [Tommasi *et al.* 2010] [Kuzborskij *et al.* 2013] [Kuzborskij *et al.* 2015] [Dai *et al.* 2007] [Yao & Doretto 2010] [Qi *et al.* 2011]
One-shot learning	labeled	single labeled	$\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$ $\mathcal{Y}_S \neq \mathcal{Y}_T$	learn $f_T(\cdot)$	[Fink 2005] [Fei-Fei *et al.* 2006] [Yu & Aloimonos 2010]
Zero-shot learning	labeled	No data	$\mathcal{X}_S \cap \mathcal{X}_T \neq \emptyset$ $\mathcal{Y}_S \neq \mathcal{Y}_T$	learn $f_T(\cdot)$	[Yu & Aloimonos 2010]

for target task which are formed with source samples. In this way we take into account explicitly the similarities and dissimilarities by using the source categories which are most positively related to target task and also the source categories which are most negatively related to target task. Furthermore, we propose to use

two parallel AdaBoost simultaneously to handle the unbalanced data distribution problem. The resulting DTL algorithm achieves state-of-the-art performance on two different scene classification datasets compared to existing transfer learning algorithms.

The Joint Transfer Learning Network (JTLN) we propose in chapter 4 falls under the feature representation level knowledge transfer with Deep Neural Networks approach which is introduced in section 2.1.4. Unlike existing works, which are mostly proposed for domain adaptation scenario, JTLN is proposed for a more general cross category transfer learning scenario in which we don't assume source domain and target domain share the same label space. To our best knowledge, this is the first work based on DNN for cross category transfer learning scenario. We perform a joint feature learning for both source and target data based on a Convolutional Neural Network. By adding an Optimal Transport loss between source and target classifier predictions as a constraint on the source classifier, the proposed Joint Transfer Learning Network can effectively learn useful knowledge for target classification from source data. Furthermore, by using different kind of metric as cost matrix for the OT loss, JTLN can incorporate different prior knowledge about the relatedness between target categories and source categories.

Chapter 3

Discriminative Transfer Learning using Similarities and Dissimilarities

3.1 Introduction

When learning a classification model for a new concept category with only a small number of training samples, brute force application of traditional machine learning algorithms generally leads to over-fitted classifiers with poor generalization skills. On the other hand, collecting a sufficient number of manually labeled training samples may prove very expensive, especially in the case of complex high-level visual concepts with large intra-class variations. In recent years, we have witnessed an increasing interest in transfer learning (TL) algorithms as solutions for this kind of problem [Pan & Yang 2010a]. The basic idea is that different categories are not always independent from each other. It is thus possible to leverage abundant labeled learning data, which may be available for those closely correlated classes or categories to improve the performance of the target learning task through knowledge transfer [Qi *et al.* 2011] [Dai *et al.* 2007]. For example, in scene image classification, the categories 'waterfall' and 'water' are correlated since they both contain the element 'water' in the images. We can thus use the knowledge of one category to help learn the other category.

A common assumption in transfer learning is that the effectiveness of knowledge transfer is greatly affected by the relationship between source and target [Dai *et al.* 2007]. The closer the relationship, the more transferred knowledge can

be expected to help improve learning of the target classifier. On the other hand, brute force transfer of knowledge learned from a source poorly related to a target may reduce the performance of the target classifier, resulting in so-called *negative transfer* [Pan & Yang 2010a]. In contrast, in this work we consider not only source tasks positively correlated to the target task, but also source tasks negatively correlated to the latter, as useful knowledge to help learning of the target task. Our intuition is that, when an input sample is closer to a source category negatively correlated to the target category, this is also a useful hint that should be taken into account to reduce the probability of the input sample belonging to the target category. The importance of similarity and dissimilarity in human visual categorization has also been outlined by psychophysicists [Stewart & Brown 2005] recently. In a preliminary work[Lu *et al.* 2014], we carried out a study on the feasibility of making use of both similarities and dissimilarities for knowledge transfer, and proposed SparseTL, a simple yet meaningful method, which highlights the interest of exploring data relatedness in terms of both similarities and dissimilarities. However, since SparseTL only explored a small amount of knowledge concerning source domain data, its performance gain with the state-of-the-art was not that evident.

In this chapter, we propose a novel discriminative knowledge transfer method, which leverages relatedness of various source categories with the target category to enhance learning of the target classifier. DTL explicitly makes use of both positively and negatively correlated source categories to help classification of the target category. Specifically, DTL chooses from source categories the most positively and negatively correlated categories to act as positive and negative dictionaries for discriminative classification of target samples using reconstruction residuals on these dictionaries. We further enhance the performance of DTL by concurrently running 2 AdaBoost processes on the positive and negative distributions of the target training set. A novel Wilcoxon-Mann-Whitney based cost function is also introduced in DTL to deal with the unbalanced nature of data distribution.

The main contributions of this work are fourfold:

1. We highlight the importance of learning both similarity and dissimilarity in

a transfer learning algorithm through joint use of positively correlated and negatively correlated source categories, and introduce a Bi-SRC classifier as the building block of the proposed DTL;

2. We propose a novel cost function based on the Wilcoxon-Mann-Whitney (WMW) statistic, and apply two parallel boosting processes, both on positive data and on negative data distribution, thus successfully avoiding the effect of unbalanced data distribution.

3. We conduct theoretical analyses on the proposed DTL algorithm and provide theoretical guarantees both in terms of error bound and time complexity.

4. Using different features and evaluating on two different performance metrics, we benchmark the proposed DTL on two different databases for the task of image categorization. We also consistently demonstrate the effectiveness of the proposed DTL: it displays the best performance with a large margin in comparison to several state-of-the-art TL methods, with a runtime that can prove 80 times faster in training and 66 times faster in testing than the other state-of-the-art TL methods that it has been compared with.

The rest of this chapter is organized as follows. Section 3.2 introduces some related works and describes their connection to the proposed work. Section 3.3 introduces the proposed DTL algorithm. Section 3.4 conducts a theoretical analysis of the proposed DTL algorithm, both in terms of error bound and time complexity. Section 4.4 presents and discusses the experimental results on natural scene classification in comparison with state-of-the-art transfer learning methods. Finally, Section 3.6 concludes our work and gives some future directions.

3.2 Related work

3.2.1 Transfer Learning

Current research on transfer learning (TL) in computer vision has featured three main approaches: *Unsupervised transfer learning* focuses on mak-

ing use of vast amounts of unlabeled data to help the target classification task [Bengio 2011] [Ding *et al.* 2015] [Long *et al.* 2015]. *Domain adaptation* [Si *et al.* 2010] [Long *et al.* 2016] aims to leverage knowledge of different source domains from the same concept category in order to narrow down the distribution difference between source data and target data, thereby enhancing learning of the target task. *Inductive transfer learning* [Raina *et al.* 2007] [Qi *et al.* 2011] [Yao & Doretto 2010] [Li *et al.* 2017] makes use of source domains from different concept categories to help the target task. In inductive transfer learning approaches, target training samples are often well labeled, while source domain samples may either be labeled or unlabeled.

A research trend consists of *multi-source transfer learning* [Shao *et al.* 2014a] approaches, which are also known as model selection methods for knowledge transfer. The assumption is that, since more than one source domain is available, the most useful need to be chosen, as they can potentially improve the target learner. Different approaches are proposed to address this problem: The *SVM-based methods* [Yang *et al.* 2007] [Duan *et al.* 2009] propose different domain adaptation methods to adapt the SVM classifier learned on the source domain to the target domain classifier. The *boosting-based methods*, *e.g.*, [Yao & Doretto 2010], extend the TrAdaBoost [Dai *et al.* 2007] method to the multi-source transfer learning situation by selecting one of the most relevant source domains to learn the weak classifier at each iteration. *Multi-kernel learning*, *e.g.*, [Jie *et al.* 2011], proposes a multiple kernel transfer learning (MKTL) method, which learns the best hyperplanes and corresponding weights assigned to each prior model in a unified optimization process.

The proposed DTL method can be considered to be a boosting-based method. Different from the existing boosting-based methods listed in [Shao *et al.* 2014a], the proposed DTL method further exploits the possibility of knowledge transfer from negatively correlated source data, thus improving the knowledge transfer rate of the source domain.

3.2.2 Multi-task learning

Multi-task learning [Kumar & Daume 2012] [Eaton & Ruvolo 2013] [Kong *et al.* 2017] is a similar research area, in which multiple related prediction tasks are learned jointly, thus sharing information across the tasks. For example, the GOMTL method [Kumar & Daume 2012] assumes that each task parameter vector is a linear combination of a finite number of underlying basis tasks. GOMTL automatically learns overlapping groups of tasks based on the assumption that task parameters within a group lie in a low dimensional subspace. However, it allows tasks in different groups to overlap with each other in one or more bases. ELLA [Eaton & Ruvolo 2013] employs this rich model of underlying task structure and applies it to online multi-task learning in the lifelong learning setting. Specifically, ELLA maintains a sparsely shared basis for all task models, transfers knowledge from the basis to learn each new task, and refines the basis over time to maximize performance across all tasks. Although in transfer learning we also use source tasks related to the target task, the goal of transfer learning is different from that of multi-task or lifelong learning. Rather than seeking good performance on all source and target tasks, transfer learning is most concerned with the target task. In transfer learning, the roles of the source and target tasks are not symmetric as in multi-task learning.

3.2.3 Sparse representation

While sparse representation and low-rank methods have been studied for many years, they have only recently become popular in the domain of computer vision. The first application was [Wright *et al.* 2009], which applies sparse representation to human face recognition. Some recent advances are [Yang *et al.* 2011] [Li *et al.* 2014] [Shao *et al.* 2014c].

These methods have also been applied to transfer learning, *e.g.*, [Maurer *et al.* 2013] [Long *et al.* 2013a] [Wang & Bensmail 2013] [Al-Shedivat *et al.* 2014] [Gong *et al.* 2012] [Long *et al.* 2013b] [Shao *et al.* 2014b]. Quattoni *et al.* [Quattoni *et al.* 2008] propose transferring knowledge from related

source categories to a target category by learning a sparse prototype image representation from a large set of unlabeled data. Jhuo *et al.* [Jhuo *et al.* 2012] apply low-rank reconstruction to the domain adaptation problem by transforming the visual samples in the source set into an intermediate representation, such that each transformed source sample can be linearly reconstructed by the samples of the target set.

In these works, [Quattoni *et al.* 2008] focuses on the semi-supervised classification problem, which makes use of both the labeled source categories and a large set of unlabeled data. Other works such as [Long *et al.* 2013a] [Wang & Bensmail 2013] [Gong *et al.* 2012] [Long *et al.* 2013b] and [Jhuo *et al.* 2012] mostly focus on the problem of domain adaptation, which aims at adapting the sample distribution of the source domain to the target domain. [Al-Shedivat *et al.* 2014] extends the work of [Long *et al.* 2013a] to a supervised version, which also makes use of some labeled target training samples, while [Maurer *et al.* 2013] investigates the use of sparse coding and dictionary learning in the context of multi-task and transfer learning. Both partially focus on the dictionary learning problem. In our work, we consider a supervised transfer learning problem, which attempts to search for and make use of helpful knowledge from the labeled source categories to improve the performance of the target classification task. The transfer learning problem studied here is thus different from the semi-supervised learning or domain adaptation considered in [Quattoni *et al.* 2008, Long *et al.* 2013a, Wang & Bensmail 2013, Gong *et al.* 2012, Long *et al.* 2013b, Jhuo *et al.* 2012]. It does not make use of unlabeled data as in semi-supervised learning. Moreover, unlike in domain adaptation, which assumes that the source domain and the target domain contain the same categories with different distributions, the source categories considered in our problem are different from the target categories. To solve this problem, we also use the sparse reconstruction residual as the basic discriminant as in [Wright *et al.* 2009]. We further improve its discriminative power by using the difference of two reconstruction residuals, one using a positive and the other a negative dictionary. A series of positive-negative dictionary pairs using samples both from the target domain and from the source domain is then built up to transfer knowledge.

3.3 The proposed DTL algorithm

Given a target category with only a few training samples, we propose in this section a new discriminative transfer learning algorithm which leverages the availability of related source categories with far more samples. We first state the problem in Section 3.3.1, which we tackle here. We then introduce the proposed DTL algorithm in three steps: the discriminative Bi-SRC (Binary SRC) classifier derived from SRC (in Section 3.3.2), the construction of the DTL basic classification model (in Section 3.3.3) and the overall boosting algorithm (in Section 3.3.4). Finally, we also provide a time complexity analysis of the proposed DTL algorithm in comparison to other boosting-based transfer learning algorithms in Section 3.4.2.

In this work we consider the problem of binary classification. The proposed DTL method can subsequently be extended to multi-class classification using one-vs-one, one-vs-all or ECOC (Error-Correcting Output Codes) based approaches. We use hereafter the following notations: calligraphic letters in upper case denote sets, *e.g.*, $\mathcal{T}, \mathcal{A}, \mathcal{D}, \mathcal{I}$, or functions, *e.g.*, $\mathcal{R}(\cdot)$, $\mathcal{C}(\cdot)$; bold letters in upper case denote matrices, *e.g.*, $\mathbf{D}, \mathbf{X}$; bold letters in lower case denote column vectors, *e.g.*, $\mathbf{x}, \mathbf{y}, \mathbf{d}, \mathbf{s}, \mathbf{w}$.

3.3.1 Problem statement

Assume that we have a small target training set $\mathcal{T} = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, N\}$ for a target concept category, where $\mathbf{x}_i \in \mathbb{R}^m$ and $y_i \in \{+1, -1\}$. We also have L sets of source categories with far more samples. These source categories are possibly related to, but in any case are different from the target category. In each source set we have only positive samples of the corresponding category[1]: $\mathcal{A}_l = \{(\mathbf{x}_{l,i}, \mathbf{y}_{l,i}) \mid \delta_l(\mathbf{y}_{l,i}) = +1, \delta_{j \neq l}(\mathbf{y}_{j,i}) = -1, i = 1, \ldots, N_l\}$ for $l = 1, \ldots, L$. In this case $\mathbf{x}_{l,i} \in \mathbb{R}^m$ denotes the feature vector, while $\mathbf{y}_{l,i}$ denotes the L-dimensional ground-truth label vector for the i-th sample of the l-th category. We assume that the feature vectors for the training samples and those for the source samples are all extracted from the same feature space, *e.g.*, histogram of visual words, so that each bin in these feature vectors has the same meaning for the different samples.

[1] In this case we use $\mathcal{A}$ instead of $\mathcal{S}$ to represent 'Source categories' as in [Qi *et al.* 2011] because $\mathcal{A}$ can be seen as 'Additional categories', which has the same meaning as 'Source categories'.

The goal is to build a discriminative classifier, which leverages the L sets of possibly related source categories with abundant training samples to help classification of the target category.

Construction of the DTL classifier is a two-step process: first, we build a customized Binary Sparse Representation Classifier (Bi-SRC) based knowledge transfer algorithm as the basic classification model. This algorithm first chooses the most useful from the L source categories and uses them together with the target training set to build a discriminative classification model. Second, we use the basic classification model built in step one as a weak learner and then build a 2-side boosting algorithm to further improve classification performance. In the following subsections we first introduce the customized Bi-SRC classifier, before explaining these two steps in detail.

3.3.2 The Bi-SRC classifier

The Sparse Representation based Classifier (SRC) was first introduced in [Wright *et al.* 2009] and applied to 2D face recognition. The idea behind SRC is to assume that the training samples from a single class lie on a subspace and that the test samples can be modeled as a sparse linear combination of the training samples from the same class. Based on this idea, the SRC uses all the training samples jointly to build a redundant dictionary. Then, for each test sample, it calculates the sparse reconstruction coefficients by solving a $l1$ norm minimization problem. The test sample is finally classified into the class with the smallest reconstruction residual for the test sample.

Inspired by this SRC algorithm, we derive the so-called *Bi-SRC (Binary SRC)* for the binary classification problem, *i.e.*, image scene classification, which also uses the reconstruction residual as a discriminant. However, we propose to further increase the discriminative power of this residual and make use of both similarities and dissimilarities by simply comparing their residuals. Specifically, we assume that we have at our disposal a pair of dictionaries, namely $\mathbf{D}^+$ and $\mathbf{D}^-$, where $\mathbf{D}^+$ contains similar training samples and $\mathbf{D}^-$ dissimilar training samples. The reconstruction residuals using $\mathbf{D}^+$ should thus be smaller than those using $\mathbf{D}^-$.

Let $\mathbf{D}^+ = [\mathbf{d}_1^+, \ldots, \mathbf{d}_{k_+}^+]$, $\mathbf{D}^+ \in \mathbb{R}^{m \times k_+}$ be the positive dictionary containing k_+ positively correlated training samples, and $\mathbf{D}^- = [\mathbf{d}_1^-, \ldots, \mathbf{d}_{k_-}^-]$, $\mathbf{D}^- \in \mathbb{R}^{m \times k_-}$ the negative dictionary containing k_- negatively correlated training samples.

The reconstruction coefficients $\mathbf{s}^+$ for a test sample $\mathbf{x}$ with the positive dictionary $\mathbf{D}^+$ can be defined as the solution to the following equation:

$$\mathbf{D}^+ \cdot \mathbf{s}^+ = \mathbf{x} \tag{3.1}$$

If k_+ is smaller than feature dimension m, this system of equations is overdetermined and a solution can be found by simply minimizing the square error between the right and left members. However, as shown in [Wright *et al.* 2009], this solution is usually dense and thus not informative for recognizing the test sample. To search for the best solution in this case, we observe that, due to intra-class variations and inter-class correlations, a target sample can share many similarities with only a small group of samples from a positively correlated source category. Therefore, in contrast to a negatively correlated source category, it should be possible to represent a valid test sample for a target category using only a few samples from a positively correlated category. This representation is naturally sparse compared to a representation using all samples from a positively correlated source category. This motivates us to seek a sparse solution to Eq. (3.1), which can be found by solving the following $l1$-norm minimization problem when $m \ll k_+$ [Candès *et al.* 2006]:

$$\hat{\mathbf{s}}^+ = \underset{\mathbf{s}^+ \in \mathbb{R}^{k+}}{\arg\min} \frac{1}{2} \|\mathbf{D}^+\mathbf{s}^+ - \mathbf{x}\|_2^2 + \lambda \|\mathbf{s}^+\|_1 \tag{3.2}$$

In this case λ is a parameter controlling the trade-off between reconstruction and sparsity of the coefficients. The search for a sparse solution actually provides the upcoming Binary SRC classification with two-fold benefits. When a test sample is a valid sample for the target category, it normally shares many similarities with the samples from the positive dictionary, thus making the sparse solution a good reconstruction for this sample. On the other hand, when the test sample is not a valid sample for the target category, the sparsity of the solution will prevent it from using too many training samples to build a good reconstruction. We can thus build

a discriminative classifier by simply comparing the reconstruction residuals (as we will define later).

The sparse reconstruction coefficients $\mathbf{s}^-$ for a test sample $\mathbf{x}$ with a negative dictionary $\mathbf{D}^-$ can be found in like manner by solving the following $l1$-norm minimization problem:

$$\hat{\mathbf{s}}^- = \underset{\mathbf{s}^- \in \mathbb{R}^{k-}}{\arg\min} \frac{1}{2}\|\mathbf{D}^-\mathbf{s}^- - \mathbf{x}\|_2^2 + \lambda\|\mathbf{s}^-\|_1 \tag{3.3}$$

We define the reconstruction residual of a sample $\mathbf{x}$ using a dictionary $\mathbf{D}$ as follows:

$$\mathcal{R}(\mathbf{x}, \mathbf{D}) = \|\mathbf{D}\hat{\mathbf{s}} - \mathbf{x}\|_2 \tag{3.4}$$

Finally, a test sample $\mathbf{x}$ is classified by comparing the two reconstruction residuals of the sample on both positive and negative dictionaries, *i.e.*, the predicted label by Bi-SRC for $\mathbf{x}$ is defined as:

$$y_{prediction} = sign(\mathcal{R}(\mathbf{x}, \mathbf{D}^-) - \mathcal{R}(\mathbf{x}, \mathbf{D}^+) - \theta_{pn}) \tag{3.5}$$

Where θ_{pn} is a small threshold that controls the balance between a positive and a negative category.

For convenience, we also denote:

$$h_{bs}(\mathbf{x}, \{\mathbf{D}^+, \mathbf{D}^-\}) = \mathcal{R}(\mathbf{x}, \mathbf{D}^-) - \mathcal{R}(\mathbf{x}, \mathbf{D}^+) \tag{3.6}$$

as the hypothesis of Bi-SRC for sample $\mathbf{x}$ and a dictionary pair $\{\mathbf{D}^+, \mathbf{D}^-\}$. Similarly, we also denote:

$$\begin{aligned} &h_{bs}(\mathbf{X}, \{\mathbf{D}^+, \mathbf{D}^-\}) \\ &\quad = [h_{bs}(\mathbf{x}_1, \{\mathbf{D}^+, \mathbf{D}^-\}), \ldots, h_{bs}(\mathbf{x}_n, \{\mathbf{D}^+, \mathbf{D}^-\})]^T \end{aligned} \tag{3.7}$$

as the hypotheses of Bi-SRC for a sample matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ and a dictionary

pair $\{\mathbf{D}^+, \mathbf{D}^-\}$.

Another benefit of this sparse representation based classifier is that, as the Eq. (3.1) is assumed to be under-determined, *i.e.*, the feature dimension is assumed to be smaller than the dictionary size. Under this assumption, the Binary SRC is naturally able to handle very compact feature vectors. We will show in section 3.5.3 that the proposed DTL, using Bi-SRC as its building block, retains its performance advantage compared to other state-of-the-art transfer learning methods even when the feature vectors are surprisingly compact.

As can be seen above, the proposed Binary SRC classifier uses a pair consisting of a positive dictionary with positively correlated samples and a negative dictionary with negatively correlated samples. With this assumption in mind, we can search for additional positive dictionaries that are positively correlated to a target category when trying to solve our transfer learning problem for target classification, *i.e.*, image scene classification, with a number of source categories with abundant labeled samples. Similarly, we can also search for additional negative dictionaries that are negatively correlated to the target category. We achieve knowledge transfer for a better target classifier by combining the hypotheses of all these dictionary pairs extracted from the source categories. In the following section, we introduce the method for finding these dictionary pairs from source categories and for learning proper weights for these dictionaries in order to obtain a better combination of hypotheses.

3.3.3 The basic classification model

Take $\mathbf{D}_{0+} \in \mathbb{R}^{m \times N_+}$ a dictionary consisting of N_+ positive samples in a target training set $\mathcal{T}$, and $\mathbf{D}_{0-} \in \mathbb{R}^{m \times N_-}$ a dictionary consisting of N_- negative samples in $\mathcal{T}$. The source sets $\mathcal{A}_l$ can also be packed into dictionaries: $\mathbf{D}_l \in \mathbb{R}^{m \times N_l}$ for $l = 1, \ldots, L$.

We remind you that, in our transfer learning setting, the target training set is small and unbalanced, *i.e.*, with very few positive training samples and relatively more negative training ones. Our goal is to leverage the existing L source categories to help learning of the target category classifier. For this purpose, we make use of

similarities and dissimilarities between source and target in two complementary ways.

First, using target negative samples as natural dissimilarities, we seek in the source for categories that are similar to the target category in the feature space and that can thus act as positive dictionaries for classifying the target samples. Take a non-negative cost function $\mathcal{C}(h(\mathbf{X}), \mathbf{y}, \mathbf{w})$, which we will define later, that measures the cost incurred in predicting $h(\mathbf{X})$ for $\mathbf{X} \in \mathbb{R}^{m \times n}$ instead of ground truth labels $\mathbf{y} \in \mathbb{R}^n$, where $\mathbf{w}$ is a weight vector for the training samples $\mathbf{X}$. In this case, the positive source categories described in the above paragraph can be defined by:

$$\begin{aligned} \mathcal{D}_{pos} = \{\mathbf{D}_i \mid \mathcal{C}(h_{bs}(\mathbf{X}_{tr}, \{\mathbf{D}_i, \mathbf{D}_{0-}\}), \mathbf{y}, \mathbf{w}) \leqslant \theta \\ and \;\; |\mathcal{D}_{pos}| \leqslant K_{max}, \;\; i \in [1, L]\} \end{aligned} \tag{3.8}$$

where $\mathbf{X}_{tr}$ is the training data matrix[2], θ a cost threshold that controls the quality of the chosen source categories for classifying the target category, and K_{max} a threshold which is the maximum number of dictionaries in this set. If the number of dictionaries yielding a cost lower than θ is larger than K_{max}, we then only keep the first K_{max} dictionaries with the lowest costs. The parameter K_{max} can be considered as a regularization parameter to avoid the model becoming too complex to cause over-fitting.

Second, we also seek both similarities and dissimilarities in the source, *i.e.*, pairs of source categories each of which has one source category positively correlated to the target, whereas the other is negatively correlated. They thus play the role of a pair of positive and negative dictionaries *w.r.t.* the target category. Specifically, these source dictionary pairs can be defined in like manner as:

$$\begin{aligned} \mathcal{D}_{pair} = \{\{\mathbf{D}_i, \mathbf{D}_j\} \mid \mathcal{C}(h_{bs}(\mathbf{X}_{tr}, \{\mathbf{D}_i, \mathbf{D}_j\}), \mathbf{y}, \mathbf{w}) \leqslant \theta \\ and \;\; |\mathcal{D}_{pair}| \leqslant K_{max}, \;\; i, j \in [1, L]\} \end{aligned} \tag{3.9}$$

[2]As in $h_{bs}(\mathbf{X}_{tr}, \{\mathbf{D}_i, \mathbf{D}_0-\})$ training data appear as both a dictionary and samples to be reconstructed: a cross-validation procedure can be used to avoid the possible over-fitting problem.

Once source categories useful for their similarities and dissimilarities with the target have been chosen, a classifier can be constructed that leverages the existence of source sets along with the target training set. Specifically, the hypothesis function h_{bcm} of the basic classification model is defined as follows:

$$\begin{aligned} h_{bcm}(\mathbf{x}, \{\mathbf{D}_0+, \mathbf{D}_0-\}, \{\mathbf{D}_l \mid l \in [1, L]\}, \mathbf{w}, \theta) = \\ \alpha_0 \cdot h_{bs}(\mathbf{x}, \{\mathbf{D}_0+, \mathbf{D}_0-\}) + \\ [h_{bs}(\mathbf{x}, \{\mathbf{D}_i, \mathbf{D}_{0-}\})]_{\mathbf{D}_i \in \mathcal{D}_{pos}} \cdot \boldsymbol{\alpha}_{pos} + \\ [h_{bs}(\mathbf{x}, \{\mathbf{D}_i, \mathbf{D}_j\})]_{\{\mathbf{D}_i, \mathbf{D}_j\} \in \mathcal{D}_{pair}} \cdot \boldsymbol{\alpha}_{pair} \end{aligned} \tag{3.10}$$

where $[h_{bs}(\mathbf{x}, \{\mathbf{D}_i,\ \mathbf{D}_{0-}\})]_{\mathbf{D}_i \in \mathcal{D}_{pos}}$ and $[h_{bs}(\mathbf{x}, \{\mathbf{D}_i, \mathbf{D}_j\})\ ]_{\{\mathbf{D}_i, \mathbf{D}_j\} \in \mathcal{D}_{pair}}$ are line vectors containing the hypotheses for a sample $\mathbf{x}$ by Bi-SRC given a pair of positive and negative dictionaries, whereas α_0, $\boldsymbol{\alpha}_{pos}$ and $\boldsymbol{\alpha}_{pair}$ are weight or column weight vectors for the corresponding hypotheses.

The weight α for a given hypothesis $h(\mathbf{x})$ is defined as:

$$\alpha = \frac{1}{2} log \left(\frac{1 - \mathcal{C}(h(\mathbf{X}_{tr}), \mathbf{y}, \mathbf{w})}{\mathcal{C}(h(\mathbf{X}_{tr}), \mathbf{y}, \mathbf{w})} \right) \tag{3.11}$$

Regarding the cost function $\mathcal{C}(\cdot)$ to be defined, the more commonly used are cross entropy and mean squared error, which aim at achieving the best correct classification rate. In our transfer learning setting, the target training set is very unbalanced. Therefore, classification accuracy is not a fair criterion for evaluating classifier performance. As a result, we propose a new Wilcoxon-Mann-Whitney (WMW) statistic [Wilcoxon 1945] [Mann & Whitney 1947] based cost function, which aims at maximizing the Area Under the ROC Curve (AUC).

The WMW statistic is defined as follows:

$$W = \frac{\sum_{i=1}^{n_a} \sum_{j=1}^{n_b} I(a_i, b_j)}{n_a \cdot n_b} \tag{3.12}$$

where:

$$I(a_i, b_j) = \begin{cases} 1 & \text{if } a_i > b_j \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

for pairwise comparison between a sample $\{a_i \mid i = 1, \ldots, n_a\}$ of a random variable A and a sample $\{b_j \mid j = 1, \ldots, n_b\}$ of a random variable B. The statistic W is an estimator of $P[A > B]$. If we identify $\{a_i\}$ as the classifier outputs for n_a positive samples, and $\{b_j\}$ as the classifier outputs for n_b negative samples, we obtain the AUC of the classifier through Eq. (3.12). Based on this, we propose the following cost function:

$$\begin{aligned} &\mathcal{C}(h(\mathbf{X}), \mathbf{y}, \mathbf{w}) = \\ &1 - \frac{\sum_{i \in \mathcal{I}^+} \sum_{j \in \mathcal{I}^-} (I(h(x_i), h(x_j)) \cdot w_i \cdot w_j)}{(\sum_{i \in \mathcal{I}^+} w_i) \cdot (\sum_{j \in \mathcal{I}^-} w_j)} \\ &s.t.\ \mathcal{I}^+ = \{i \in [1, |\mathbf{y}|] \mid y_i = +1\} \\ &\ and\ \mathcal{I}^- = \{i \in [1, |\mathbf{y}|] \mid y_i = -1\} \end{aligned} \tag{3.14}$$

where:

$$I(h(x_i), h(x_j)) = \begin{cases} 1 & \text{if } h(x_i) > h(x_j) \\ 0.5 & \text{if } h(x_i) = h(x_j) \\ 0 & \text{if } h(x_i) < h(x_j) \end{cases} \tag{3.15}$$

This cost function is non-differentiable. Fortunately, in our proposed basic classification model, the search space is not large, with a space size L for Eq. (3.8) and L^2 for Eq. (3.9)). We only need to conduct an exhaustive search in the corresponding dictionary space. No gradient-based optimization is needed.

3.3.4 Boosting the Area Under the ROC Curve

In the previous DTL basic classification model, which combines several Bi-SRC models using some source categories, the weights α for the model combination are learned simply based on the outputs of the cost function over various dictionary

pairs. Therefore, we cannot guarantee that the resultant combined model is the best out of all the possible combined models. In this subsection, we propose a boosting method to further boost the performance of the basic DTL model taken as a weak learner. The aim here is to find the proper weights for each of the additional dictionary pairs to ensure that the final model combining all these dictionary pairs has the best discriminative ability.

However, given the unbalanced distribution of positive and negative samples in the training set, we propose to boost as in [Long & Servedio 2007] the Area Under the ROC Curve (AUC) instead of classification accuracy, which, although it is the most commonly utilized optimization criterion, cannot work well with unbalanced data.

Specifically, to solve unbalanced data distribution, we need to pursue good classification performance on both positive and negative distribution. Consequently, we run two parallel AdaBoost processes: one on positive and the other on negative data distribution. At the end of each iteration, the model weights of the two parallel AdaBoost processes are combined.

Algorithm 1 depicts the boosting process of the proposed DTL. The inputs are: $\mathbf{D}_{0+}$ with N_+ positive training samples, $\mathbf{D}_{0-}$ with N_- negative training samples, $\mathbf{D}_l$ with N_l samples in l-th source category, and a maximum iteration number T. The parameter λ for Eq. (3.2) and Eq. (3.3) can either be prefixed or learned through cross-validation on training data.

The first step in the DTL training algorithm is a preparation step: we form the training data matrix $\mathbf{X}_{tr}$ by concatenating $\mathbf{D}_{0+}$ and $\mathbf{D}_{0-}$. We then form the ground-truth label vector $\mathbf{y}$ by concatenating a vector of length N_+ with all $+1$ in it, and another vector of length N_- with all -1 in it.

In DTL we maintain the sample weights in a vector $\mathbf{w}^t$ at iteration t. This weight vector is first initialized as $\mathbf{w}^1$ in step 2 of the algorithm.

To choose dictionaries as in Eq. (3.8) and Eq. (3.9), we need a cost threshold θ. As this threshold varies across iterations, we note it at iteration t as θ_t. In step 3 of DTL we initialize it as θ_1, which is the cost of Bi-SRC classification of the training samples with the initial sample weights using the training samples themselves as

positive and negative dictionary pairs (cost is calculated using cross-validation).

Steps 4 through 9 form the main part of the boosting process with T iterations. At each iteration, we first obtain the DTL basic classification model (in step 5) for the current sample weights and the corresponding cost threshold. We then calculate (in step 6) the current model weight α_t as the average of a model weight α_{t+} calculated on the positive distribution of the training data and another model weight α_{t-} calculated on the negative distribution of the training data. We then update the sample weight vector $\mathbf{w}^{t+1}$ in step 7. Finally, we update the cost threshold θ_{t+1} in step 8, by calculating the cost of Bi-SRC classification using cross-validation as in step 3 with the updated sample weights.

After T iterations, in step 10 we merge the classification models of all iterations into one final model $h_{DTL}(\cdot)$.

In the final merge process, the weights of the same additional dictionary pair from different models can be simply added up. Eq. (3.16) defines how two DTL basic models can be easily merged into one single model by adding up the weights of the same additional dictionary pairs.

$$\begin{aligned} h_{bcm}(\mathbf{x}, \{\mathbf{D}_0+, \mathbf{D}_0-\}, \{\mathbf{D}_l \mid l \in [1, L]\}, \mathbf{w}_1, \theta_1) + \\ h_{bcm}(\mathbf{x}, \{\mathbf{D}_0+, \mathbf{D}_0-\}, \{\mathbf{D}_l \mid l \in [1, L]\}, \mathbf{w}_2, \theta_2) = \\ (\alpha_{0,1} + \alpha_{0,2}) \cdot h_{bs}(\mathbf{x}, \{\mathbf{D}_0+, \mathbf{D}_0-\}) + \\ [h_{bs}(\mathbf{x}, \{\mathbf{D}_i, \mathbf{D}_{0-}\})]_{\mathbf{D}_i \in \mathcal{D}_{pos,1} \cup \mathcal{D}_{pos,2}} \cdot \boldsymbol{\alpha}_{pos} + \\ [h_{bs}(\mathbf{x}, \{\mathbf{D}_i, \mathbf{D}_j\})]_{\{\mathbf{D}_i, \mathbf{D}_j\} \in \mathcal{D}_{pair,1} \cup \mathcal{D}_{pair,2}} \cdot \boldsymbol{\alpha}_{pair} \end{aligned} \tag{3.16}$$

Where $\boldsymbol{\alpha}_{pos}$ is the weight vector for the dictionary pairs in the union set $\mathcal{D}_{pos,1} \cup \mathcal{D}_{pos,2}$ by adding up corresponding weights in $\boldsymbol{\alpha}_{pos,1}$ and $\boldsymbol{\alpha}_{pos,2}$; and $\boldsymbol{\alpha}_{pair}$ is the weight vector for the dictionary pairs in the union set $\mathcal{D}_{pair,1} \cup \mathcal{D}_{pair,2}$ by adding up corresponding weights in $\boldsymbol{\alpha}_{pair,1}$ and $\boldsymbol{\alpha}_{pair,2}$.

Algorithm 1 DTL training algorithm

Input:

1: $\mathbf{D}_{0+} \in \mathbb{R}^{m \times N_+}$; $\mathbf{D}_{0-} \in \mathbb{R}^{m \times N_-}$; $\{\mathbf{D}_l \in \mathbb{R}^{m \times N_l} \mid l = 1, \ldots, L\}$; maximum iteration T;

2: $\mathbf{X}_{tr} = [\mathbf{D}_{0+}, \mathbf{D}_{0-}]$; $\mathbf{y} = \begin{bmatrix} \mathbf{y}^+ \\ \mathbf{y}^- \end{bmatrix}$ with $\mathbf{y}^+ = [+1]^{N_+}$ and $\mathbf{y}^- = [-1]^{N_-}$;

3: Initialize sample weights: $\mathbf{w}^1 = \begin{bmatrix} \mathbf{w}^{1+} \\ \mathbf{w}^{1-} \end{bmatrix}$
with $\mathbf{w}^{1+} \in [0,1]^{N_+}$, $\mathbf{w}^{1-} \in [0,1]^{N_-}$,
and $\sum_{i=1}^{(N_+ + N_-)} w_i^1 = 1$;

4: initialize cost threshold:
$\theta_1 = \mathcal{C}(h_{bs}(\mathbf{X}_{tr}, \{\mathbf{D}_{0+}, \mathbf{D}_{0-}\}), \mathbf{y}, \mathbf{w}^1)$

5: **for** $t = 1, \ldots, T$ **do**

6: get the current basic classification model $h_t(\mathbf{x})$:
$h_t(\mathbf{x}) =$
$h_{bcm}(\mathbf{x}, \{\mathbf{D}_0+, \mathbf{D}_0-\}, \{\mathbf{D}_l \mid l \in [1, L]\}, \mathbf{w}^t, \theta_t)$
as defined in Eq. (3.10), where $\mathcal{D}_{pos}$, $\mathcal{D}_{pair}$ and α are defined in Eq. (3.8), Eq. (3.9) and Eq. (3.11) for input weight vector $\mathbf{w}^t$ and cost threshold θ_t;

7: calculate the current model weight α_t:
$\varepsilon_{t+} = \frac{1}{2}(1 - sign(h_t(\mathbf{D}_{0+})))^{Transpose} \cdot \mathbf{w}^{t+}$
$\alpha_{t+} = \frac{1}{2}\log(\frac{1-\varepsilon_{t+}}{\varepsilon_{t+}})$;
$\varepsilon_{t-} = \frac{1}{2}(1 + sign(h_t(\mathbf{D}_{0-})))^{Transpose} \cdot \mathbf{w}^{t-}$
$\alpha_{t-} = \frac{1}{2}\log(\frac{1-\varepsilon_{t-}}{\varepsilon_{t-}})$;
$\alpha_t = \frac{\alpha_{t+} + \alpha_{t-}}{2}$;

8: update sample weights:
$w_i^{(t+1)+} = w_i^{t+} e^{-\alpha_{t+} h_t(\mathbf{x}_i^+)}, \ \forall i \in [1, N^+]$;
$w_i^{(t+1)-} = w_i^{t-} e^{\alpha_{t-} h_t(\mathbf{x}_i^-)}, \ \forall i \in [1, N^-]$;
$\mathbf{w}^{t+1} = \frac{\mathbf{w}^{t+1}}{\sum_{i=1}^{N} w_i^{t+1}}$;

9: update cost threshold:
$\theta_{t+1} = \mathcal{C}(h_{bs}(\mathbf{X}_{tr}, \{\mathbf{D}_{0+}, \mathbf{D}_{0-}\}), \mathbf{y}, \mathbf{w}^{t+1})$;

10: **end for**

11: merge classification models of all iterations into one final model as in Eq. (3.16):
$h_{DTL}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t \cdot h_t(\mathbf{x})$

Output:

12: $h_{DTL}(\mathbf{x})$

3.4 Theoretical Analysis of the proposed DTL boosting

In this section, the error bound of the proposed DTL boosting is analyzed and its time complexity estimated.

3.4.1 Error bounds for DTL boosting

Theorem 6 in [Freund & Schapire 1997] gives an upper bound for the error rate of the AdaBoost final output:

$$\varepsilon \leqslant 2^T \prod_{t=1}^{T} \sqrt{\varepsilon_t(1-\varepsilon_t)} \tag{3.17}$$

where ε_t is the error rate for a weak learner hypothesis in iteration t, and T is the number of iterations. Just as in AdaBoost we assume $\varepsilon_t = \frac{1}{2} - \gamma_t$, where $\gamma_t \in [0, \frac{1}{2}]$ is the advantage of t-th weak learner over random guess. This upper bound can also be written in the following form:

$$\varepsilon \leqslant \prod_{t=1}^{T} \sqrt{1-4\gamma_t^2} \tag{3.18}$$

As can be seen above, since $\sqrt{1-4\gamma_t^2} \leqslant 1$, this upper bound shows that, when not all γ_t equal 0, the error rate of the AdaBoost final hypothesis can continue to decrease as iterations increase.

If we assume all the error rates are equal to $\frac{1}{2} - \gamma$, Eq. (3.18) can be further simplified as:

$$\begin{aligned} \varepsilon &\leqslant (1-4\gamma^2)^{\frac{T}{2}} \\ &= e^{-T \cdot KL(\frac{1}{2} \parallel \frac{1}{2}-\gamma)} \\ &\leqslant e^{-2T\gamma^2} \end{aligned} \tag{3.19}$$

where $KL(a \parallel b) = a \cdot log(\frac{a}{b}) + (1-a) \cdot log(\frac{1-a}{1-b})$ the Kullback-Leibler divergence. Eq. (3.19) can be used to obtain the maximum number of iterations of the boosting algorithm, which is sufficient to achieve an error rate ε as:

$$T = O\left(\frac{log(\frac{1}{\varepsilon})}{2\gamma^2}\right) \tag{3.20}$$

In DTL, we conduct two parallel AdaBoost processes, one on positive distribution and the other on negative distribution. By defining positive distribution

as $\mathcal{D}^+$ and negative distribution as $\mathcal{D}^-$, DTL functions by running AdaBoost on $\frac{1}{2}\mathcal{D}^+ + \frac{1}{2}\mathcal{D}^-$.

In round t, each AdaBoost process passes its reweighted distribution $\mathcal{D}_t$ to the weak learner $h_t(\cdot)$. Assume that the weak learner $h_t(\cdot)$ has an advantage γ over random guess on positive distribution as well as having an advantage γ over random guess on negative distribution. In this case, no matter how $\mathcal{D}_t$ can be broken down into a mixture of $\mathcal{D}_t^+$ and $\mathcal{D}_t^-$, the following inequality always holds:

$$P_{(\mathbf{x},y)\in\mathcal{D}_t}[h_t(\mathbf{x}) \neq y] \leqslant \frac{1}{2} - \gamma \tag{3.21}$$

As shown in Eq. (3.19) and Eq. (3.20), with an error rate $\frac{1}{2}-\gamma$ at each iteration, the AdaBoost process takes at most $T = O\left(\frac{log(\frac{1}{\varepsilon})}{2\gamma^2}\right)$ iterations to achieve an error rate of at most ε on distribution $\frac{1}{2}\mathcal{D}^+ + \frac{1}{2}\mathcal{D}^-$.

Assume that the final output of DTL displays an error rate of ε_+ over positive distribution and shows an error rate of ε_- over negative distribution, *i.e.*:

$$P_{x\in\mathcal{D}^+}[h_{DTL}(\mathbf{x}) = +1] = 1 - \varepsilon_+ \tag{3.22}$$

$$P_{x\in\mathcal{D}^-}[h_{DTL}(\mathbf{x}) = -1] = 1 - \varepsilon_- \tag{3.23}$$

With respect to the error rate ε on distribution $\frac{1}{2}\mathcal{D}^+ + \frac{1}{2}\mathcal{D}^-$, we also have $\varepsilon = \frac{\varepsilon_+ + \varepsilon_-}{2}$.

As shown in subsection 3.3.3, the AUC of a classifier h_{DTL} over distribution $\mathcal{D}$ can be defined as follows:

$$\begin{aligned} AUC(h_{DTL};\mathcal{D}) = \\ P_{\mathbf{x}_1\in\mathcal{D}^+,\mathbf{x}_2\in\mathcal{D}^-}[h_{DTL}(\mathbf{x}_1) > h_{DTL}(\mathbf{x}_2)] \\ +\frac{1}{2}P_{\mathbf{x}_1\in\mathcal{D}^+,\mathbf{x}_2\in\mathcal{D}^-}[h_{DTL}(\mathbf{x}_1) = h_{DTL}(\mathbf{x}_2)] \end{aligned} \tag{3.24}$$

Therefore, we have:

$$
\begin{aligned}
&AUC(h_{DTL};\mathcal{D}) = \\
&P_{\mathbf{x}_1\in\mathcal{D}^+,\mathbf{x}_2\in\mathcal{D}^-}[h_{DTL}(\mathbf{x}_1)=+1, h_{DTL}(\mathbf{x}_2)=-1] \\
&+\frac{1}{2}(P_{\mathbf{x}_1\in\mathcal{D}^+,\mathbf{x}_2\in\mathcal{D}^-}[h_{DTL}(\mathbf{x}_1)=+1, h_{DTL}(\mathbf{x}_2)=+1] \\
&+P_{\mathbf{x}_1\in\mathcal{D}^+,\mathbf{x}_2\in\mathcal{D}^-}[h_{DTL}(\mathbf{x}_1)=-1, h_{DTL}(\mathbf{x}_2)=-1]) \\
&=(1-\varepsilon_+)(1-\varepsilon_-)+\frac{1}{2}(\varepsilon_+(1-\varepsilon_-)+\varepsilon_-(1-\varepsilon_+)) \\
&=1-\frac{\varepsilon_+ + \varepsilon_-}{2} \\
&=1-\varepsilon
\end{aligned}
\tag{3.25}
$$

This shows that the AUC of DTL is bounded with a lower bound (since ε is bounded with an upper bound according to Eq. (3.17)), and it takes at most $T = O\left(\frac{log(\frac{1}{\varepsilon})}{2\gamma^2}\right)$ iterations to achieve an AUC of at least $1-\varepsilon$.

Generalization error: The previous AUC bound is estimated on training distribution. To ensure a generalization error close to the empirical error on the training set, we could use the structural risk minimization method introduced in Vapnik's book [Vapnik & Kotz 1982]. As shown in Theorem 6.7 in [Vapnik & Kotz 1982], for a probability less than δ, an upper bound, which is a function of training set size, VC-dimension of the class of hypotheses and δ, is given for the difference in generalization error and empirical error. Theorem 8 in [Freund & Schapire 1997] gives an upper bound on the VC-dimension of the class of final hypotheses generated by AdaBoost after T iterations as a function of T. Using these two bounds, we could therefore choose the proper T minimizing the upper bound of the generalization error.

However, as shown in [Freund & Schapire 1997], the upper bounds on the generalization error generated in this way might be larger than the actual value. Thus the chosen number of iterations T might be much smaller than the optimal value, leading to inferior performance. A simple but effective alternative is to use "cross-validation", in which the value of T is chosen to be that for which the error of the final hypothesis on the validation set is minimized. In this work, we estimate the value of T using cross-validation on target samples before training.

3.4.2 Time complexity of DTL

In Algorithm 1, as the source dictionaries $\{\mathbf{D}_l \mid l = 1, \ldots, L\}$ are not modified during the Algorithm, we can actually calculate the sparse coefficients for training data with each source dictionary (using Eq. (3.2)) at the beginning and then reuse these coefficients in each boosting iteration for the Bi-SRC hypothesis. In this way, duplicate calculations in each iteration can be avoided, thus increasing algorithm efficiency.

Consequently, DTL training time can be divided into two parts: first, the sparse reconstruction step and, second, the boosting iterations. For the sparse reconstruction step we use a fast implementation [Mairal 2014] of the LARS algorithm [Efron *et al.* 2004] to solve the $l1$ minimization problem, making time complexity of the reconstruction part approximately $O(mLN_lN)$, where m is the feature vector size, L the number of source categories, N_l the size of one source category, and N the size of the target training set. For the boosting iterations, costing calculations in each iteration, *e.g.*, optimization algorithms as in CCTL [Qi *et al.* 2011]), are not needed. Its time complexity is only $O(N^2L^2T)$, where T is the number of iterations. This is apparently less than the time complexity for boosting iterations in CCTL, which is $O(m^2NN_lLT)$, since each boosting iteration of CCTL has $L+1$ optimization problems to solve.

For DTL prediction time, since we have merged all the learned models in all the iterations into one final model, time complexity only depends on reconstruction of the test set with the chosen source sets, which is at most (if all the source categories are chosen as dictionaries, normally not all of them are chosen) $O(mLN_lN_{test})$, where N_{test} is the number of samples in the test set. This is less than the time complexity for the prediction time of normal boosting algorithms since, for most boosting algorithms, this prediction time usually depends on the number of iterations T: the more iterations we have in the training phase, the more time we spend in prediction.

The real training and prediction times of these two methods are given in Section 3.5.2, TABLE 3.2. As can be seen, with BOW features, DTL is 100 times faster

than CCTL for training and 70 times faster than CCTL for testing (predicting). Prediction time is usually considerable for a classification model, because a model can be pre-trained, while prediction time determines only how long we have to wait to get the predictions for new samples. Thus, with a much smaller average prediction time, DTL is also in practice a more efficient method than CCTL.

3.5 Experiments

The proposed DTL algorithm was benchmarked using two different datasets. The first is the NUS-WIDE Scene dataset [Chua *et al.* 2009], which was also used by G. Qi *et al.* in [Qi *et al.* 2011] for benchmarking their cross-category transfer learning experiments. This dataset allows a comparison with other state-of-the-art transfer learning algorithms, and in particular CCTL as proposed in [Qi *et al.* 2011]. A second scene dataset with fewer target training samples and more source categories is also used to further highlight the behavior of the proposed DTL algorithm. This second dataset uses 52 categories from the SUN database [Xiao *et al.* 2010]. Two performance metrics, namely AUC (Area Under the ROC Curve) and AP (Average Precision), are used for a fair comparison of different methods.

3.5.1 Results on the NUS-WIDE Scene

The NUS-WIDE Scene dataset is a natural scene image set crawled from Flicker.com, with 17463 training images and 17463 testing images. It contains 33 natural scene categories. Using exactly the same experimental setting as in [Qi *et al.* 2011], the 10 categories with the fewest positive examples are chosen as the target categories, while the remaining 23 categories are treated as source categories. The number of positive image samples in the training set and the test set of each target category, along with the number of image samples in each source category, are shown in Fig. 3.1. This dataset provides for each image a 500-dimensional feature vector computed through the bag-of-visual words approach using SIFT descriptors [3]. TABLE 3.1 shows the results achieved by the proposed

[3] http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm

DTL method in comparison with a baseline AdaBoost algorithm, several state-of-the-art transfer learning algorithms experimented in [Qi *et al.* 2011], and SparseTL in [Lu *et al.* 2014]. The results by AdaBoost were achieved using only the target training samples and are used as the baseline for comparison. The CML (Correlative Multi-Label) [Qi *et al.* 2007] classifier is a multi-label classifier based on structural SVM. The TaskTrAdaBoost (Task Transfer AdaBoost) [Yao & Doretto 2010] algorithm is a parameter transfer method combining a set of pre-learned SVM classifiers. The CCTL (Cross Category Transfer Learning) [Qi *et al.* 2011] algorithm is a cross category label propagation algorithm, which also uses an AdaBoost framework. The SparseTL is the first method using both similarity and dissimilarity for transfer learning. For DTL experiments, the number of iterations is pre-estimated using cross-validation on the target set: $T = 30$, the maximum number of dictionary pairs at each iteration is set to $K_{max} = 2$, while the parameter λ is determined by a cross-validation process for each target category during training. As can be seen in TABLE 3.1, the proposed DTL algorithm outperforms other transfer learning algorithms with the best average AUC of 0.7738. In particular, DTL surpasses CCTL with a performance gain of 6 points.

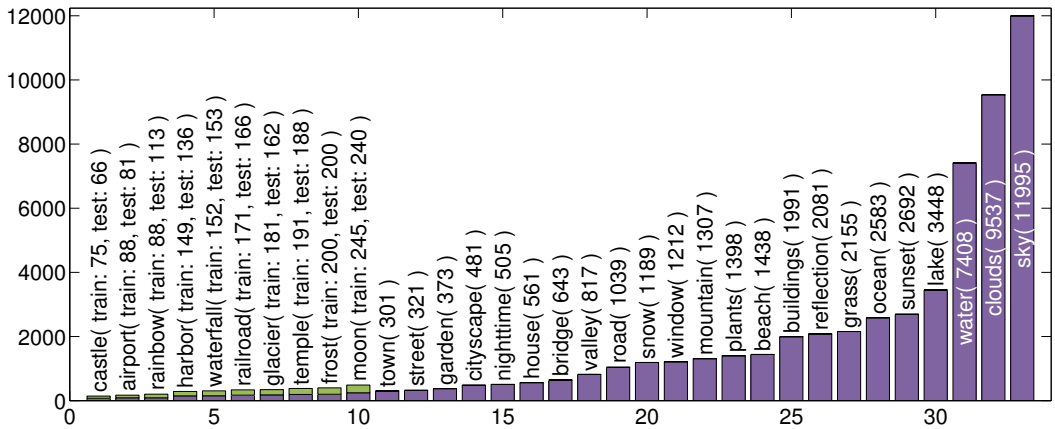

Figure 3.1: Nus-Wide Scene image dataset. In this Figure each bar represents a category: the height of the bar represents the number of positive samples in the corresponding category; the text on the top of each bar shows the category name and the number of positive samples; the first 10 categories with the fewest image samples are chosen as target categories, and the remaining 23 categories are treated as source categories

Table 3.1: Experimental results on the NUS-WIDE SCENE dataset (The results are Average AUC (Area Under the ROC Curve) with standard deviation; the results in the first 5 rows are directly quoted from [Qi *et al.* 2011], the result for SparseTL is quoted from [Lu *et al.* 2014])

Category	Average AUC
AdaBoost	0.5717 ± 0.07
CML	0.6542 ± 0.11
TaskTrAdaBoost	0.6637 ± 0.1
CCTL	0.7159 ± 0.1
SparseTL	0.7238 ± 0.08
DTL	**0.7738** ± 0.08

3.5.2 Results on the SUN dataset

The SUN database [Xiao *et al.* 2010] is a large-scale scene recognition database. It has a total of 899 categories and 130519 images, out of which 397 categories are properly sampled, *i.e.*, have at least 100 images per category, and can be downloaded from the SUN database website. To form a dataset for testing transfer learning algorithms, we choose 10 of these 397 categories, which have 100 images per category, as the target categories.

Another 42 categories, with more than 500 images per category, are chosen as source categories.

We use the first 50 images in each target category as training samples and the other 50 images as test samples. The number of positive image samples in each source category, as well as the numbers of training and testing samples in each target category, are illustrated in Fig. 3.2.

We use two different features for experiments on the SUN database. The first is a 500-dimensional bag-of-visual-words feature computed through densely sampled SIFT local descriptors. The second is a 50-dimensional PCA feature selection output, selected from the 4096-dimensional fc7-layer output of an AlexNet fine-tuned on the SUN 52 categories (10 target categories with 42 source categories). As a hyper-parameter, this feature size is tuned through a validation process using preliminary experiments. We also provide a comparison for using different feature sizes in sub-section 3.5.3.2.

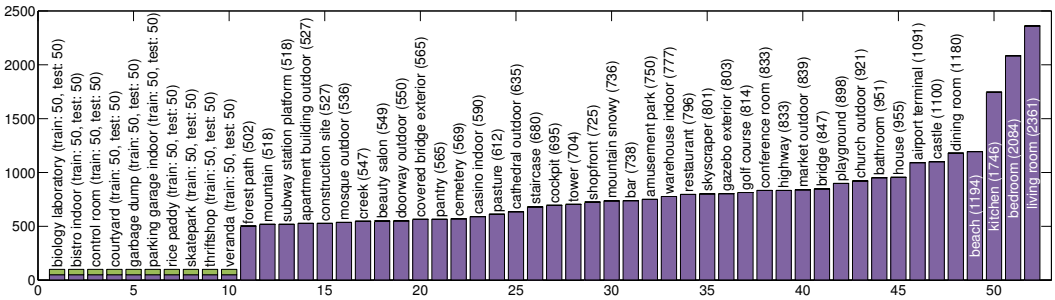

Figure 3.2: SUN dataset: in this Figure each bar represents a category: the height of the bar represents the number of positive samples in the corresponding category; the text on the top of each bar shows the category name and the number of positive samples; the first 10 categories with the fewest image samples are chosen as target categories, and the remaining 42 categories are treated as source categories

We use 5 baseline methods for comparison with our DTL method. The first is the Sparse Representation based Classifier (SRC) [Wright *et al.* 2009]. The second is the AdaBoost method [Freund & Schapire 1997] with a CART decision tree (with 3 nodes) as the weak learner. They are both baseline classification methods without knowledge transfer. The last three methods for comparison are state-of-the-art transfer learning methods, including MULTIpLE [Kuzborskij *et al.* 2013], GreedyTL [Kuzborskij *et al.* 2015] and CCTL [Qi *et al.* 2011]. The SRC and AdaBoost[4] are not transfer learning methods, and are trained using only the target training samples. However, MULTIpLE, GreedyTL, CCTL and DTL are trained using both the target training samples and the samples from the 42 source categories.

For MULTIpLE, GreedyTL and CCTL experiments, we use the default parameter settings along with the code provided by the authors. The MULTIpLE method is designed for the N to N + 1 incremental learning scenario, which assumes there is a classifier for N known categories with the addition of some new training samples from a new category extending this classifier to classify the N + 1 categories. For a fair comparison with other TL methods, when classifying a target category, we assume that the other 9 target categories along with the 42 source categories are the N known categories for MULTIpLE, and that the current target category is the new

[4]The AdaBoost experiments are conducted with the 'GML AdaBoost Matlab Toolbox'.

category to be added. Furthermore, as MULTIpLE and GreedyTL are hypothesis transfer learning methods, which assume that the learner cannot access the source domain directly but rather operates on the basis of induced hypotheses, we have carried out SRC classifications on the known categories to prepare the hypotheses needed for MULTIpLE and GreedyTL.

For DTL experiments, the number of iterations is pre-estimated using cross-validation on the target set: $T = 30$ for BOW features and $T = 10$ for AN50D features. The maximum number of dictionary pairs in each iteration is set to $K_{max} = 2$, while the parameter λ is determined by a cross-validation process for each target category during training.

In order to conduct a comprehensive comparison of different methods, we use two different performance metrics. The first is the average Area Under the ROC Curve (average AUC), while the second is the mean Average Precision (mean AP). The results are provided in TABLE 3.2. An interesting fact is that the baseline methods show different preferences for the two features used. For the two baseline methods, SRC shows better performance with AN50D features, whereas AdaBoost shows better performance with BOW features. For the transfer learning methods, MULTIpLE shows better performance with AN50D features. With BOW features, MULTIpLE shows a performance that is worse even than the baseline methods SRC and AdaBoost. On the contrary, CCTL yields second-best performance with BOW features, but the worst performance with AN50D features. Unlike the four other methods, both GreedyTL and DTL display regular performance on the two kinds of features utilized and improve their performance when using AN50D features. Remarkably, on both metrics, *i.e.*, average AUC and mean AP, DTL outperforms all other methods for both features utilized. Furthermore, in comparison with SRC taken as its baseline, DTL displays a performance gain as high as 25 points in mean AP and 20 points in average AUC when using BOW features, and 14 points in mean AP and 8 points in average AUC for AN50D features. These results thus demonstrate the effectiveness of the proposed method for leveraging existing source data in the transfer learning process for classification of target categories. On the other hand, DTL also displays a large margin in terms of performance gain in comparison

with other state-of-the-art TL methods, *i.e.*, 12 points (5 points, respectively) in mean AP (average AUC, respectively) in comparison with CCTL, the best state-of-the-art TL method when using the BOW feature. Concerning the AN50D features, DTL displays a performance gain of 11 points (2 points, respectively) in mean AP (average AUC, respectively) in comparison with MULTIpLE, the best state-of-the-art TL performer in the Table. This favorable comparison suggests that the proposed DTL, thanks to explicit exploration of similarity and dissimilarity in training, allows better knowledge transfer than state-of-the-art methods in leveraging source data.

In terms of runtime, the average training and test times are also provided in TABLE 3.2. For a fair comparison, we have added to this Table the time for calculating hypotheses, required for hypothesis transfer learning methods, to the training and test times of MULTIpLE and GreedyTL. As expected from the theoretical time complexity analysis in Section 3.4.2, DTL further shows its superiority in comparison, displaying a training time roughly 80 times (20x, respectively) faster, as well as a test time 66 times faster (33x, respectively), than the second fastest TL method in TABLE 3.2 when using the BOW features (AN50D, respectively).

Fig. 3.3 shows the chosen dictionary pairs and their corresponding weights in the DTL models for the 10 target categories after 30 training iterations. As can be observed, in most cases, the target training set is chosen as the dictionary pair with the highest weight (except for the 'courtyard' and 'garbage dump' categories). Apart from the training set, most of the positive dictionaries chosen by DTL are, as expected, source categories quite closely correlated to the target categories, *e.g.*, 'kitchen' for 'biology laboratory', 'bar' for 'bistro indoor'. On the other hand, most of the negative dictionaries chosen by DTL are source categories which, intuitively, are rather negatively correlated to the target ones, *e.g.*, 'playground' and 'gazebo exterior' for 'biology laboratory', 'staircase' and 'construction site' for 'bistro indoor'. The chosen positive dictionary 'mountain snowy' and the target category 'skate park' seem not to be related, but on looking closer can actually be found to be very similar visually.

Table 3.2: Experimental results on the SUN dataset (Results in the first subtable use 500-dimensional BOW features, while results in the second subtable use 50-dimensional AlexNet fc7 layer outputs as features. The experiments are carried out on a server with 100G memory and 4x8-cored AMD OpteronTM CPU 6128 @2GHz. The time unit is the second)

(a) BOW

Methods	Average AUC	Mean AP	Average TrainTime	Average TestTime
SRC	0.6699 ±0.09	0.2858 ±0.12	–	–
AdaBoost	0.7826 ±0.1	0.436 ±0.2	–	–
MULTIpLE	0.5203 ±0.13	0.128 ±0.06	1.11×10^5 ±933.81	2.13×10^3 ±1.42
GreedyTL	0.7926 ±0.1	0.3764 ±0.16	4.39×10^4 ±0.24	1.57×10^4 ±0.001
CCTL	0.8252 ±0.08	0.407 ±0.16	6.6×10^4 $\pm1.9\times10^3$	744.18 ±322.23
DTL	**0.8745** ±0.08	**0.5267** ±0.19	555.15 ±107.27	11.14 ±4.84

(b) AN50D

Methods	Average AUC	Mean AP	Average TrainTime	Average TestTime
SRC	0.8987 ±0.08	0.7256 ±0.19	–	–
AdaBoost	0.833 ±0.09	0.5821 ±0.17	–	–
MULTIpLE	0.9554 ±0.03	0.7574 ±0.15	1.02×10^5 ±113	2.35×10^4 ±0.26
GreedyTL	0.8994 ±0.04	0.5996 ±0.14	9.18×10^4 ±0.46	2.35×10^4 ±0.004
CCTL	0.6339 ±0.11	0.1563 ±0.03	3.4×10^3 ±149.07	102.39 ±17.32
DTL	**0.9745** ±0.02	**0.863** ±0.11	166.26 ±10.5	3.02 ±3.03

3.5.3 Analysis of the Experimental Results

In this subsection we conduct an analysis of the experimental results in light of the theoretical analysis in section 3.3.

3.5.3.1 Effectiveness of the WMW statistic based cost function

In Eq. (3.14) and Eq. (3.15) we defined a Wilcoxon-Mann-Whitney (WMW) statistic based cost function, which aims at maximizing the Area Under the ROC Curve (AUC). In Table 3.3 we show a comparison between this proposed cost function and a mean squared error cost function for selection of dictionary pairs. Specifically, for each target category in the SUN dataset, we choose the positive-negative source dictionary pair to minimize each of the two cost functions. Then, for each pair of chosen source dictionaries, we test their classification performance on the target test samples. For a fair comparison, we use Average Precision (AP) as the performance metric. In Table 3.3, we observe that the average AP for dictionary pairs chosen with the WMW based cost function is higher than the MSE-based one, thus showing the advantage of this proposed cost function. Furthermore, from Table 3.3 we also observe that the dictionary pairs chosen with the mean squared error cost function are not as varied as the dictionary pairs chosen with the WMW based cost function: for example the dictionary pair '+11 -33' appears several times in the left part of the Table, whereas there is no such repetition in the right part of the Table. This result confirms that the WMW based cost function, by maximizing the area under the ROC curve, manages unbalanced data distribution better than the MSE-based cost function, thereby introducing more diversity to the selection of dictionary pairs and better leveraging the existing source data for the target category classification.

3.5.3.2 Effectiveness of DTL in dealing with compact feature vectors

As defined in section 3.3.2, the proposed DTL makes use of sparse representations, namely bi-SRC, as its building blocks. Consequently, we mentioned above that the proposed DTL can handle very compact feature vectors, which lead to high redundancy of the dictionary formed with data samples. Figure 3.4 provides a comparison of DTL with the other 5 baseline methods introduced in section 3.5.2, using 3 sets of features of different lengths, namely AN5D, AN10D and AN50D. The three sets of features are all PCA outputs selected from a 4096-dimensional fc7-layer

Table 3.3: Comparison of the WMW (Wilcoxon-Mann-Whitney) statistic based cost function with the MSE (Mean Squared Error) cost function used as selection criteria for dictionary pairs

Target Category	MSE cost		WMW based cost	
	Dict Pair	Test AP	Dict Pair	Test AP
Biology laboratory	+11 -29	0.1664	+28 -29	**0.2024**
Bistro indoor	+23 -37	0.1369	+23 -20	**0.1951**
Control room	+11 -33	0.1373	+21 -26	**0.1711**
Courtyard	+7 -33	0.0974	+15 -42	**0.1515**
Garbage dump	+1 -29	**0.2573**	+19 -24	0.1039
Parking garage indoor	+11 -33	**0.1332**	+11 -42	0.0851
Rice paddy	+9 -18	0.0916	+14 -30	**0.4285**
Skate park	+7 -33	**0.1107**	+35 -33	0.097
Thrift shop	+11 -33	0.1679	+3 -20	**0.1771**
Veranda	+11 -33	0.1241	+12 -14	**0.2172**
Average		0.1423		**0.1829**

output of an AlexNet fine-tuned on SUN 52 categories. Among them, AN50D is a set of 50-dimensional feature vectors consisting of the 50 most discriminative feature dimensions selected by PCA, AN10D is 10-dimensional with the 10 most discriminative feature dimensions, and AN5D is a set of 5-dimensional features built in like manner. In Figure 3.4 we can see that DTL outperforms all the other methods with the 3 sets of feature vectors, both on average AUC and on mean AP.

3.5.3.3 Effectiveness of using dissimilarities in addition to similarities

A major difference of the proposed DTL compared to other transfer learning methods is that DTL explicitly makes use of negatively correlated source categories as negative dictionaries. By using negative dictionaries in addition to positive dictionaries, DTL thus makes use of both similarities and dissimilarities to perform discriminative classification. The benefit of using dissimilarities in addition to similarities is highlighted in Table 3.4. The two subtables show performance using two different performance metrics: average AUC and mean AP. In each Table, the 'Pos' column shows the prediction performance of DTL using only positive dictionaries (*i.e.*, using reconstruction residuals on positive dictionaries as discriminant). The

'Pos+Neg' column shows the prediction performance of DTL using both positive and negative dictionaries (*i.e.*, using the difference in reconstruction residuals on each dictionary pair as discriminant). As can be observed, by introducing negative dictionaries in addition to positive ones, DTL achieves a significant performance gain of as much as 32 points (22 points, respectively) in mean AP (average AUC) when the BOW features are utilized. In the case of AN50D features, which display the best performance, performance gain is 12 points in mean AP and 4 points in average AUC. These results thus consistently support our claim that joint use of similarity and dissimilarity enhances the discrimination power of transfer learning algorithm. This tallies with the findings of psychophysicists [Stewart & Brown 2005] in their studies on the importance of similarity and dissimilarity in human visual categorization.

Table 3.4: Benefits of using both positive and negative dictionaries. The Table on the left shows the results using average AUC as the performance metric, while the Table on the right shows the results using mean AP as the performance metric. In both Tables, the 'Pos' columns show DTL performance when using only positive dictionaries, while the 'Pos+Neg' columns show DTL performance when using both positive and negative dictionaries. The experiments are conducted on the SUN dataset.

(a) Average AUC

Features	Pos	Pos+Neg
AN5D	0.8657	0.9116
AN10D	0.8535	0.9236
AN50D	0.9393	0.9745
BOW	0.6597	0.8745

(b) Mean AP

Features	Pos	Pos+Neg
AN5D	0.5352	0.6294
AN10D	0.5516	0.6874
AN50D	0.7464	0.863
BOW	0.1994	0.5267

3.6 Conclusion

In this chapter, we have introduced DTL, a novel sparse representation based discriminative transfer learning algorithm, for learning target categories with only a few training samples by transferring knowledge from source categories that are different from target categories. Unlike other transfer learning algorithms, DTL explicitly makes use of both positively and negatively correlated source categories to

help classification of the target category. Moreover, we introduce a novel Wilcoxon-Mann-Whitney based cost function to optimize the Area Under the ROC Curve (AUC), thereby enabling the proposed TL algorithm to deal with the unbalanced nature of data distribution. We further enhance DTL performance by running 2 AdaBoost processes concurrently on both the positive and negative distributions of the target training set. Experimental results on both the NUS-WIDE Scene and SUN datasets show that DTL consistently displays the best classification performance compared to other state-of-the-art transfer learning algorithms, when using different features and two different performance metrics. Time complexity analysis and experimental results both show the superiority of DTL compared to other boosting-based transfer learning algorithms: a training runtime roughly 80 times (20x, respectively) faster, as well as a testing runtime 66 times faster (33x, respectively), than the other state-of-the-art TL methods compared when using the BOW features (AN50D, respectively).

Future work includes the extension of the proposed binary DTL algorithm to solve multi-class classification problems. Another future direction could be to add a dictionary learning step to the DTL method to further enhance its discriminative power.

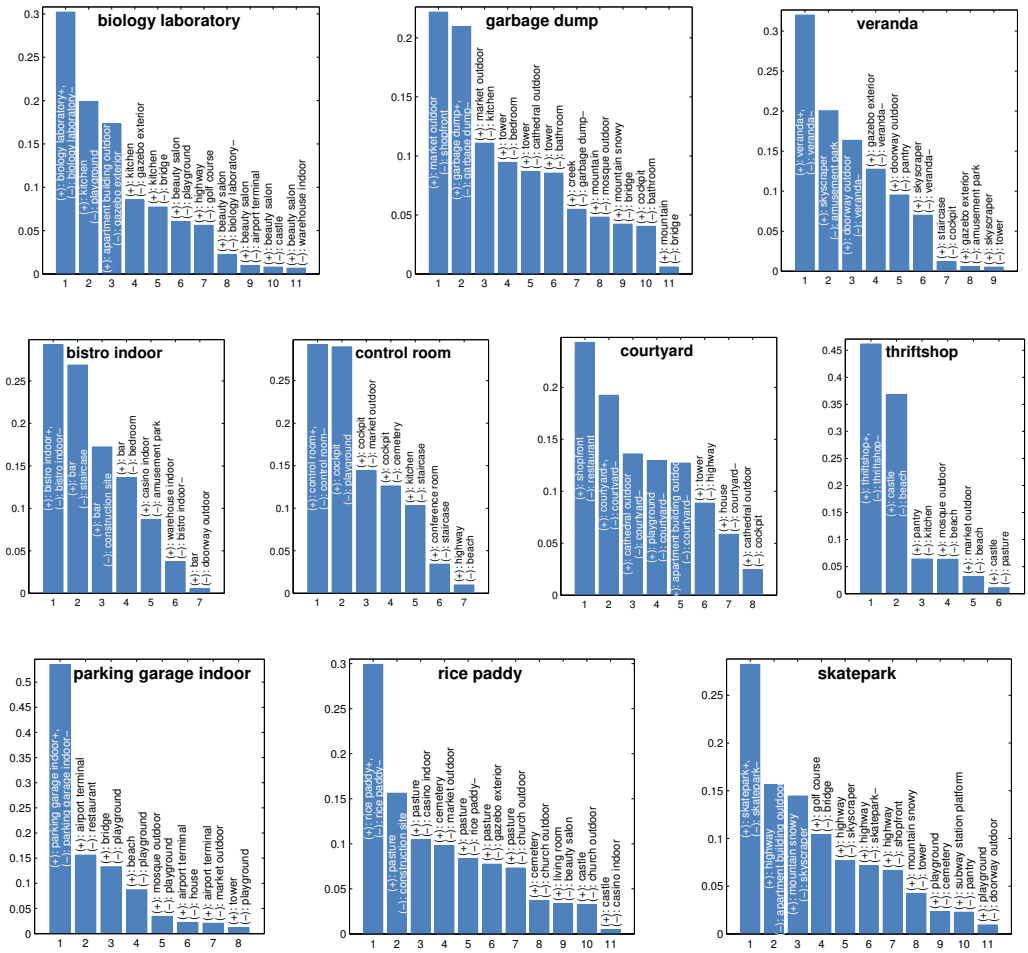

Figure 3.3: Illustration of selected dictionary pairs for each classification model on the SUN dataset with BOW features. In each subfigure, the name of the target category is displayed on the top space of the Figure: the horizontal axis shows the number of dictionary pairs, and the vertical axis the weight for each dictionary pair; the chosen dictionary pairs are presented in the descending order of their weights, while above or in the bars are the names of the selected source categories which form the dictionaries: (+) stands for positive dictionary and (-) stands for negative dictionary.

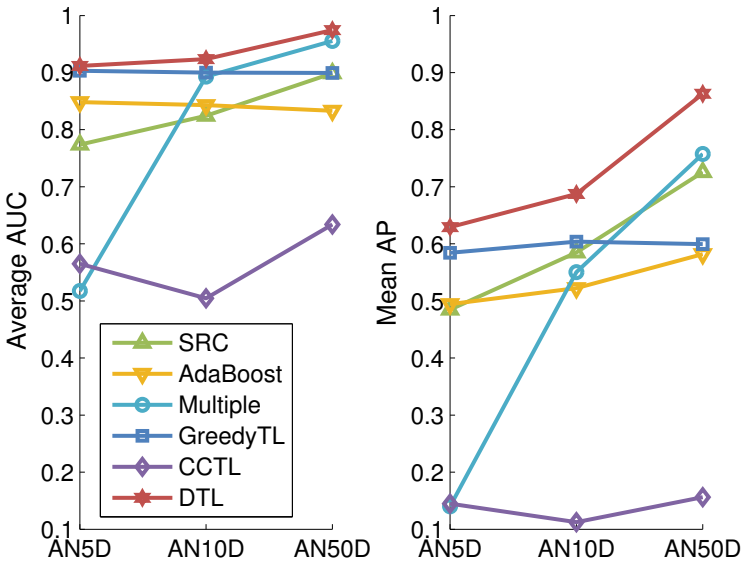

Figure 3.4: Comparison of DTL with other methods on compact features. The graph on the left shows the result using average AUC as the performance metric, while the graph on the right shows the result using mean AP as the performance metric. In both graphs, the horizontal axis shows 3 different features: AN5D a set of 5-dimensional features, AN10D a set of 10-dimensional features and AN50D a set of 50-dimensional features. The experiments are conducted on the SUN dataset.

Chapter 4

Optimal Transport for Deep Joint Transfer Learning

4.1 Introduction

Supervised machine learning generally requires a large amount of labeled training data for an effective training of the underlying prediction model, especially when the prediction model is complex, *e.g.*, Deep Neural Networks (DNN), where the number of parameters is at a scale of thousand millions. However in practice, collecting a sufficient number of manually labeled training samples may prove tedious, time consuming, even impractical, and therefore prohibitive, *e.g.*, object edge detection, medical image segmentation, where a pixel-wise ground truth is needed. This is all the more true when the task is novel or rare. For example, for a fine-grained image classification task, for some rare categories we can only gather very limited number of image samples. Transfer Learning (TL) aims to leverage existing related source domain data for an informed knowledge transfer to a target task and thereby solve or mitigate this kind of "data starvation" problem to help the learning of a target task. As such, TL has received an increasing interest from several research communities [Pan & Yang 2010b] [Shao *et al.* 2015].

In this work we also consider the Inductive Transfer Learning (ITL) problem [Pan & Yang 2010b], which aims at learning an effective classification model for some target categories with few training samples, by leveraging knowledge from different but related source categories with far more training samples. Given the breakthrough of Deep Neural Networks (DNNs) in an increasing number of applications, a natural yet simple solution to this problem consists of fine-tuning a DNN

which is pre-learned on some related source data for a given target classification task [Yosinski *et al.* 2014]. However, although this fine-tuning process can inherit or preserve the knowledge learned during pre-training on source data, prior knowledge about the relatedness between source and target tasks is not explicitly explored. As a result, such a fine-tuning process may fall short to achieve an effective adaptation of a pre-trained DNN for a given target task , especially when the latter has very few labeled data.

A recent move is [Ge & Yu 2017] on selective joint fine-tuning, which tackles this problem by first selecting relevant samples in the source domain, then performing a joint fine-tuning on target training data and the selected source data. Although the selective step ensures the fine-tuning process to use only source samples which are related to a given target domain, in the joint fine-tuning step the source classifier and target classifier are still trained as two different classifiers.

In this work, we propose to explicitly account for the relatedness between source and target tasks and explore such prior knowledge through the design of a novel loss function, namely Optimal Transport loss (OT loss), which is minimized during joint training of the underlying neural network, in order to bridge the gap between the source and target classifiers. This results in a Joint Transfer Learning Network (JTLN). This JTLN can be built upon common Deep Neural Network structure. In JTLN, the source data and target data go trough same feature extraction layers simultaneously, and then separate into two different classification layers. The Optimal Transport loss is added between the two classification layers' outputs, in order to minimize the distance between two classifiers' predictions.

As the Optimal Transport loss is calculated with a pre-defined cost matrix, this JTLN can therefore incorporates different prior knowledge about the relations between source and target tasks by using different kind of cost metric. In this work, we show two examples of using the distance between category distributions as cost metric.

The contributions of this work are threefold:

1. We propose a Joint Transfer Learning framework built upon existing Deep

Neural Networks for Inductive Transfer Learning.

2. We extend the Wasserstein loss proposed in [Frogner *et al.* 2015] to a more general Optimal Transport loss for comparing probability measures with different length, and use it as a soft penalty in our JTLN.

3. We show two different ways of using the distance between category distributions as cost metric for OT loss. Experimental results on two ITL image classification datasets show that JTLN with these two cost metrics can achieve better performance than consecutive fine-tuning or simple joint fine-tuning without extra constraint.

4.2 Related work

A related problem in transfer learning (TL) is the Domain Adaptation (DA) problem, which is Transductive TL [Pan & Yang 2010b] and assumes that the source domain and target domain share the same label space, while following different probability distributions. Optimal Transport has already been successfully applied to DA in [Courty *et al.* 2016]. Recently several deep joint learning methods have been proposed to solve this problem. For example in [Long *et al.* 2015] the authors propose to add multiple adaptation layers upon deep convolutional networks. Through these adaptation layers the mean embeddings of source distribution and target distribution are matched, therefore encouraging the network to learn a shared feature space for source domain and target domain. In [Long *et al.* 2016] the authors extend the previous work by adding additional residual layers to adjust classifier mismatch between source domain and target domain. Although these methods work well for domain adaptation problems, their assumption that the source domain and target domain share a same label space and have a limited distribution discrepancy restrict the possibility of applying these methods for Inductive Transfer Learning.

Until recently most state-of-the-art ITL methods are based on shallow machine learning models [Shao *et al.* 2015] [Kuzborskij *et al.* 2013] [Kuzborskij *et al.* 2015] [Li *et al.* 2017]. For example in [Kuzborskij *et al.* 2015] the authors propose to

select relevant source hypotheses and feature dimensions through greedy subset selection. In [Li *et al.* 2017] the authors propose to learn a high quality dictionary for low-rank coding across source domain and target domain for self-taught learning (which is ITL with only unlabeled samples in source domain). To the best of our knowledge, the proposed JTLN is the first work to tackle ITL with Deep Neural Networks and optimal transport theory.

The proposed JTLN has been inspired by a recent work on wasserstein loss [Frogner *et al.* 2015]. Frogner et al. proposed a wasserstein loss as a soft penalty for multi-label prediction. Although their wasserstein loss is calculated between predicted label vector and ground-truth label vector with the same length, the matrix scaling process used to calculate this wasserstein loss is actually not restricted to square transportation matrix. In this work, we extend this wassertein loss to a more general Optimal Transport loss for label vectors with different length. As a result, the proposed JTLN enables the exploration of prior knowledge through the initial cost matrix and makes use of the OT loss as a soft penalty for bridging the gap between target and source classifier predictions.

4.3 Joint Transfer Learning Network

4.3.1 Problem definition and the JTLN structure

Assume that we have a small target training set $\mathcal{T} = \{(\mathbf{x}_i^t, y_i^t)\}_{i=1}^{n_t}$ of n_t training samples, with $\mathbf{x}_i^t \in \mathcal{X}_t,\ y_i^t \in \mathcal{L}_t$ and $\mathcal{L}_t = \{l_i^t\}_{i=1}^{L_t}$ is the target label set. In Inductive Transfer Learning we are also given a larger source set $\mathcal{S} = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ of n_s samples, with $\mathbf{x}_i^s \in \mathcal{X}_s,\ y_i^s \in \mathcal{L}_s$ and $\mathcal{L}_s = \{l_i^s\}_{i=1}^{L_s}$ is the source label set. (No specific assumption is made for $\mathcal{X}_t$ and $\mathcal{X}_s$, meaning they can either be equal or not equal.) We assume that $\mathcal{L}_s \neq \mathcal{L}_t$, this means that the target samples and source samples are from different concept categories. We also assume that a cost metric $c(\cdot,\cdot) : \mathcal{L}_s \times \mathcal{L}_t \to \mathbb{R}$ could be found, which indicates the relationships between each pair of source category and target category (We will show in section 4.3.4 two examples on defining this cost metric).

We build the Joint Transfer Learning Network upon common Deep Neural Net-

works (*e.g.* Alexnet for image classification), an illustration of a JTLN built upon Alexnet can be found in Figure 4.1. In JTLN the feature extraction layers are shared by source data and target data and give $f(\mathbf{x}_i)$ as the feature vector for input sample $\mathbf{x}_i$. Following are two fully-connected layers with different output dimensions, which are considered as the source classifier and target classifier. The output of the source classifier is noted as: $h_s(\mathbf{x}_i) = a(\mathbf{W}_s \cdot f(\mathbf{x}_i) + \mathbf{b}_s)$, where $a(\cdot)$ is the softmax activation function, $\mathbf{W}_s$ and $\mathbf{b}_s$ are layer weight and bias for source classifier. The output of the target classifier is noted similarly: $h_t(\mathbf{x}_i) = a(\mathbf{W}_t \cdot f(\mathbf{x}_i) + \mathbf{b}_t)$, with $\mathbf{W}_t$ and $\mathbf{b}_t$ the layer weight and bias for target classifier. Two cross-entropy losses are added for joint learning with source data and target data. The source cross-entropy loss term is defined as:

$$\frac{1}{n_s} \sum_{i=1}^{n_s} \ell_{ce}(h_s(\mathbf{x}_i^s), y_i^s) \tag{4.1}$$

where $\ell_{ce}(\cdot, \cdot)$ is the cross-entropy loss function. The target cross-entropy loss term is defined similarly:

$$\frac{1}{n_t} \sum_{i=1}^{n_t} \ell_{ce}(h_t(\mathbf{x}_i^t), y_i^t) \tag{4.2}$$

To express our prior knowledge about the relatedness between source and target tasks, we propose to add a third Optimal Transport loss term for target data to restrict the distance between source classifier output and target classifier output, the OT loss term is noted as:

$$\frac{1}{n_t} \sum_{i=1}^{n_t} \ell_{ot}(h_s(\mathbf{x}_i^t), h_t(\mathbf{x}_i^t)) \tag{4.3}$$

where $\ell_{ot}(\cdot, \cdot)$ is the OT loss which will be defined in section 4.3.2.

Therefore training with JTLN is a problem of minimizing the empirical risk which is a combination of the three loss terms shown above:

$$\min_{\Theta} \frac{1}{n_t} \sum_{i=1}^{n_t} \ell_{ce}(h_t(\mathbf{x}_i^t), y_i^t) + \frac{\lambda_s}{n_s} \sum_{i=1}^{n_s} \ell_{ce}(h_s(\mathbf{x}_i^s), y_i^s) + \frac{\lambda_{ot}}{n_t} \sum_{i=1}^{n_t} \ell_{ot}(h_s(\mathbf{x}_i^t), h_t(\mathbf{x}_i^t)) \tag{4.4}$$

where Θ denote the set of all parameters in JTLN , λ_s is the loss weight for

source cross-entropy loss and λ_{ot} is the loss weight for OT loss.

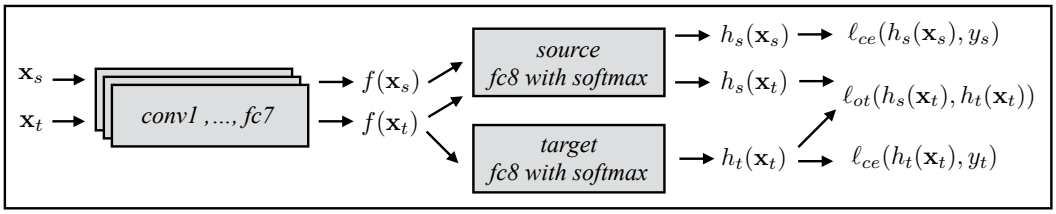

Figure 4.1: The structure and data flow of a Joint Transfer Learning Network based on Alexnet

4.3.2 Optimal Transport Loss

In this work we consider the discrete optimal transport problem. As the output of the source classifier (*i.e.* $h_s(\mathbf{x}_i)$) and that of the target classifier (*i.e.* $h_t(\mathbf{x}_i)$) are outputs of softmax activation, meaning that $\sum_{j=1}^{L_s}(h_s(\mathbf{x}_i))_j = 1$ and $\sum_{j=1}^{L_t}(h_t(\mathbf{x}_i))_j = 1$. We can therefore consider them as two probability measures over their corresponding label space. We define:

$$\begin{aligned} \mu &= h_s(\mathbf{x}_i) \in \mathcal{Y}_s \\ \nu &= h_t(\mathbf{x}_i) \in \mathcal{Y}_t \end{aligned} \tag{4.5}$$

where $\mathcal{Y}_s = \mathbb{R}_+^{L_s}$ is the space of measures over the source label set $\mathcal{L}_s$ and $\mathcal{Y}_t = \mathbb{R}_+^{L_t}$ is the space of measures over the target label set $\mathcal{L}_t$. Note that μ and ν defined here are discrete probability measures, *i.e.* histograms in the probability simplex $\Delta^{\mathcal{L}_s}$ and $\Delta^{\mathcal{L}_t}$.

Given a cost metric $c(\cdot,\cdot) : \mathcal{L}_s \times \mathcal{L}_t \to \mathbb{R}$, the optimal transport problem aims at finding the optimal transportation plan γ_0 which minimizes the cost to transport the mass in probability measure μ to match that in ν. The Kantorovich formulation [Kantorovich 2006] of this discrete optimal transport problem can be defined as follows:

$$\gamma_0 = \arg\min_{\gamma \in \Pi(\mu,\nu)} \langle \gamma, \mathbf{C} \rangle_F$$
$$\Pi(\mu,\nu) = \{\gamma \in (\mathbb{R}_+)^{L_s \times L_t} \mid \gamma \mathbf{1}_{L_t} = \mu,\ \gamma^\top \mathbf{1}_{L_s} = \nu\} \tag{4.6}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius dot product. $\mathbf{C}$ is the cost matrix with $\mathbf{C}_{i,j} = c(l_i^s, l_j^t)$, the cost term $\mathbf{C}_{i,j}$, which can be interpreted as the cost to move a probability mass from l_i^s to l_j^t (In our case we can think it as the cost to transfer the prediction on source category l_i^s to the prediction on target category l_j^t). $\Pi(\mu,\nu)$ is the set of all valid transportation plans, *i.e.* the set of joint probability measures on $\mathcal{L}_s \times \mathcal{L}_t$ with μ and ν as marginals. $\mathbf{1}_d$ is a d-dimensional vector of ones.

If $\mathcal{L}_s = \mathcal{L}_t$, the wasserstein distance can be defined and in [Frogner *et al.* 2015] the authors use the wasserstein distance formulation as a loss function for their multi-label prediction problem. In our case, we have assumed $\mathcal{L}_s \neq \mathcal{L}_t$, we therefore define our loss function in a similar way directly based on the Optimal Transport formulation:

Definition 2. (Optimal Transport Loss) *For any source classifier $h_s : \mathcal{X} \to \Delta^{\mathcal{L}_s}$, and any target classifier $h_t : \mathcal{X} \to \Delta^{\mathcal{L}_t}$, given input $\mathbf{x} \in \mathcal{X}$, and a cost metric $c(\cdot,\cdot) : \mathcal{L}_s \times \mathcal{L}_t \to \mathbb{R}$, the Optimal Transport Loss is defined as:*

$$\ell_{ot}(h_s(\mathbf{x}), h_t(\mathbf{x})) \triangleq \inf_{\gamma \in \Pi(h_s(\mathbf{x}), h_t(\mathbf{x}))} \langle \gamma, \mathbf{C} \rangle_F \tag{4.7}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius dot product. $\mathbf{C}$ is the cost matrix with $\mathbf{C}_{i,j} = c(l_i^s, l_j^t)$. $\Pi(h_s(\mathbf{x}), h_t(\mathbf{x}))$ is the set of valid transportation plans defined as:

$$\Pi(h_s(\mathbf{x}), h_t(\mathbf{x})) = \{\gamma \in (\mathbb{R}_+)^{L_s \times L_t} \mid \gamma \mathbf{1}_{L_t} = h_s(\mathbf{x}),\ \gamma^\top \mathbf{1}_{L_s} = h_t(\mathbf{x})\} \tag{4.8}$$

where $\mathbf{1}_d$ is a d-dimensional vector of ones.

This Optimal Transport Loss in Definition 2 can therefore be calculated by solving the discrete optimal transport problem shown in (4.6). Problem in (4.6) is a linear programming problem and can be solved with combinatorial algorithms,

e.g. the simplex methods and its network variants [Courty *et al.* 2016]. However, the computational complexity is shown to be $O((L_s + L_t)L_s L_t log(L_s + L_t))$ at best [Ahuja *et al.* 1993]. This limits the usage of this formulation for large scale dataset. Recently, Cuturi et al. [Cuturi 2013] [Benamou *et al.* 2015] proposed an entropy regularized optimal transport problem, which can be efficiently solved by iterative Bregman Projections. The discrete optimal transport problem with entropy regularization can be defined as:

$$\begin{aligned} \gamma_0 &= \underset{\gamma \in \Pi(\mu,\nu)}{\arg\min} \langle \gamma, \mathbf{C} \rangle_F - \frac{1}{\lambda} H(\gamma) \\ \Pi(\mu,\nu) &= \{\gamma \in (\mathbb{R}_+)^{L_s \times L_t} \mid \gamma \mathbf{1}_{L_t} = \mu,\ \gamma^\top \mathbf{1}_{L_s} = \nu\} \\ H(\gamma) &= \sum_{i,j} \gamma_{i,j} \log \gamma_{i,j} \end{aligned} \tag{4.9}$$

where $H(\gamma)$ is the entropy of γ and $-\frac{1}{\lambda}$ is the regularization weight.

This entropy regularization forces the solution of (4.9) to be smoother as $\frac{1}{\lambda}$ increase, *i.e.*, as $\frac{1}{\lambda}$ increases, the sparsity of γ_0^λ decreases. This non-sparsity of the solution helps to stabilize the computation by making the problem strongly convex with a unique solution. The advantage of this entropic regularized OT problem is that its solution is a diagonal scaling of $e^{-\lambda \mathbf{C} - 1}$, where $e^{-\lambda \mathbf{C} - 1}$ is the element-wise exponential matrix of $-\lambda \mathbf{C} - 1$. The solution to this diagonal scaling problem can be found by iterative Bregman projections [Benamou *et al.* 2015].

In this work we calculate the approximation to the Optimal Transport Loss in Definition 2 by solving the entropic regularized optimal transport problem defined in (4.9) using iterative Bregman projections as shown in [Benamou *et al.* 2015]. The computation of this approximate OT Loss is defined in Algorithm 2, where ./ means element-wise division.

4.3.3 Back-propagation with OT Loss

The empirical risk minimization problem defined in Equation 4.4 is normally solved by a gradient descent algorithm, therefore the gradient of each loss term with respect to their corresponding inputs should be expressed analytically for back-propagation.

Algorithm 2 Computation of the approximate OT Loss

Input: $h_s(\mathbf{x}) \in \Delta^{\mathcal{L}_s}$, $h_t(\mathbf{x}) \in \Delta^{\mathcal{L}_t}$, λ, $\mathbf{C}$

1:
2: Initialize: $\mathbf{u} = \mathbf{1}_{L_s}/L_s$, $\mathbf{v} = \mathbf{1}_{L_t}/L_t$, $\mathbf{K} = e^{-\lambda \mathbf{C} - 1}$
3: **while** $\mathbf{u}$ has not converged **do**
4: $\quad \mathbf{v} = h_t(\mathbf{x})./(\mathbf{K}^\top \mathbf{u})$
5: $\quad \mathbf{u} = h_s(\mathbf{x})./(\mathbf{K}\mathbf{v})$
6: **end while**
7: $\ell_{ot}(h_s(\mathbf{x}), h_t(\mathbf{x})) = \langle diag(\mathbf{u}) \cdot \mathbf{K} \cdot diag(\mathbf{v}), \mathbf{C} \rangle_F$

As in [Frogner *et al.* 2015] we define the Lagrange dual problem of LP problem 4.7 as :

$$\begin{aligned} \ell_{ot}(h_s(\mathbf{x}), h_t(\mathbf{x})) &= \sup_{\alpha,\beta \in \mathcal{C}} \alpha^\top h_s(\mathbf{x}) + \beta^\top h_t(\mathbf{x}) \\ \mathcal{C} &= \{(\alpha, \beta) \in \mathbb{R}^{L_s \times L_t} : \alpha_i + \beta_j \leq \mathbf{C}_{i,j}\} \end{aligned} \tag{4.10}$$

The $\mathbf{u}$ and $\mathbf{v}$ defined in Algorithm 2 can be expressed as: $\mathbf{u} = e^{\lambda\alpha}$ and $\mathbf{v} = e^{\lambda\beta}$. As 4.7 is a linear program, at an optimum the values of the dual and the primal are equal, therefore the dual optimal α is a sub-gradient of the OT loss with respect to $h_s(\mathbf{x})$ and β is a sub-gradient of the OT loss with respect to $h_t(\mathbf{x})$.

The gradient of OT loss with respect to its two arguments can therefore be expressed as follows and can be easily computed with the optimal scaling vectors $\mathbf{u}$ and $\mathbf{v}$ after matrix scaling with Algorithm 2:

$$\begin{aligned} \frac{\partial \ell_{ot}(h_s(\mathbf{x}), h_t(\mathbf{x}))}{\partial h_s(\mathbf{x})} &= \alpha = \frac{\log \mathbf{u}}{\lambda} - \frac{\log \mathbf{u}^\top \mathbf{1}_{L_s}}{\lambda L_s} \mathbf{1}_{L_s} \\ \frac{\partial \ell_{ot}(h_s(\mathbf{x}), h_t(\mathbf{x}))}{\partial h_t(\mathbf{x})} &= \beta = \frac{\log \mathbf{v}}{\lambda} - \frac{\log \mathbf{v}^\top \mathbf{1}_{L_t}}{\lambda L_t} \mathbf{1}_{L_t} \end{aligned} \tag{4.11}$$

Note that α and β are defined up to a constant shift, *i.e.* any upscaling of the vector $\mathbf{u}$ can be paired with a corresponding downscaling of the vector $\mathbf{v}$ (and vice versa) without altering the matrix γ_0, therefore the second terms in Equation 4.11 are added to ensure that α and β are tangent to their corresponding simplex.

4.3.4 Choosing the cost metric

In Definition 2, the cost metric $c(\cdot,\cdot)$ can be interpreted as the cost to transfer the prediction on a source category to that of a target category. This cost metric embodies prior knowledge which describes the relatedness between each pair of source and target categories. The choice of this cost metric is crucial in JTLN for having a better joint learning performance.

A reasonable choice is to consider that the sample features in each category follow a probability distribution in a joint feature space, and to define this cost metric as the distance between two distributions. For example given a source category l_i^s and a target category l_j^t, and a feature extractor $f(\mathbf{x})$ for sample $\mathbf{x}$. Suppose $\{f(\mathbf{x}^s) \mid \forall(\mathbf{x}^s, y^s),\ y^s = l_i^s\}$ follows the distribution μ_s, and $\{f(\mathbf{x}^t) \mid \forall(\mathbf{x}^t, y^t),\ y^t = l_j^t\}$ follows the distribution μ_t, our goal is to define a distance $d(\mu_s, \mu_t)$ between the two distributions as the cost metric for OT loss: $c(l_i^s, l_j^t) = d(\mu_s, \mu_t)$. To simplify the notations, in the following of this section we will use $\mathbf{x}^s$ and $\mathbf{x}^t$ instead of $f(\mathbf{x}^s)$ and $f(\mathbf{x}^t)$ to represent samples from the two distributions.

This definition implies that if the distribution of a target category and that of a source category lie close to each other in the feature space, their corresponding labels are more probable related and therefore cost less effort to transfer the prediction of one to that of the other.

There are various ways to calculate the distance between two distributions. One way is to use the two-sample test with Multi-Kernel Maximum Mean Discrepancy (MK-MMD) as test statistics, which is successfully applied for solving domain adaptation problems [Long *et al.* 2015]. Another way is to use Optimal Transport and employ a basic distance metric (*e.g.* Euclidean distance) as the cost metric. In the following we show details on how to apply these two methods as cost metrics for evaluating the distance between a given pair of source and target categories.

4.3.4.1 MK-MMD as cost metric

Given samples from two distributions μ_s and μ_t, a two-sample test determines whether to reject the null hypothesis $H_0 : \mu_s = \mu_t$, based on the value of a test

statistics measuring the distance between the samples. One choice of the test statistics is the maximum mean discrepancy (MMD), which is a distance between embeddings of the probability distributions in a reproducing kernel Hilbert space. Here we make use of the multi-kernel variant of MMD (MK-MMD) proposed in [Gretton *et al.* 2012], which maximizes the two-sample test power and minimizes the Type II error (*i.e.*, the probability of wrongly accepting H_0 when $\mu_s \neq \mu_t$), given an upper bound on Type I error (*i.e.*, the probability of wrongly rejecting H_0 when $\mu_s \neq \mu_t$), by leveraging different kernels for kernel embeddings.

Let $\mathcal{H}_k$ be a reproducing kernel Hilbert space (RKHS) endowed with a characteristic kernel k. The mean embedding of distribution μ_s in $\mathcal{H}_k$ is a unique element $\phi_k(\mu_s) \in \mathcal{H}_k$ such that $\mathbf{E}_{\mathbf{x}\sim\mu_s} g(\mathbf{x}) = \langle g(\mathbf{x}), \phi_k(\mu_s)\rangle_{\mathcal{H}_k}$ for all $g \in \mathcal{H}_k$. The MMD between probability distributions μ_s and μ_t is defined as the RKHS distance between the mean embeddings of μ_s and μ_t. The squared formulation of MMD can be defined as:

$$d_k^2(\mu_s, \mu_t) = \| \phi_k(\mu_s) - \phi_k(\mu_t) \|_{\mathcal{H}_k}^2 = \mathbf{E}_{\mathbf{x}^s\mathbf{x}^{s\prime}} k(\mathbf{x}^s, \mathbf{x}^{s\prime}) + \mathbf{E}_{\mathbf{x}^t\mathbf{x}^{t\prime}} k(\mathbf{x}^t, \mathbf{x}^{t\prime}) - 2\mathbf{E}_{\mathbf{x}^s\mathbf{x}^t} k(\mathbf{x}^s, \mathbf{x}^t) \tag{4.12}$$

where $\mathbf{x}^s, \mathbf{x}^{s\prime} \overset{\text{i.i.d.}}{\sim} \mu_s$ and $\mathbf{x}^t, \mathbf{x}^{t\prime} \overset{\text{i.i.d.}}{\sim} \mu_t$. With ϕ_k an injective map, *i.e.* k is a characteristic kernel, the MMD is a metric on the space of Borel probability measures, *i.e.* $d_k(\mu_s, \mu_t) = 0$ if and only if $\mu_s = \mu_t$. The characteristic kernel k is defined as the convex combination of m PSD (positive semi-definite) kernels k_u:

$$\mathcal{K} \triangleq \{k = \sum_{u=1}^{m} \beta_u k_u | \sum_{u=1}^{m} \beta_u = 1, \beta_u \geqslant 0, \forall u\} \tag{4.13}$$

where the constraints on coefficients $\{\beta_u\}$ are imposed to guarantee that the derived multi-kernel k is characteristic. This multi-kernel k can leverage different kernels to enhance the power of two-sample test.

For computation efficiency, we adopt the unbiased estimate of MK-MMD which can be computed with linear complexity:

$$d_k^2(\mu_s, \mu_t) = \frac{2}{n} \sum_{i=1}^{n/2} g_k(\mathbf{z}_i) \tag{4.14}$$

where $\mathbf{z}_i \triangleq (\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s, \mathbf{x}_{2i-1}^t, \mathbf{x}_{2i}^t)$ is a random quad-tuple sampled from μ_s and μ_t , and we evaluate each quad-tuple with $g_k(\mathbf{z}_i) \triangleq k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s) + k(\mathbf{x}_{2i-1}^t, \mathbf{x}_{2i}^t) - k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^t) - k(\mathbf{x}_{2i}^s, \mathbf{x}_{2i-1}^t)$.

4.3.4.2 OT as cost metric

Consider μ_s and μ_t as two empirical distributions defined by their corresponding discrete samples:

$$\mu_s = \sum_{i=1}^{n_s} p_i^s \delta_{\mathbf{x}_i^s}, \quad \mu_t = \sum_{i=1}^{n_t} p_i^t \delta_{\mathbf{x}_i^t} \tag{4.15}$$

where $\delta_{\mathbf{x}_i}$ is the Dirac function at location $\mathbf{x}_i$. $p_i^s \in \Delta^{n_s}$ and $p_i^t \in \Delta^{n_t}$ are probability masses associated to the i-th sample. We can therefore define a discrete optimal transport problem with entropy regularization as in equation (4.9):

$$\begin{aligned} \gamma_0 &= \underset{\gamma \in \Pi(\mu_s, \mu_t)}{\arg\min} \langle \gamma, \mathbf{C} \rangle_F - \frac{1}{\lambda} H(\gamma) \\ \Pi(\mu_s, \mu_t) &= \{\gamma \in (\mathbb{R}_+)^{n_s \times n_t} \mid \gamma \mathbf{1}_{n_t} = \mu_s,\ \gamma^\top \mathbf{1}_{n_s} = \mu_t\} \\ H(\gamma) &= \sum_{i,j} \gamma_{i,j} \log \gamma_{i,j} \end{aligned} \tag{4.16}$$

We define the cost metric in Equation (4.16) as squared Ecuclidean distance between two samples $\mathbf{C}_{i,j} = \| \mathbf{x}_i^s - \mathbf{x}_j^t \|_2^2$, and define the distance between the two distributions as $d(\mu_s, \mu_t) = \langle \gamma_0, \mathbf{C} \rangle_F$. This distance can be computed using the same matrix scaling procedure as in Algorithm 2.

4.4 Experiments

In this section we show the experiments of our proposed JTLN built upon Alexnet for Inductive Transfer Learning (ITL) with fine-grained image classification

datasets.

Note that all experiments perform fine-tuning based on an Alexnet model pre-trained with the ImageNet database. Further to the recommendations in [Yosinski *et al.* 2014], we fix the first three convolution layers since the features learned by these layers are general for different tasks, fine-tune the 4-th and 5-th convolutional layers with a small learning rate because the features learned in these two layers are less general, and fine-tune the 6-th and 7-th fully connected layers with a larger learning rate because the features learned in these layers are more task specific. The classification layers (fc8 with softmax) are trained from scratch.

We make use of two different image datasets, the first one is the FGVC-Aircraft Dataset [Maji *et al.* 2013] and the second one is the Office-31 dataset[1].

4.4.1 Experiments on FGVC-Aircraft Dataset

The FGVC-Aircraft dataset contains 10000 images of aircraft, with 100 images for each of 100 different aircraft model variants. These aircraft variants are from 30 different manufacturers. We build our Inductive Transfer Learning datasets upon this dataset by choosing the model variants from one manufacturer as the target domain categories, and consider the rest of the model variants as source domain categories. The images in source categories are all used for JTLN training, while the images in target categories are split into a subset for training and a subset for testing. We choose the two manufacturers with the most model variants to form two different ITL datasets, the characteristics of these ITL datasets are listed in Table 4.1, where the dataset name is indexed by the target manufacturer name.

We compare our proposed JTLN with three baseline methods: the first one consists of fine-tuning only with target training samples; the second one is the commonly adopted method, which first fine-tunes the pre-trained model with source samples, then continues fine-tuning with target training samples; the third baseline performs fine-tuning jointly with source samples and target training samples without applying the OT loss.

We also show the results of two variants of the proposed JTLN: (1) JTLN

[1] https://people.eecs.berkeley.edu/~jhoffman/domainadapt/#datasets_code

(fc7MKMMD) is JTLN using MK-MMD as cost metric as shown in section 4.3.4.1, and using the fc7-layer output of the Alexnet model pre-trained on ImageNet as features for the computation of the MK-MMD distances. (2) JTLN (fc7OT) is JTLN using OT as cost metric as shown in section 4.3.4.2, using the same Alexnet fc7-layer output as features.

Table 4.1: ITL Datasets with FGVC-Aircraft images

Dataset properties	Boeing	Airbus
Number of target categories	22	13
Number of target training images	1466	867
Number of target testing images	734	433
Number of source categories	78	87
Number of source images	7800	8700

Table 4.2: Experimental Results on the ITL Datasets (results are multi-class classification accuracy)

Methods	Boeing	Airbus
Finetuning on target	0.4796	0.4965
Consecutive finetuning on source+target	0.5286	0.545
Joint finetuning on source+target	0.5395	0.5497
JTLN (fc7MKMMD)	0.5422	**0.5982**
JTLN (fc7OT)	**0.5436**	0.5704

The classification accuracies of these methods for the two ITL datasets are shown in table 4.2. We can see that with fc7MKMMD as cost metric, JTLN for ITL-Airbus successfully improved the performance of joint fine-tuning by 5 points. JTLNs (fc7MKMMD and fc7OT) on ITL-Boeing also improved in comparison with joint fine-tuning. However, the performance increase is not as high as that with the ITL-Airbus dataset. We believe this can be partially explained by the fact that ITL-Boeing has less source categories and less source samples than ITL-Airbus.

4.4.2 Experiments on Office-31 Dataset

The Office-31 contains 3 domains: Amazon, Webcam, and Dslr. Each domain contains images from amazon.com, or office environment images taken with varying lighting and pose changes using a webcam or a dslr camera, respectively. All three domains in this dataset contain images of the same 31 categories.

Given two domains, one as the source domain and the other one as the target domain, our goal is to lean a classifier for target domain with all images in the source domain and a small amount of images in the target domain, the remaining images in the target domain are used as test samples. Since the 31 categories are the same in different domain, we can therefore use a simple cost matrix with zeros in diagonal and nonzero numbers (here we use 2) elsewhere. Since the numbers of target training images are small in this dataset, we will not perform target fine-tuning or consecutive fine-tuning as baselines. We only compare our proposed JTLN using the zero-diagonal cost matrix with simple joint fine-tuning, both fine-tuned on an Alexnet pretrained with ImageNet, with base learning rate = 0.001, max iteration = 50000, parameter λ for OTloss is set to 1, the results are shown in Table 4.3.

Table 4.3: Experimental results on Office-31 Dataset: the number in brackets after each source domain is the number of images in this domain, the number pair in brackets after each target domain contains the number of training images and number of test images in this domain, the last two columns show accuracy results.

Source domain	Target domain	Joint Finetune	JTLN (zero-diagonal cost)
Amazon (2817)	Webcam (41-754)	0.669	**0.675**
Amazon (2817)	Dslr (43-455)	0.6571	**0.6747**
Dslr (498)	Webcam (41-754)	0.8342	**0.9244**
Dslr (498)	Amazon (59-2758)	0.5036	**0.5083**
Webcam (795)	Amazon (59-2758)	**0.512**	0.511
Webcam (795)	Dslr (43-455)	0.9253	**0.9451**

As can be seen, the proposed JTLN outperforms Joint fine-tuning in almost ev-

ery setting. This further shows that incorporating prior knowledge into DNNs really help the knowledge transfer between domains, and the proposed JTLN enables the incorporation of such prior knowledge in the training process.

4.5 Conclusion and Future work

In this chapter we have proposed a novel Joint Transfer Learning Network (JTLN) for Inductive Transfer Learning. By adding an Optimal Transport loss (OT loss) between the source and target classifier predictions during the joint fine-tuning process, the proposed JTLN can effectively learn useful knowledge for target tasks from source data. Another advantage of JTLN is the possibility of incorporating prior knowledge about the relatedness between the target and source categories by using different cost metric for OT loss. We show experimental results of JTLN with two different cost metrics in comparison with three baseline methods on two Inductive Transfer Learning datasets. The results verify the effectiveness of the proposed JTLN.

Future work includes further exploration of different cost metrics for OT loss. An interesting variant of JTLN could be to dynamically learn the cost matrix along the fine-tuning process while using the current fine-tuned model as feature extractor.

Chapter 5

Conclusion and Future Work

Throughout this thesis, we have firstly made a comprehensive literature review about the transfer learning algorithms related to image classification problems. We have then proposed two contributions for image classification problem under inductive transfer learning scenario. The first work is a boosting based algorithm which performs classifier level knowledge transfer, and the second work is a Deep Convolutional Neural Network based algorithm which performs feature level knowledge transfer.

More specifically, in chapter 3 we introduce DTL, a novel sparse representation based discriminative transfer learning algorithm, for learning target categories with only a few training samples by transferring knowledge from source categories that are different from target categories. Unlike other transfer learning algorithms, DTL explicitly makes use of both positively and negatively correlated source categories to help classification of the target category. Moreover, we introduce a novel Wilcoxon-Mann-Whitney based cost function to optimize the Area Under the ROC Curve (AUC), thereby enabling the proposed TL algorithm to deal with the unbalanced nature of data distribution. We further enhance DTL performance by running 2 AdaBoost processes concurrently on both the positive and negative distributions of the target training set. Experimental results on both the NUS-WIDE Scene and SUN datasets show that DTL consistently displays the best classification performance compared to other state-of-the-art transfer learning algorithms, when using different features and two different performance metrics. Time complexity analysis and experimental results both show the superiority of DTL compared to other boosting-based transfer learning algorithms: a training runtime roughly 80 times (20x, respectively) faster, as well as a testing runtime 66 times faster (33x,

respectively), than the other state-of-the-art TL methods compared when using the BOW features (AN50D, respectively).

In chapter 4 we have proposed a novel Joint Transfer Learning Network (JTLN) for Inductive Transfer Learning. By adding an Optimal Transport loss (OT loss) between the source and target classifier predictions during the joint fine-tuning process, the proposed JTLN can effectively learn useful knowledge for target tasks from source data. Another advantage of JTLN is the possibility of incorporating prior knowledge about the relatedness between the target and source categories by using different cost metric for OT loss. We show experimental results of JTLN with two different cost metrics in comparison with three baseline methods on two Inductive Transfer Learning datasets. The results verify the effectiveness of the proposed JTLN.

Some possible avenues of future work are:

Learn more fine-grained dictionaries for DTL: Since in current DTL algorithm the basic weak learners are sparse representation based classifiers (*i.e.* re-constructive dictionaries) formed with samples in two source categories. The samples in each source category are considered as a whole part without considering the possible intra-category variations. Therefore a more fine-grained dictionary learning process would possibly improve accuracy of the classification performance.

Dynamically learn the cost matrix along the fine-tuning process of JTLN: Since the cost matrix incorporates prior knowledge about the relatedness between each source-target category pair, therefore during feature learning process, as the feature space changes, the relatedness could also be different. Calculating the cost matrix dynamically in the current feature space would better incorporate prior knowledge about the relatedness.

Bibliography

[Ahuja *et al.* 1993] Ravindra K Ahuja, Thomas L Magnanti and James B Orlin. *Network flows: theory, algorithms, and applications.* 1993. 108

[Al-Shedivat *et al.* 2014] M. Al-Shedivat, J. Wang, M. Alzahrani, J. Z. Huang and X. Gao. *Supervised Transfer Sparse Coding.* In 28th AAAI Conf. Artificial Intelligence, pages 1665–1672, 2014. 71, 72

[Amit *et al.* 2007] Yonatan Amit, Michael Fink, Nathan Srebro and Shimon Ullman. *Uncovering shared structures in multiclass classification.* In Proceedings of the 24th international conference on Machine learning, pages 17–24. ACM, 2007. 13, 16, 64

[Ando & Zhang 2005] Rie Kubota Ando and Tong Zhang. *A framework for learning predictive structures from multiple tasks and unlabeled data.* Journal of Machine Learning Research, vol. 6, no. Nov, pages 1817–1853, 2005. 13, 14, 64

[Argyriou *et al.* 2007] Andreas Argyriou, Theodoros Evgeniou and Massimiliano Pontil. *Multi-task feature learning.* In Advances in neural information processing systems, pages 41–48, 2007. 13, 15, 64

[Aytar & Zisserman 2011] Y. Aytar and A. Zisserman. *Tabula Rasa: Model Transfer for Object Category Detection.* In Proc. 2011 Int. Conf. Computer Vision, pages 2252–2259, Washington, DC, USA, 2011. IEEE Computer Society. 50, 51, 64

[Benamou *et al.* 2015] Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna and Gabriel Peyré. *Iterative Bregman projections for regularized transportation problems.* SIAM Journal on Scientific Computing, vol. 37, no. 2, pages A1111–A1138, 2015. 108

[Bengio 2011] Y. Bengio. *Deep Learning of Representations for Unsupervised and Transfer Learning.* In Unsupervised and Transfer Learning - Workshop held

at ICML 2011, volume 27, pages 17–36, Bellevue, Washington, USA, July 2011. 70

[Burl & Perona 1996] Michael C Burl and Pietro Perona. *Recognition of planar object classes.* In Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on, pages 223–230. IEEE, 1996. 58

[Candès *et al.* 2006] E. J. Candès, J. K. Romberg and T. Tao. *Stable Signal Recovery from Incomplete and Inaccurate Measurements.* Comm. Pure Appl. Math., vol. 59, no. 8, pages 1207–1223, August 2006. 75

[Cawley 2006] Gavin C Cawley. *Leave-one-out cross-validation based model selection criteria for weighted LS-SVMs.* In Neural Networks, 2006. IJCNN'06. International Joint Conference on, pages 1661–1668. IEEE, 2006. 52

[Chua *et al.* 2009] T. Chua, J. Tang, R. Hong, H. Li, Z. Luo and Y. Zheng. *NUS-WIDE: A Real-world Web Image Database from National University of Singapore.* In Proc. ACM Int. Conf. Image and Video Retrieval, pages 48:1–48:9, New York, NY, USA, 2009. ACM. 88

[Courty *et al.* 2016] Nicolas Courty, Rémi Flamary, Devis Tuia and Alain Rakotomamonjy. *Optimal transport for Domain adaptation.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016. 103, 108

[Cuturi 2013] Marco Cuturi. *Sinkhorn distances: Lightspeed computation of optimal transport.* In Advances in Neural Information Processing Systems, pages 2292–2300, 2013. 108

[Dai *et al.* 2007] W. Dai, Q. Yang, G. Xue and Y. Yu. *Boosting for Transfer Learning.* In Proc. 24th Int. Conf. Machine Learning, pages 193–200, New York, NY, USA, 2007. ACM. 54, 64, 67, 70

[Ding *et al.* 2015] Z.-M. Ding, M. Shao and Y. Fu. *Deep low-rank coding for transfer learning.* In Proc. 24th Int. Conf. Artificial Intelligence, pages 3453–3459. AAAI Press, 2015. 70

[Duan *et al.* 2009] L. Duan, I.W. Tsang, D. Xu and S.J. Maybank. *Domain Transfer SVM for video concept detection.* In Proc. 2009 IEEE Conf. Computer Vision and Pattern Recognition, pages 1375–1381, June 2009. 70

[Eaton & Ruvolo 2013] Eric Eaton and Paul L. Ruvolo. *ELLA: An Efficient Lifelong Learning Algorithm.* In Proc. 30th Int. Conf. Machine Learning (ICML), volume 28, pages 507–515, 2013. 71

[Efron *et al.* 2004] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani*et al. Least angle regression.* Ann. of statistics, vol. 32, no. 2, pages 407–499, 2004. 87

[Fei-Fei *et al.* 2006] Li Fei-Fei, Rob Fergus and Pietro Perona. *One-shot learning of object categories.* IEEE transactions on pattern analysis and machine intelligence, vol. 28, no. 4, pages 594–611, 2006. 58, 64

[Fergus *et al.* 2003] Robert Fergus, Pietro Perona and Andrew Zisserman. *Object class recognition by unsupervised scale-invariant learning.* In Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on, volume 2, pages II–II. IEEE, 2003. 58

[Fernando *et al.* 2013] Basura Fernando, Amaury Habrard, Marc Sebban and Tinne Tuytelaars. *Unsupervised Visual Domain Adaptation Using Subspace Alignment.* In IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013, pages 2960–2967, 2013. 27, 64

[Fink 2005] Michael Fink. *Object classification from a single example utilizing class relevance metrics.* In Advances in neural information processing systems, pages 449–456, 2005. 31, 32, 64

[Freund & Schapire 1997] Y. Freund and R. Schapire. *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.* J. of Comput. and Syst. Sci., vol. 55, no. 1, pages 119–139, August 1997. 53, 84, 86, 91

[Frogner *et al.* 2015] Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya and Tomaso A Poggio. *Learning with a Wasserstein loss.* In Advances

in Neural Information Processing Systems, pages 2053–2061, 2015. 9, 103, 104, 107, 109

[Ganin & Lempitsky 2015] Yaroslav Ganin and Victor Lempitsky. *Unsupervised domain adaptation by backpropagation.* In International Conference on Machine Learning, pages 1180–1189, 2015. x, 36, 39, 40, 64

[Ge & Yu 2017] Weifeng Ge and Yizhou Yu. *Borrowing Treasures from the Wealthy: Deep Transfer Learning through Selective Joint Fine-tuning.* arXiv preprint arXiv:1702.08690, 2017. 102

[Gong *et al.* 2012] B. Gong, Y. Shi, F. Sha and K. Grauman. *Geodesic Flow Kernel for Unsupervised Domain Adaptation.* In Proc. 2012 IEEE Conf. Computer Vision and Pattern Recognition, pages 2066–2073, June 2012. 71, 72

[Gretton *et al.* 2012] Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu and Bharath K Sriperumbudur. *Optimal kernel choice for large-scale two-sample tests.* In Advances in neural information processing systems, pages 1205–1213, 2012. 36, 37, 111

[Jhuo *et al.* 2012] I. Jhuo, D. Liu, D.T. Lee and S. Chang. *Robust Visual Domain Adaptation with Low-rank Reconstruction.* In Proc. 2012 IEEE Conf. Computer Vision and Pattern Recognition, pages 2168–2175, June 2012. 72

[Jiang *et al.* 2008] Wei Jiang, Eric Zavesky, Shih-Fu Chang and Alex Loui. *Cross-domain learning methods for high-level visual concept classification.* In Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on, pages 161–164. IEEE, 2008. 51, 64

[Jie *et al.* 2011] L. Jie, T. Tommasi and B. Caputo. *Multiclass transfer learning from unconstrained priors.* In Proc. 2011 Int. Conf. Computer Vision, pages 1863–1870, November 2011. 70

[Kantorovich 2006] Leonid V Kantorovich. *On the translocation of masses.* Journal of Mathematical Sciences, vol. 133, no. 4, pages 1381–1382, 2006. 106

[Kong *et al.* 2017] Yu Kong, Ming Shao, Kang Li and Yun Fu. *Probabilistic Low-Rank Multitask Learning.* IEEE Trans. Neural Netw. Learn. Syst., 2017. 71

[Krizhevsky *et al.* 2012] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. *Imagenet classification with deep convolutional neural networks.* In Advances in neural information processing systems, pages 1097–1105, 2012. 38

[Kumar & Daume 2012] Abhishek Kumar and Hal Daume. *Learning Task Grouping and Overlap in Multi-task Learning.* In Proc. 29th Int. Conf. Machine Learning (ICML), pages 1383–1390, NY, USA, 2012. ACM. 71

[Kuzborskij *et al.* 2013] I. Kuzborskij, F. Orabona and B. Caputo. *From N to N+1: Multiclass Transfer Incremental Learning.* In Proc. 2013 IEEE Conf. Computer Vision and Pattern Recognition, pages 3358–3365, June 2013. 52, 53, 64, 91, 103

[Kuzborskij *et al.* 2015] I. Kuzborskij, F. Orabona and Barbara Caputo. *Transfer Learning Through Greedy Subset Selection.* In Proc. Int. Conf. Image Analysis and Processing, pages 3–14, 2015. 63, 64, 91, 103

[Li *et al.* 2014] L.-Y. Li, S. Li and Y. Fu. *Learning low-rank and discriminative dictionary for image classification.* Image and Vision Computing, vol. 32, no. 10, pages 814–823, 2014. 71

[Li *et al.* 2017] Sheng Li, Kang Li and Yun Fu. *Self-Taught Low-Rank Coding for Visual Learning.* IEEE Trans. Neural Netw. Learn. Syst., 2017. 45, 64, 70, 103, 104

[Li 2007] Xiao Li. Regularized adaptation: Theory, algorithms and applications, volume 68. Citeseer, 2007. 50

[Long & Servedio 2007] P. M. Long and R. A. Servedio. *Boosting the Area Under the ROC Curve.* In Adv. Neural Inf. Process Syst. (NIPS), 2007. 81

[Long *et al.* 2013a] M. Long, G. Ding, J. Wang, J. Sun, Y. Guo and P. S. Yu. *Transfer Sparse Coding for Robust Image Representation.* In Proc. 2013 IEEE Conf. Computer Vision and Pattern Recognition, pages 407–414, June 2013. 71, 72

[Long *et al.* 2013b] M. Long, J. Wang, G. Ding, J. Sun and P. S. Yu. *Transfer Feature Learning with Joint Distribution Adaptation.* In 2013 IEEE Int. Conf. Computer Vision, pages 2200–2207, December 2013. 17, 23, 64, 71, 72

[Long *et al.* 2015] M.-S. Long, Y. Cao, J.-M. Wang and M Jordan. *Learning Transferable Features with Deep Adaptation Networks.* In Proc. 32nd Int. Conf. Machine Learning, pages 97–105, 2015. ix, 17, 35, 36, 37, 64, 70, 103, 110

[Long *et al.* 2016] Mingsheng Long, Han Zhu, Jianmin Wang and Michael I. Jordan. *Unsupervised Domain Adaptation with Residual Transfer Networks.* In Adv. Neural Inf. Process Syst. (NIPS), 2016. 17, 39, 63, 64, 70, 103

[Lu *et al.* 2014] Ying Lu, Liming Chen, Alexandre Saidi, Zhaoxiang Zhang and Yunhong Wang. *Learning visual categories through a sparse representation classifier based cross-category knowledge transfer.* In IEEE Int. Conf. Image Processing (ICIP), pages 165–169. IEEE, 2014. vii, 68, 89, 90

[Mairal 2014] Julien Mairal. *SPAMS: a SPArse Modeling Software, v2. 5.* 2014. 87

[Maji *et al.* 2013] S. Maji, J. Kannala, E. Rahtu, M. Blaschko and A. Vedaldi. *Fine-Grained Visual Classification of Aircraft.* Rapport technique, 2013. 113

[Mann & Whitney 1947] H. B. Mann and D. R. Whitney. *On a test of whether one of two random variables is stochastically larger than the other.* Ann. of mathematical statistics, pages 50–60, 1947. 79

[Maurer *et al.* 2013] A. Maurer, M. Pontil and B. Romera-paredes. *Sparse Coding for Multitask and Transfer Learning.* In Proc. 30th Int. Conf. Machine Learning, volume 28, pages 343–351, May 2013. 71, 72

[Muller *et al.* 2001] K-R Muller, Sebastian Mika, Gunnar Ratsch, Koji Tsuda and Bernhard Scholkopf. *An introduction to kernel-based learning algorithms.* IEEE transactions on neural networks, vol. 12, no. 2, pages 181–201, 2001. 23

[Olshausen & Field 1996] Bruno A Olshausen and David J Field. *Emergence of simple-cell receptive field properties by learning a sparse code for natural images.* Nature, vol. 381, no. 6583, page 607, 1996. 43

[Pan & Yang 2010a] S. J. Pan and Q. Yang. *A Survey on Transfer Learning.* IEEE Trans. Knowl. Data Eng., vol. 22, pages 1345–1359, October 2010. 5, 6, 67, 68

[Pan & Yang 2010b] Sinno Jialin Pan and Qiang Yang. *A Survey on Transfer Learning.* IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pages 1345–1359, October 2010. 101, 103

[Pan *et al.* 2008] Sinno Jialin Pan, James T Kwok and Qiang Yang. *Transfer Learning via Dimensionality Reduction.* In AAAI, volume 8, pages 677–682, 2008. 17, 19, 64

[Pan *et al.* 2011] Sinno Jialin Pan, Ivor W Tsang, James T Kwok and Qiang Yang. *Domain adaptation via transfer component analysis.* IEEE Transactions on Neural Networks, vol. 22, no. 2, pages 199–210, 2011. 17, 21, 64

[Pan 2014] Sinno Jialin Pan. *Transfer Learning.* In Data Classification: Algorithms and Applications, pages 537–570. 2014. 6

[Parameswaran & Weinberger 2010] Shibin Parameswaran and Kilian Q Weinberger. *Large margin multi-task metric learning.* In Advances in neural information processing systems, pages 1867–1875, 2010. ix, 32, 33, 64

[Perrot & Habrard 2015] Michaël Perrot and Amaury Habrard. *A theoretical analysis of metric hypothesis transfer learning.* In International Conference on Machine Learning, pages 1708–1717, 2015. 34, 64

[Qi *et al.* 2007] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei and H.-J. Zhang. *Correlative Multi-label Video Annotation.* In Proc. 15th Int. Conf. Multimedia, MULTIMEDIA '07, pages 17–26, New York, NY, USA, 2007. ACM. 89

[Qi *et al.* 2011] G.-J. Qi, C. Aggarwal, Y. Rui, Q. Tian, S. Chang and T. Huang. *Towards Cross-category Knowledge Propagation for Learning Visual Concepts.* In Proc. 2011 IEEE Conf. Computer Vision and Pattern Recognition, pages 897–904, June 2011. vii, 56, 64, 67, 70, 73, 87, 88, 89, 90, 91

[Quattoni *et al.* 2007] Ariadna Quattoni, Michael Collins and Trevor Darrell. *Learning visual representations using images with captions.* In Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pages 1–8. IEEE, 2007. 15, 64

[Quattoni *et al.* 2008] A. Quattoni, M. Collins and T. Darrell. *Transfer Learning for Image Classification with Sparse Prototype Representations.* In Proc. 2008 IEEE Conf. Computer Vision and Pattern Recognition, pages 1–8, June 2008. 71, 72

[Quattoni *et al.* 2009] Ariadna Quattoni, Michael Collins and Trevor Darrell. *Transfer learning algorithms for image classification.* PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2009. 3

[Raina *et al.* 2007] R. Raina, A. Battle, H. Lee, B. Packer and A. Y. Ng. *Self-taught Learning: Transfer Learning from Unlabeled Data.* In Proc. 24th Int. Conf. Machine Learning, 2007. 43, 46, 64, 70

[Rosen-Zvi *et al.* 2010] Michal Rosen-Zvi, Chaitanya Chemudugunta, Thomas Griffiths, Padhraic Smyth and Mark Steyvers. *Learning author-topic models from text corpora.* ACM Transactions on Information Systems (TOIS), vol. 28, no. 1, page 4, 2010. 61

[Russell *et al.* 1995] Stuart Russell, Peter Norvig and Artificial Intelligence. *A modern approach.* Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs, vol. 25, page 27, 1995. 1

[Samuel 1959] Arthur L Samuel. *Some studies in machine learning using the game of checkers.* IBM Journal of research and development, vol. 44, no. 1.2, pages 206–226, 1959. 1

[Satpal & Sarawagi 2007] Sandeepkumar Satpal and Sunita Sarawagi. *Domain adaptation of conditional probability models via feature subsetting.* In PKDD, volume 4702, pages 224–235. Springer, 2007. 17, 64

[Shao *et al.* 2014a] L. Shao, F. Zhu and X. Li. *Transfer Learning for Visual Categorization: A Survey.* IEEE Trans. Neural Netw. Learn. Syst., vol. PP, no. 99, pages 1–1, July 2014. 70

[Shao *et al.* 2014b] M. Shao, D. Kit and Y. Fu. *Generalized Transfer Subspace Learning Through Low-Rank Constraint.* Int. J. Comput. Vis., vol. 109, no. 1-2, pages 74–93, January 2014. 71

[Shao *et al.* 2014c] M. Shao, M.-B. Ma and Y. Fu. *Sparse Manifold Subspace Learning.* In Low-Rank and Sparse Modeling for Visual Analysis, pages 117–132. Springer, 2014. 71

[Shao *et al.* 2015] Ling Shao, Fan Zhu and Xuelong Li. *Transfer learning for visual categorization: A survey.* IEEE transactions on neural networks and learning systems, vol. 26, no. 5, pages 1019–1034, 2015. 101, 103

[Si *et al.* 2010] S. Si, D. Tao and B. Geng. *Bregman Divergence-Based Regularization for Transfer Subspace Learning.* IEEE Trans. Knowl. Data Eng., vol. 22, pages 929–942, July 2010. 17, 64, 70

[Srebro & Jaakkola 2003] Nathan Srebro and Tommi Jaakkola. *Weighted low-rank approximations.* In Proceedings of the 20th International Conference on Machine Learning (ICML-03), pages 720–727, 2003. 16

[Stewart & Brown 2005] N. Stewart and G. D. A. Brown. *Similarity and Dissimilarity as Evidence in Perceptual Categorization.* J. Math. Psychology, vol. 49, no. 5, pages 403–409, October 2005. 68, 97

[Sun & Saenko 2015] Baochen Sun and Kate Saenko. *Subspace Distribution Alignment for Unsupervised Domain Adaptation.* In BMVC, pages 24–1, 2015. 29, 64

[Thrun 1996] Sebastian Thrun. *Is Learning The n-th Thing Any Easier Than Learning The First?* In Advances in Neural Information Processing Systems, pages 640–646. The MIT Press, 1996. 12, 31, 64

[Tommasi *et al.* 2010] T. Tommasi, F. Orabona and B. Caputo. *Safety in Numbers: Learning Categories from Few Examples with Multi Model Knowledge Transfer.* In Proc. 2010 IEEE Conf. Computer Vision and Pattern Recognition, pages 3081–3088, June 2010. 51, 53, 64

[Tzeng *et al.* 2014] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko and Trevor Darrell. *Deep domain confusion: Maximizing for domain invariance.* arXiv preprint arXiv:1412.3474, 2014. 35, 64

[Tzeng *et al.* 2015] Eric Tzeng, Judy Hoffman, Trevor Darrell and Kate Saenko. *Simultaneous deep transfer across domains and tasks.* In Proceedings of the IEEE International Conference on Computer Vision, pages 4068–4076, 2015. 36, 64

[Valiant 1984] Leslie G Valiant. *A theory of the learnable.* Communications of the ACM, vol. 27, no. 11, pages 1134–1142, 1984. 1

[Vapnik & Kotz 1982] Vladimir Naumovich Vapnik and Samuel Kotz. Estimation of dependences based on empirical data, volume 40. Springer-Verlag New York, 1982. 86

[Vapnik & Vapnik 1998] Vladimir Naumovich Vapnik and Vlamimir Vapnik. Statistical learning theory, volume 1. Wiley New York, 1998. 1

[Wang & Bensmail 2013] J. Wang and H. Bensmail. *Cross-domain Sparse Coding.* In Proc. 22nd ACM Int. Conf. Information and Knowledge Management, pages 1461–1464, New York, NY, USA, 2013. ACM. 71, 72

[Wang *et al.* 2013] Hua Wang, Feiping Nie and Heng Huang. *Robust and discriminative self-taught learning.* In International Conference on Machine Learning, pages 298–306, 2013. 45, 64

[Weinberger & Saul 2009] Kilian Q Weinberger and Lawrence K Saul. *Distance metric learning for large margin nearest neighbor classification.* Journal of Machine Learning Research, vol. 10, no. Feb, pages 207–244, 2009. 34

[Wilcoxon 1945] F. Wilcoxon. *Individual comparisons by ranking methods.* Biometrics bulletin, vol. 1, no. 6, pages 80–83, 1945. 79

[Wright *et al.* 2009] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry and Yi Ma. *Robust Face Recognition via Sparse Representation.* IEEE Trans. Pattern Anal. Mach. Intell., vol. 31, pages 210–227, February 2009. 71, 72, 74, 75, 91

[Xiao *et al.* 2010] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva and A. Torralba. *SUN Database: Large-scale Scene Recognition from Abbey to Zoo.* In Proc. 2010 IEEE Conf. Computer Vision and Pattern Recognition, pages 3485–3492, June 2010. 88, 90

[Yang *et al.* 2007] J. Yang, R. Yan and A. G. Hauptmann. *Cross-domain Video Concept Detection Using Adaptive Svms.* In Proc. 15th Int. Conf. Multimedia, pages 188–197, New York, NY, USA, 2007. ACM. 49, 70

[Yang *et al.* 2011] M. Yang, L. Zhang, J. Yang and D. Zhang. *Robust sparse coding for face recognition.* In Proc. 2011 IEEE Conf. Computer Vision and Pattern Recognition, pages 625–632. IEEE, 2011. 71

[Yao & Doretto 2010] Y. Yao and G. Doretto. *Boosting for Transfer Learning With Multiple Sources.* In Proc. 2010 IEEE Conf. Computer Vision and Pattern Recognition, pages 1855–1862, June 2010. 54, 55, 64, 70, 89

[Yosinski *et al.* 2014] Jason Yosinski, Jeff Clune, Yoshua Bengio and Hod Lipson. *How transferable are features in deep neural networks?* In Advances in neural information processing systems, pages 3320–3328, 2014. 38, 102, 113

[Yu & Aloimonos 2010] Xiaodong Yu and Yiannis Aloimonos. *Attribute-based transfer learning for object categorization with zero/one training example.* Computer Vision–ECCV 2010, pages 127–140, 2010. 60, 64

[Zhang *et al.* 2017] Jing Zhang, Wanqing Li and Philip Ogunbona. *Joint Geometrical and Statistical Alignment for Visual Domain Adaptation.* arXiv preprint arXiv:1705.05498, 2017. 29, 64