Study of Transfer Learning

Ziyue Liu*
1036109

1 Introduction

Domain adaptation is an interesting problem in NLP, referring to the development of learning algorithms that can be easily ported from one domain to another. For example, we have a large amount of labeled data in a "source" domain (for example, a news-line), but we want a model that performs well in other "target" domains. This article mainly learns the idea proposed in Daume III's paper [1] based on transforming the domain adaptive learning problem into a standard supervised learning problem that can apply any standard algorithm (such as neural network, etc.). Fully supervised machine learning cases are performed by expanding the feature space of the source and target data, which is a very simple transformation, while using the results as input to standard learning algorithms.

The data sources used in this article are divided into three different domain: MALE.csv, FEMALE.csv, and MIXED.csv, which respectively count the personal information and scores of male students, female students, and coeducational students. The purpose is to build a model with mean square error as small as possible through transfer learning.

2 Prior Work

In order to get the source domain and the target domain from the three domains, firstly divide 100 instances of the development set and 100 instances of the test set from each of the three domains, and the remaining examples are used as the test set. When one of the domains serves as the target domain, the remaining two domains together become the source domain.

In addition, in this paper, Linear Regression (LR) and Multilayer Perceptron (MLP) are chosen training algorithm.

LR is the basic machine learning algorithm and it is easy to implement. The main idea is constructing the loss function and calculating the parameters w and b when the loss function is minimum are solved.

$$\widehat{y} = wx + b \tag{1}$$

And MLP is a kind of neural network, which is one of the most popular field in machine learning. By connecting multiple eigenvalues, through a combination of linear and nonlinear, a goal is finally achieved. For example, a three-layer MLP is summarized by the formula:

$$f(x) = G(b^{(2)} + W^{(2)}(s(b^{(1)} + W^{(1)}x))) \tag{2}$$

All the parameters of MLP are the connection weight and offset between the various layers, including W1, b1, W2, b2.

2.1 Baseline Calculation

According to Daume III's article, there are several "obvious" ways to attack the domain adaptation problem without developing new algorithms [1]: SRCONLY, TGTONLY, ALL, WEIGHTED, PRED and LININT. In this report, some changes are made to fit the current experiment:

*Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies.

		SRCONLY	TGTONLY	ALL	WEIGHTED	PRED	LININT
MLP	Male	132.37	135.87	**131.51**	136.54	135.93	132.07
	Female	120.13	153.52	119.91	124.12	**117.25**	121.91
	Mixed	140.82	**125.03**	140.27	139.69	126.08	129.01
LR	Male	164.4	159.91	164.34	164.02	**159.89**	159.91
	Female	175.48	156.94	175.24	173.46	**156.64**	156.94
	Mixed	173.69	173.44	173.42	**172.77**	173.44	172.88

Figure 1: Baseline MSE Result

- In ALL baseline, only 100 instances in target training set is used in source domain.
- In WEIGHTED baseline, several time of copying the training examples of the target domain (100 instances) is used in the source domain to reduce the proportion of out-of-domain instances.

Mean square error (MSE) is used to evaluate the quality of the training model. The result is shown in table 1.

2.2 Analysis

In table 1, generally, MSE that obtained by MLP is smaller than the results get from LR, which indicates the better performance of MLP. Because mlp performs multiple fittings and then assigns weights to the eigenvalues, the perormance of fitting the data is better than linear regression when the data distribution is very fragmented. Further more, different with the referenced paper, which shows that ALL and WEIGHTED baseline preform better than other baseline approaches, LININT baseline has the best performance in this experiment.

In MLP, the hyperparameter I changed is hidden layer size and learning rate. The range of hidden layer size and learning rate are 600 to 2000 and 0.001 to 0.0002, that 0.0002 is the recommend setting according to Young-Bum K. [3]. however, the MSE really changed a little and the same situation is happened in changing hyperparameter of LR. Because, the higher the hidden layer, the slower the learning rate, the deeper the learning will be. If the transformation is not obvious, it can be considered that the hidden links between different domains are indeed very small.

3 Frustratingly Easy Domain Adaptation (FEDA)

3.1 Data Prepossess

Considering X and Y are the input and output spaces respectively. D^s and D^t are source domain sets and target domain sets separately. Before augmenting features, $X = \mathbb{R}^F$. Then the input space will be added to $X = \mathbb{R}^{(K+1)F}$, which K is the amount of the total domains. This operation is defined by replicating the origin feature vector **x** and adding zero vector, $\mathbf{0} = \langle 0, 0, \ldots, 0 \rangle$. In this project, the final feature vectors is:

$$\Phi^M = \langle x, x, 0, 0 \rangle, \Phi^F = \langle x, 0, x, 0 \rangle, \Phi^{Mix} = \langle x, 0, 0, x \rangle \tag{3}$$

Then choose the best performed algorithm in section 2.1 (LININT baseline with MSE) is doing to compare the effect with baseline results. the result is shown in table 2.

3.2 Analysis

The setting hyperparameters are compared between paper[1] recommend and settinf in section 2. Latter one performs better. First, different zero vector insertion methods are used to generate new feature vectors (i.e. zero vectors are tried to be inserted into different positions). After comparison, the new feature vectors shown in this article have the best performance. Comparing with previous baseline approaches, FEDA really has improvement in MSE results. I think this is because the weights of the target domain and the source domain increase at the same time.

Task	Dom	SRCONLY	TGTONLY	ALL	WEIGHTED	PRED	LININT	AUG	T<S Win
MLP	Male	132.37	135.87	**131.51**	136.54	135.93	132.07	135.41	+ +
	Female	120.13	153.52	119.91	124.12	119.25	118.91	**99.71**	+ +
	Mixed	140.82	125.03	140.27	139.69	126.08	129.01	**111.17**	- -
LR	Male	164.4	159.91	164.34	164.02	159.92	**159.71**	180.95	- -
	Female	175.48	156.94	175.24	173.46	156.64	155.94	**135.17**	- -
	Mixed	173.59	173.64	173.42	172.77	173.44	172.88	**142.13**	+ +

Figure 2: Results of Baseline and FEDA

Task	Dom	SRCONLY	TGTONLY	ALL	WEIGHTED	PRED	LININT	AUG	KMM
MLP	Male	132.37	135.87	131.51	136.54	135.93	132.07	135.41	**93.88**
	Female	120.13	153.52	119.91	124.12	119.25	118.91	**99.71**	133.38
	Mixed	140.82	125.03	140.27	139.69	126.08	129.01	**111.17**	119.39
LR	Male	164.4	159.91	164.34	164.02	159.92	159.71	180.95	**120.11**
	Female	175.48	156.94	175.24	173.46	156.64	155.94	**135.17**	188.27
	Mixed	173.69	173.44	173.42	172.77	173.44	172.88	**142.13**	156.21

Figure 3: Result of Previous Work and KMM

4 Append Study of Transfer Learning

In section 2, WEIGHTED and LININT baseline method are both focusing on the weight of data. However, all the data in the entire data set are only multiplied by a coefficient w. And because each instance in the data set used in this experiment is independent, I want to separately and fit a weight coefficient for each data. Finally, I found kernel mean matching (KMM), which is pointed in

4.1 Kernel Mean Matching

Kernel mean matching (KMM) uses a non-parametric approach (unsupervised) to directly infer the resampling weights through distribution matching between the training set and the test set in the feature space. The formula [5] is shown below:

$$\min_{\beta} \left\| E_{x' \sim P_r'} \left[\Phi(x') \right] - E_{x \sim P_r} \left[\beta(x) \Phi(x) \right] \right\| \tag{4}$$

$$s.t. \beta(x) \geq 0, E_{x \sim P_r} \left[\beta(x) \right] = 1 \tag{5}$$

The core code about how to get β is obtained from `https://github.com/jindongwang/transferlearning/blob/master/code/traditional/KMM.py`. In this experiment, one domain is regarded as target domain and the other two are source domain.Then compute the β with target domain and the domain in source respectively. After getting β and calculate the weighted source domain data, MLP is used to train the regression model.

4.2 Analysis

From the table 3, the only well performance is happened when male domain is the target domain. FEDA seems work better in general cases. Due to computer performance, KMM cannot be performed on the basis of FEDA dataset.

Acknowledgments

Thanks to my computer, even if the program crashes many times, the program and the paper I'm writing are saved for me.

References

[1] III, Hal. (2009) Frustratingly Easy Domain Adaptation. *ACL 2007 - Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

[2] Daume III, H. & D. Marcu. (2006) Domain Adaptation for Statistical Classifiers. *Journal of Artificial Intelligence Research* **26**: 101–126. Crossref. Web.

[3] Young-Bum K. , Karl S. , Ruhi S. (2016) Frustratingly Easy Neural Domain Adaptation. The COLING 2016 Organizing Committee pp: 387–396

[4] Diederik P. Kingma, Jimmy Ba. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.

[5] Bernhard S. John P. Thomas H. (2007) Correcting Sample Selection Bias by Unlabeled Data. Advances in Neural Information Processing Systems **19**: Proceedings of the 2006 Conference , MITP, pp.601-608.