Automatic Domain Partitioning for Multi-Domain Learning

Di Wang
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
diwang@cs.cmu.edu

Chenyan Xiong
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
cx@cs.cmu.edu

William Yang Wang
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
ww@cmu.edu

Abstract

Multi-Domain learning (MDL) assumes that the domain labels in the dataset are known. However, when there are multiple metadata attributes available, it is not always straightforward to select a single best attribute for domain partition, and it is possible that combining more than one metadata attributes (including continuous attributes) can lead to better MDL performance. In this work, we propose an automatic domain partitioning approach that aims at providing better domain identities for MDL. We use a supervised clustering approach that learns the domain distance between data instances , and then cluster the data into better domains for MDL. Our experiment on real multi-domain datasets shows that using our automatically generated domain partition improves over popular MDL methods.

1 Introduction

Instead of assuming data are *i.i.d*, Multi-domain learning (MDL) methods assumes that data come from several domains and make use of domain labels to improve modeling performance (Daumé III, 2007). The motivation of using MDL is that datasets from different domains could be different, in two ways. First, the feature distribution $p(x)$ could be domain specific, meaning that the importance of each feature is different across domains. Second, the distribution of label Y given X, $p(y|x)$, of different domains could be different. These differences could create problems for traditional machine learning methods: models learned from one domain might not be generalizable to other domains (Ben-David et al., 2006; Ben-David et al., 2010).

One common assumption of MDL methods is that the domain identities are pre-defined. For example, in the multi-domain Amazon product review dataset (Finkel and Manning, 2009), the product categories are typically used as the domain identities. However, a question raised by Joshi et al. (2012) is that, in real-world data sets, there could be many ways to split data into domains, and it is hard to decide which one to use. Consider the Amazon product reviews, where we have multiple attributes attached to each review: for example, product category, reviewer location, price, and number of feedback. Which attribute is the most informative domain label? Or we should use all of these meta-data and partition the data into many small domains?

In this paper, we investigate the problem of automatic domain partitioning. We propose an empirical domain difference testing method to examine whether two groups of data are *i.i.d*, or generated from different distributions, and how different they are. Using this approach, we generate data pairs that belong to the same distribution, and data pairs that should be partitioned into different domains. These pairs are then used as training data for a supervised clustering algorithm, which automatically partitions the dataset into several domains. In the evaluation, we show that our automatically-partitioned domains improve the performances of two popular MDL methods on real sentiment analysis data sets.

Note that Joshi et al. (2013) proposed a Multi-Attribute Multi-Domain learning (MAMD) method, which also exploited multiple dimensions of meta-

data and provided extensions to two traditional MDL methods. However, extensions to the MAMD setting may not be trivial for every MDL algorithm, while our method serves as a pre-processing step and can be easily used for all MDL approaches. In addition to this, MAMD only works with categorical metadata, and can not fully utilize information in the form of continuous metadata values.

2 Automatic Domain Partitioning

In this section, we introduce the Automatic Domain Partitioning (ADP) problem: given data X, metadata M and label Y, find a function $g : M \mapsto I$ such that the common MDL methods perform better with data X and domain identity I in the prediction of Y. For example, on Amazon sentiment analysis data, X is the feature matrix extracted from reviews, Y is the positive or negative label vector, and M is the metadata matrix associated with reviews (e.g. product price and category).

Our approach works as follows: in training, we first use an empirical domain difference testing method to detect whether two groups of data should be considered as different domains; after that we apply supervised clustering to learn the distance metric between two data points, i.e. how different thay are in MDL view, from training data generated by our domain difference test method; finally, based on the distance metric learned, we cluster our data into several clusters, and train MDL models with those clusters as domain labels; in testing, we assign data instance to its nearest cluster and use that cluster as its domain identity, and then apply the trained MDL models for prediction.

2.1 Empirical Domain Difference Test

The key motivation of MDL is that a model fits for one domain may not fit well for other domains. Following the same motivation, we propose an empirical method for domain difference test called Domain Model Loss (DML) that provides us the domain difference score $d(G_1, G_2)$ between two groups of data $G_1 = \{X_1, Y_1\}$ and $G_2 = \{X_2, Y_2\}$.

Domain Model Loss If the mapping functions $f_1 : X_1 \mapsto Y_1$ and $f_2 : X_2 \mapsto Y_2$ are different for two data groups, we could directly use the disagreement of f_1 and f_2 as domain difference score. More specifically, if we train two classifiers $\hat{f}_1 : X_1 \mapsto Y_1, \hat{f}_2 : X_2 \mapsto Y_2$ individually on G_1 and G_2, we could have the K-fold empirical loss:

$$\hat{l}(f_1, G_1) = \frac{1}{K} \sum_i \text{Error of } f_1 \text{ on i-th fold of } G_1,$$

$$\hat{l}(f_2, G_2) = \frac{1}{K} \sum_i \text{Error of } f_2 \text{ on i-th fold of } G_2.$$

And we could also apply the trained model f_1 on G_2, and f_2 on G_1 to get:

$$\hat{l}(f_1, G_2) = \text{Error of } f_1 \text{ on } G_2,$$
$$\hat{l}(f_2, G_1) = \text{Error of } f_2 \text{ on } G_1.$$

Then, if G_1 and G_2 are actually the same with each other, then both models will have same empirical loss on either data set, but if they are not, we will have a positive DML score:

$$DML(G_1, G_2) = \frac{1}{2}(\hat{L}(f_1, G_2) + \hat{L}(f_2, G_1)),$$

where:

$$\hat{L}(f_1, G_2) = \frac{\hat{l}(f_1, G_2) - \hat{l}(f_1, G_1)}{\hat{l}(f_1, G_1)},$$
$$\hat{L}(f_2, G_1) = \frac{\hat{l}(f_2, G_1) - \hat{l}(f_2, G_2)}{\hat{l}(f_2, G_2)}.$$

2.2 Supervised Clustering for Domain Partitioning

Our domain difference test method calculates the distance between two partitioned data groups. However, to directly use it for domain partitioning, we must go through all possible combinations of domain assignments in exponential time, which is infeasible. Our solution is to use a polynomial-time supervised clustering method developed by Xing et al. (2002) to learn a distance function that calculates the distance between any two data points. Formally, given a set of data pairs D, which belong to different domains, and a set of data pairs S, which belong to the same domain, it

$$\begin{aligned} \max_A \ g(A) &= \sum_{(i,j)\in D} \sqrt{(m_i - m_j)^T A (m_i - m_j)} \\ s.t. \ f(A) &= \sum_{(i,j)\in S} (m_i - m_j)^T A (m_i - m_j) \le 1 \\ A &\succeq 0, \end{aligned}$$

where m_i, m_j are meta data of i and j.

The metadata M are preprocessed as follows: 1) Each categorical attribute was converted to several binary questions, one per category, and each binary question was considered as one metadata dimension in ADP method. For example, if categorical attribute "Product Type" has two values "Music" and "Electronics", then there will be two dimensions of metadata corresponding to "Product Type" in ADP. Two metadata dimensions correspond to binary questions: "Is Product Type Music" and "Is Product Type Electronics". 2) Each continuous attribute was normalized by scaling between 0 and 1.

The training data S, D for metric learning are generated as follows:

1. For each dimension M_k of M, split data at value 0.5, sample two equally sized groups, apply our domain difference testing method and find the difference between these data groups.
2. Assign distance to each pair of instances by the average distance of all partitions that partitions the pair into different groups.
3. Select top n similar pairs as S and top n different pairs as D.

The learned distance metric A now conveys the domain difference information obtained from our domain distance test results: which meta attributes are important for domain partitioning and which are not as important. Following Xing et al. (2002), we transfer the instance's metadata feature M by MB^T, where $B^T B = A$. Then we use a clustering method on MB^T, and the output is our domain partitioning result.

3 Experiment Methodology

Datasets To evaluate our methods, we used two subsets of Amazon review corpus (Jindal and Liu, 2008), which originally contain 5.8 million reviews with a variety of metadata about products and users. The first subset (`BOOK`) contains 20,000 reviews on books published by eleven most popular publishers, while the second (`PROD`) is reviews about products within seven most common product categories. We randomly split each dataset into training and testing sets with equal size. The task is to predict a positive or negative label for each review. Case insensitive unigrams excluding stop words are used as features, and all features appear less than 500 times are removed for efficient experiment processing. Reviews of 4 or 5 stars are considered positive and 1 or 2 stars are considered negative, while 3 stars reviews are excluded. Each review has multiple metadata such as book's publisher, product's type, user's state location, product price, review year, and number of other user feedback. Reviews with missing metadata are filtered out.

MDL Methods Our first MDL algorithm is the Frustratingly Easy Domain Adaptation (**FEDA**) (Daumé III, 2007) which is easy to implement and achieved competitive performance on many applications. It creates an augmented feature space as the Cartesian product of the input features and the original domains plus a shared domain. Then it uses a SVM classifier over the augmented feature space to obtain classification result. Specifically, our FEDA methods use L_2-regularized SVM with linear kernel by LIBLINEAR package[1]. The parameters C = 0.01 was selected using five-fold cross-validation on training set.

Our second MDL algorithm is Multi-Domain Regularization (**MDR**) (Dredze and Crammer, 2008), which is a classifier combination approach based on Confidence-Weighted (CW) learning (Dredze et al., 2008). The CW learning is an online update method that maintains probabilistic confidence for each parameter by keeping track of its variance. In our experiments, we use the CW implementation provided by its authors and choose the best performing configurations described in (Dredze and Crammer, 2008).

Domain Partition Methods We evaluated the domain partition results provided by our ADP on the two MDL methods (FEDA & MDR). For simplicity and efficiency, we use Naive Bayes as our base prediction model f_1 and f_2 to generate the domain model loss score, described in section 2.1. In training data generation, we choose top 10% similar pairs as S and top 10% different pairs as D. And given the learnt distance metric A, we use K-means to do the clustering. The number of clusters is selected by five-fold cross-validation on training set.

[1] http://www.csie.ntu.edu.tw/∼cjlin/liblinear

We compare our domain partition quality with three other methods: 1) **1-Best** chooses best performing categorical metadata on a validation set as domain indicators, where the original training set was splitted equally to train and validate the performance of each categorical attribute; 2) **Random** partition that assigns domain identities to instances randomly with same number of domains as 1-Best. We run each random partition ten times and took the average; 3) **MAMD** proposed by Joshi et al. (2013). However, the original version of MAMD does not support continuous attribute such as price. So we made an extension that sorts these values to ten bins and then treats them as categorical values.

4 Results and Discussions

Partition + MDL	PROD	BOOK
ADP + FEDA	82.02 $*$	86.22 ‡$*$
MAMD + FEDA	81.04	86.08
1-Best + FEDA	82.00	85.85
Random + FEDA	79.36	84.72
ADP + MDR	82.10 ♯ ‡ $*$	86.62 ♯ ‡ $*$
MAMD + MDR	80.17	84.37
1-Best + MDR	79.79	83.68
Random + MDR	74.65	81.16

Table 1: Overall accuracies on PROD and BOOK datasets. ADP results that are statistically significantly better than MAMD are marked with ♯, and better than 1-Best and Random are indicated by ‡ and $*$ respectively, using a paired t-test, with $p < 0.05$.

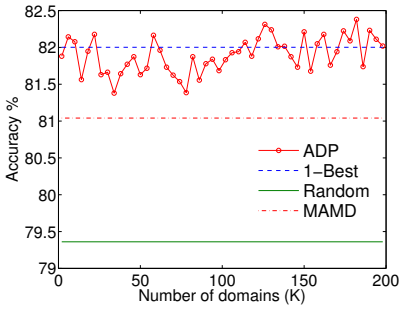
Table 1 shows the overall experimental results of four domain partition methods with two MDL methods on PROD and BOOK datasets. One could see that when using MDR method, ADP could significantly outperform all baselines on both data sets, with relatively more than 2% gains. For FEDA, on PROD data, ADP performs the same with MAMD and 1-Best; on BOOK data, ADP outperforms 1-Best significantly, but is just slightly better than MAMD. One possible reason is that the best numbers of cluster selected by cross-validation are around 150. With such large number of none-perfect domains, FEDA will generate huge dimension of features and perhaps require more training data to provide better performances. Another possible reason is that FEDA and the SVM underlying FEDA are very robust against bad domain partition results. This might be the reason of high FEDA baselines. In general, our ADP method helps existing MDL approaches achieve better performance, while bad (Random) partitioning does hurt.

Figure 1(a) and 1(b) shows the performances of applying FEDA on different domain partitioning methods on PROD and BOOK, while Figure 1(c) and 1(d) shows experiment results with MDR. The x-axis is the size of the output domains (the K in our K-means clustering), and y-axis is the accuracy of models. With our domain partitioning approach, MDR can perform consistently higher than all the three baselines on both dataset when $k > 50$. As we discussed for Table 1, FEDA might be less sensitive to domain partition results, which causes high baseline performance and high ADP+FEDA performance with small K. Since the performance trends to increase along with k until 50 in three figures (1(b), 1(c) and 1(d)), we believe that the ground-truth domain size is likely larger than 50. These results clearly indicate ADP does provide more desirable domain assignments for MDL. The domain selected by 1-Best such as publishers has only 11 domains, which limits the ability of 1-Best to completely express domain information. And our generated domains integrate multiple metadata attributes, lead to more detailed domain partitions, and enhance the ability of MDL methods to capture the difference between different groups of data. Although accuracies are growing with k in general, we also see that there are fluctuations on curves especially when curves are zoomed to a small range. To get smoother results, we can sample more data to calculate domain similarity and repeat the K-means clustering with more different initializations.

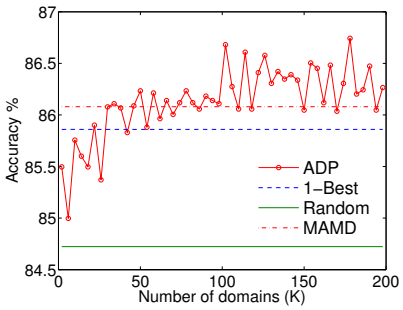
5 Conclusions

In this paper, we propose an Automatic Domain Partition (ADP) method that provides better domain identities for multi-domain learning methods. We first propose a new approach to identify whether two data groups should be considered as different domains, by comparing the differences using Domain Model Loss. We use a supervised clustering approach to train our model with labels generated by domain difference tests, and cluster the re-weighted

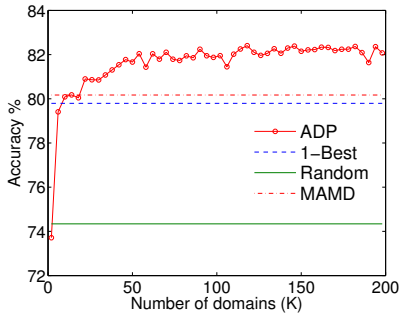
metadata as our domain partition by K-means. Experiments on real world multi-domain data show that the domain identities generated by our method can improve the performance of MDL models.



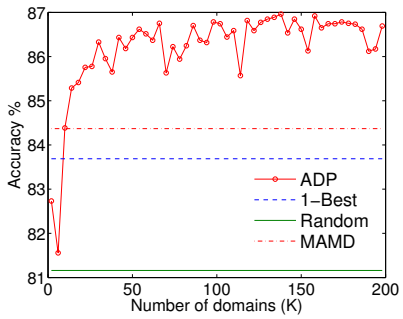
(a) FEDA results on PROD



(b) FEDA results on BOOK



(c) MDR results on PROD



(d) MDR results on BOOK

Figure 1: Accuracies over different size of the output domains (K)

References

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 137–144.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175.

Mark Dredze and Koby Crammer. 2008. Online methods for multi-domain learning and adaptation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 689–697.

Mark Dredze, Koby Crammer, and Fernando Pereira. 2008. Confidence-weighted linear classification. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML)*, pages 264–271.

Jenny Rose Finkel and Christopher D. Manning. 2009. Hierarchical bayesian domain adaptation. In *Proceedings of the 2009 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 602–610.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the International Conference on Web Search and Web Data Mining (WSDM)*, pages 219–230.

Mahesh Joshi, Mark Dredze, William W. Cohen, and Carolyn Penstein Rosé. 2012. Multi-domain learning: When do domains matter? In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, (EMNLP-CoNLL)*, pages 1302–1312.

Mahesh Joshi, Mark Dredze, William W. Cohen, and Carolyn P. Rosé. 2013. Whats in a domain? multi-domain learning for multi-attribute data. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 685–690, Atlanta, Georgia, June. Association for Computational Linguistics.

Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart J. Russell. 2002. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems (NIPS)*, pages 505–512.