Project 2: Domain Adaptation

Yazad J. Davur, [redacted] ydavur@student.unimelb.edu.au

1 Introduction

The task of domain adaptation is to develop learning algorithms with a central goal of generalization. The difference in training and test distributions raises the challenge where training is required to be robust to domain disparity, such that domain-general perceptions are learned rather than domain-specific phenomena, which will not adapt well to out-of-domain evaluation.

The dataset used for evaluation emanates from the Inner London Education Authority (ILEA). It has been split by school-gender into three separate files, *male*, *female* and *mixed* (our three 'domains'). Each file includes a row for each student, with a number of features about the student and their exam score (Response variable). The data is sourced from: http://www.bristol.ac.uk/cmm/media/migrated/ilea567.zip.

We utilize two machine learning algorithms throughout this paper to generate predictions of the exam score. While none of the features single handedly are strongly (anti) correlated to the response variable, a linear combination of these features does correlate well and hence the Linear Regression (LR) model performs effectively as compared to other statistical learning models (Decision Trees, Random Forests, XGBoost). For our second model, a Multilayer Perceptron (MLP) Regressor is chosen since it is simple, yet a flexible feed forward neural network that can be used to learn a mapping between the inputs and outputs. Since we use a LR model, the categorical data in the original feature set is transformed to a set of new binary features (One-hot encoding). Furthermore, our Neural network also performs better with this manipulated feature set.

2 Baseline Approach

There are several "obvious" ways to attack the domain adaptation problem without developing new algorithms as presented and evaluated in [2]. For each of our 3 domains (*male, female, mixed*), we partition the data into Train, Development and Test, and to mimic a data impoverished scenario, the amount of train and development data samples of the target domain are always restricted to 100 instances only. The other two domains are treated as source domains.

Taking inspiration from [3], hyper-parameters are optimized appropriately using random search rather than a grid search technique, on selective parameters (hidden layer sizes, (h), learning rate initialization (η) & L2 penalty parameter, alpha) that impact the performance positively. In certain cases, a line search is also utilized. The maximum iterations are set at 1000 for all cases to ensure convergence. Following is a brief description of the implementation of each baseline method as presented in [2].

1. **SRCONLY**: Ignores the target data and trains two models, on the source data only.
2. **TGTONLY**: Trains a single model on 100 instances of the target data only.
3. **ALL**: Trains the learning algorithm on the union of the above two datasets.
4. **WEIGHTED (W)**: The 100 target domain instances are repeated to match the number of samples in the source domain. The union of the repeated target data with the source data is used to train the learning algorithm. This is done to balance the weights of the source and target data.
5. **PRED**: Uses the predictions made by the *SRCONLY* model as an additional feature, and a second model is trained on the target data, augmented with this new feature.
6. **LININT**: The predictions of the *SRCONLY* (y_1) and *TGTONLY* (y_2) models are linearly interpolated and the interpolation parameter is adjusted based on the target development data. To achieve this, we train a linear regression model with features y_1 and (y_2 - y_1) and set the weight $w_{y_1} = 1$. This allows the regressor to learn the interpolation parameter w, in the interpolation formula, $\hat{y} = y_1 + w * (y_2 - y_1)$.

The key hyper-parameters that optimize the neural network are hidden layer sizes, h and learning rate, η. Their values are reported in Table 1 for each baseline technique, for when each domain serves as the target domain.

3 Feature Augmentation Approach

The feature augmentation approach essentially takes each feature in the original problem and creates three versions of it: a general, source-specific and a target-specific version. For K domains, the augmented feature space consists of $(K + 1)$ copies of the original feature space [1]. In our case, $K = 3$, implying the augmented feature set will contain 4 copies of the original feature set.

The male, female and mixed augmented feature set is now $< x, x, 0, 0 >$, $< x, 0, x, 0 >$ and $< x, 0, 0, x >$ where x is the vector in the original feature space. Hence, the augmented target data will contain the general and target-specific versions whereas the augmented source data will only hold general and source-specific versions. This method is then evaluated on the Schools dataset (similar to the ALL baseline data condition) with hyper-parameters adjusted as seen in Table 1, using a random search technique as before.

To recognize the effect of how hyper-parameters impact the treatment of weights in the "general" versus out-of-domain, we scale groups of features by a constant while using a uniform regulariser. We replicate the features as $< ax, x, 0, 0 >$, $< ax, 0, x, 0 >$ and $< ax, 0, 0, x >$, where a > 1 and is a scalar. We find that the weights learned by the MLP for the general component will be lower than the domain specific components and since we train with a regulariser, it will effectively be harsher on the domain specific components.

A secondary experiment is also performed to determine the effect of the amount of training data in the target domain on the test error. This experiment assesses MSE values for when 10%, 20%, 50%, 75% and 100% of training data is used in the target domain for each domain. The hyper-parameters are set to default values to make the experiment faster.

4 Domain Adaptation Extension

There are an abundant number of methods towards performing domain adaptation and transfer learning over the last two decades in related literature as seen in [4]. The idea of a model that uses the *SRCONLY* baseline as a *prior* on the weights, for a second model trained on the target data, was proposed in [5].

A boosting algorithm, *TrAdaBoost*, which is an extension of the AdaBoost algorithm was proposed in [6], that can also be applied to our task. However, the rate of convergence of *TrAdaBoost* may be slow and the algorithm cannot deal with multiple different distributions simultaneously.

An effective, and efficient method for unsupervised domain adaptation called **COR**relation **AL**ignment (*CORAL*) that involves a transformation of the feature set was proposed in [7]. The algorithm minimizes the domain shift by aligning the second-order statistics (covariance) of source and target distributions, without demanding target labels. We attempt to experiment with this algorithm on our dataset as if the target response variable were nonexistent. The motivation to follow this approach comes from the fact that *CORAL* is quite novel and results in being empirically superior over many existing state-of-the-art methods. Also, while it may seem *easy* to implement, it is mathematically *complicated*. The algorithm has been formulated through a series of derivations as presented in Section 3.1 and 3.2 of [7]. There are two key parts of the transformation. The first part *whitens* the source data (source decorrelation) while the second part *re-colors* it with the target covariance. After the transformation of the source features to the target space, a regressor/classifier can be trained on the adjusted source features and directly be applied to the target features [7]. Figure 1 illustrates this visually and is sourced from [7].

In our case, the School data feature set is transformed separately for each domain according to the algorithm as presented in [7] and is used to train the two learning algorithms. Hyper-parameters are adjusted accordingly (see Table 1) through a random search technique as before and the experiment is evaluated.

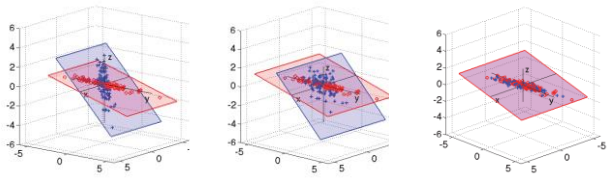

Figure 1: (a) The original source & target distributions have different distribution covariances.
(b) The same two domains after source decorrelation (*Whitening*).
(c) Target re-correlation (*Re-coloring*).

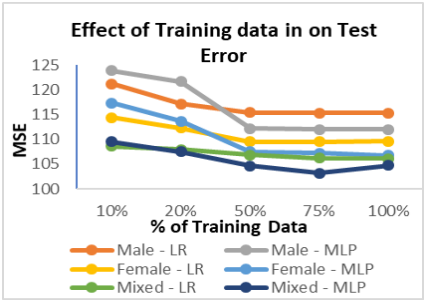

Figure 2: Secondary Experiment showing the effect of amount of training data in target domain (Augmentation Approach)

Table 1 : NN hyper-parameter values for learning rate, η and size of hidden layers, h, for each technique.

Target Domain	SRC		TGT		ALL		W		PRED		AUG		CORAL	
	h	η	h	η	h	η	h	η	h	η	h	η	h	η
M	12	10^{-3}	17	10^{-3}	12	10^{-3}	10	10^{-3}	6	0.0018	14	10^{-3}	101	10^{-3}
F	50	0.015	42	10^{-3}	50	10^{-3}	12	10^{-3}	8	10^{-3}	11	10^{-3}	17	10^{-3}
Mix	104	10^{-3}	12	10^{-3}	13	10^{-3}	12	10^{-3}	6	10^{-3}	6	10^{-3}	99	10^{-3}

5 Analysis & Results

The full table of results is presented in Table 2. The first two columns specify the domain (*Male, Female, Mixed*) and learning algorithm used and subsequent columns give the corresponding Mean Squared Errors (MSEs) for the task using one of the different techniques. For each row, the error rate of the best technique is emboldened (as are those, whose MSEs are not significantly different at the 99% level).

Amongst the 6 **Baseline** techniques, for both the male and female domains, *LININT* is the winner with least MSE values. If we imagine that *TGTONLY* predictions consistently over-estimates the exam score (response variable) and *SRCONLY* predictions consistently under-estimates the exam score, then interpolating between the two is expected to be better than either prediction on its own. However, this is not always the case as an interpolation coefficient that is optimal on the development set might not be on the test set which can explain why the MSE value goes up due to the overfitting on the small development set in the mixed domain.

Also, we observe that *SRCONLY* performs better than *TGTONLY* in all domains. Due to the similarity of the two domains, a large amount of source data would outperform a small amount of target data (100 instances). The *ALL*, *WEIGHTED* and *PRED* baselines perform somewhat similarly to the *SRCONLY* technique in many cases. The *WEIGHTED* baseline is the clear winner in the mixed domain which can be attributed to the fact that since the mixed domain contains instances of both male and female students, with equal weight between source and target domains, it would lead to more accurate predictions as compared to the other domains or baselines.

The **Augmentation** approach performs quite averagely (see Table 2) and is not a by far, clear winner for any domain, overall. Daum´e III hypothesizes that it is unlikely that augmenting the feature space will improve performance when the domains are similar [1]. The augmenting of out of domain data certainly reduces the error rate when compared to *TGTONLY*.

Considering the sensitivity of hyper-parameters in the MLP, it is observed that there is a significant reduction in MSE on finding the optimal values for learning rate (η) and number of hidden layer sizes (h), while there isn't ample *positive* disparity on altering other parameters such as alpha or batch size. The best combinations of hyper-parameters are reported in Table 1.

The secondary experiment performed, determines the effect on Mean Squared Error (MSE), for when 10%, 20%, 50%, 75% and 100% of training data is individually used in the target domain, for each domain and learning algorithm. The result is graphically visualized in Figure 2. We see a pattern showing that the MSE decreases with increasing amounts of in-domain training data until it stagnates to a minimum in most cases.

As seen in Table 2, the ***CORAL*** technique performs the best by far in both the male and female domains as expected but does not perform as remarkably in the mixed domain. We explain this as follows: The mixed domain contains both male and female student instances. Our hypothesis is that the distribution of the source data is expected to be similar to the target data. However, the randomly chosen target training data samples (100 instances) may contain an imbalance in the number of male and female students, which could cause a minor skew in its distribution. *CORAL* ends up removing the (large number of) feature correlations of the source domain and re-aligns it with this (imbalanced) target domain distribution which could lead to a drop in performance in the mixed domain.

Table 2: Mean Squared Error values for each technique by domain and learning algorithm used.

Domain	Algo	SRC	TGT	ALL	W	PRED	LIN	Aug	CORAL
Male	LR	123.51	158.24	123.34	126.50	158.25	121.44	129.54	**104.79**
	MLP	122.31	137.70	122.83	127.28	121.11	121.92	122.68	**103.32**
Female	LR	127.61	134.97	126.86	113.13	135.61	112.74	116.81	**108.46**
	MLP	120.16	141.89	126.47	121.74	120.76	115.63	115.46	**107.52**
Mixed	LR	**108.82**	117.55	119.70	**109.68**	117.53	115.93	112.90	121.66
	MLP	112.20	129.22	115.83	**110.94**	115.30	115.82	**110.81**	115.83

References

[1] H. Daumé, “Frustratingly easy domain adaptation,” *ACL 2007 - Proc. 45th Annu. Meet. Assoc. Comput. Linguist.*, pp. 256–263, 2007.

[2] H. Daumé and D. Marcu, “Domain adaptation for statistical classifiers,” *J. Artif. Intell. Res.*, vol. 26, pp. 101–126, 2006.

[3] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[4] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010.

[5] C. Chelba and A. Acero, “Adaptation of maximum entropy capitalizer: Little data can help a lot,” *Comput. Speech Lang.*, vol. 20, no. 4, pp. 382–399, 2006.

[6] W. Dai, Q. Yang, G. R. Xue, and Y. Yu, “Boosting for transfer learning,” *ACM Int. Conf. Proceeding Ser.*, vol. 227, pp. 193–200, 2007.

[7] B. Sun, J. Feng, and K. Saenko, “Return of frustratingly easy domain adaptation,” *30th AAAI Conf. Artif. Intell. AAAI 2016*, no. 1, pp. 2058–2065, 2016.