COMP90051 Statistical Machine Learning

Assignment 2 - Transfer Learning

Carlos Davalos - 1020724

I. Introduction

In the theoretical world of mathematics, models are developed to solve different types of problems and are based on assumptions to define a basis for analysis that allows the model to be developed in a stable manner, idealizing these assumptions and hoping that they can always and strictly be met. Meanwhile in real life it is not possible to fulfill the model requirements always and alternative measures must be taken to use the models. Moreover, the access to labeled data in certain areas of knowledge is reduced or costly.

The above two problems are addressed by the transfer learning approach and some of this methods will be tested in this document.

II. Transfer Learning

Transfer learning approach consists of transferring knowledge from one or multiple known domains to a target domain. Usually, the target domain dataset is scarce with little or no labeled data, and the source domains contains enough amount of labeled data and may or may not be related to the domain of interest. Based on the characteristics of both of the domains (source and target), several studies have been carried out using various methodologies for the different scenarios.

Delving into the literature, multiple methodologies and different statistical techniques can be found. (Pan & Yang, 2010) summarised the more relevant approaches, and classified the methods into 4 main groups: instance transfer, feature-representation transfer, parameter transfer and relational knowledge transfer.

In this document two methodologies of two main groups will be analyzed. The FEDA method developed in (Daumé-III, 2009) representing the feature-representation approaches, and TrAdaBoost initially developed by (Freund & Schapire, 1997) and later analyzed and improved by (Dai, Yang, Xue, & Yu, 2007) and (Pardoe & Stone, 2010)

(Daumé-III, 2009) addressed the transfer learning problem by taking each feature in the original problem and making three versions of it: a general version, a source-specific version and a target-specific version. Then a base learner is used which will learn and determine the weight (importance) to each of the feature and version of it. If the method assigns big weight to the general version of the feature means that it is important regardless the domain, on the other hand, if it assigns big weight to an specific feature, mean that it is important to determining a specific domain.

Alternatively, (Freund & Schapire, 1997) adapted the AdaBoost approach, in which multiple iterations are run adjusting the weights of the training instances, giving more weight to the misclassified ones, on the assumption that this instances will be more important for the next iteration. The adaptation consisted on filtering out the data from the source domain that does not follow the distribution of the target domain by adjusting automatically the weights of training instances (Dai et al., 2007). TrAdaBoost combines one or more source domains with the target domain information to form a single dataset. At each step of the process, it increases the weight of the target instances that are not being properly classified and decreases the weights of those source domain instances when misclassified. The final purpose is to find those instances from the source domains that are like the target domain ignoring the ones that does not.

Besides this, (Yao & Doretto, 2010) improves the TrAdaBoost, allowing the algorithm to analyze multiple source domains separately, in a method called MultiSourceTrAdaBoost. At this approach, in every iteration a weak classifier in trained on each available source domain and the one more closely related with the target domain (domain that minimizes the target classification error) is selected, and re weighting each instance separately.

The last two approaches about TrAdaBoost mentioned above were built for the classification problem. Given that, (Pardoe & Stone, 2010) extended the approach for regressors, with TrAdaBoost.R2 where the author allows the inclusion of multiple sources, and taking into account that the weight of each instance is handled separately the author warns that the initial weight of each instance must reflect the inclusion of multiple sources and is important for the final results. Furthermore, thr author determines TrAdaBoost.R2 is prone to overfitting, so a Two-stage TrAdaBoost.R2 is also implemented where the weights are adjusted in two stages. In the first stage, the weights of source instances are adjusted gradually, ans then in the second stage weights of all source instances are frozen while the weights of target instances are updated as normal in AdaBoost.R2

III. Evaluation

A. Data set Details

In this document the dataset in (Evgeniou, Micchelli, & Pontil, 2005) will be used, where three domains exist: female, male and mixed marks. At each iteration one of the domains will be left out treating it as the target domain while the other two sources will be as the source domains.

The fields 'Year', 'VR Band of Student', 'Ethnic group of student' and 'School denomination' are modeled as categorical variables so a one-hot encoding approach is used, while the remaining 'FSM' and 'VR1 Band' fields as numerical variables.

B. Baselines

The same approach as in (Daumé-III, 2009) will be used to create a set of baselines with which FEDA and TrAdaBoost can be compared. The following models were implemented: ALL, LININT, PRED, SRCONLY, TGTONLY and WEIGHTED.

It should be noted that some of these baseline models use the brute force practice to get anything the source domain can offer, which could lead to a negative transfer and therefore affecting the predictability of the model.

C. Algorithms

Initially the following algorithms were defined for all the approaches, given their different approach to the data: Lasso, Neural Network and Support Vector Machines

After several tests with each method, it was defined that these three methods will be used for the baseline and the FEDA. However, for TrAdaBoosting the Neural Network and Support Vector Machines required a lot of computational time, making their implementation and evaluation difficult. It was decided to use the two following method for the TrAdaBoosting approach: Lasso and Decision Tree Regressor

With the help of the scikit-learn ((Pedregosa et al., 2011)), a python module, the models were implemented and evaluated.

D. Data split, Hyper parameters and Cross Validation

A 70-20-10 percentage approach, to determine the number of samples for the train, development and test data sets respectively.

For each of the above algorithms there are several parameters that need to be tune given a dataset. Due to this and given the computational cost, it was decided that only certain parameters will be tuned:

- Lasso: C
- Neural Network: hidden layer sizes, alpha, learning rate and batch size
- Support Vector Machine: epsilon, C and kernel
- Decision Tree Regressor: max depth, min samples leaf and min samples split

The Cross-Validation procedure was used to determine some appropriate hyper parameters based on the mean square error as measurement of performance of the models. 50 iterations for randomly picking the different hyper parameters were run and a 20-fold approach, was used to evaluate each selection of hyper-parameters and get the mean performance in the development dataset.

IV. Analysis

A. Exploratory Data Analysis

Given the number of variables, it was decided to carry out a Principal Components analysis, with the purpose of being able to graph the first two components (components with greater variability).

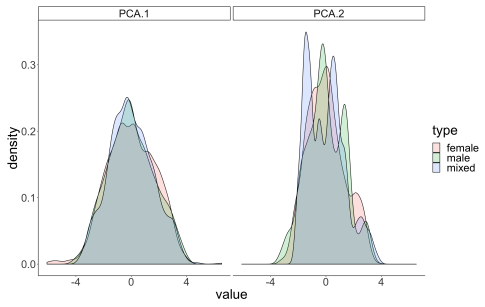

Fig. 1. Covariates PCA histogram

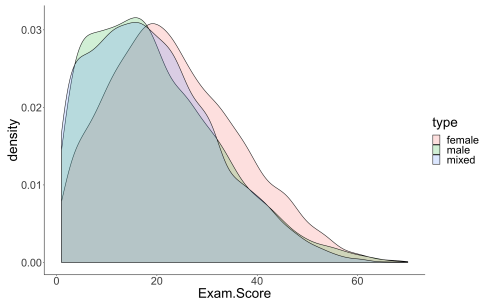

Fig. 2. Response histogram

In both, principal components Fig. 1 and the response variable Fig. 2 histograms, it is observed that the three domains have a similar behavior, with a difference in the female dataset, where the response variable is slightly skewed to the right.

B. Baselines and FEDA

All the baselines are run together with the FEDA algorithm, where the models are evaluated in the development dataset with 100 instances randomly picked. In 3 the results are presented.

Algorithm	Dominium	SRCONLY	TGTONLY	ALL	WEIGHT	PRED	LININT	Augment	TrAdaBoost
Female	Lasso	101	117	124	116	111	123	124	122
	NN	150	**95**	123	115	103	121	115	
	SVM	182	114	127	132	107	121	109	
	DT								125
Male	Lasso	**103**	117	133	146	124	138	118	127
	NN	168	204	130	135	124	134	127	
	SVM	141	116	130	128	126	131	116	
	DT								151
Mixed	Lasso	**81**	106	110	123	116	111	101	110
	NN	144	92	112	106	113	107	127	
	SVM	115	100	108	114	115	107	109	
	DT								125

Fig. 3. Summary Baseline and FEDA: Mean MSE

The augmented model does not perform better than the baselines, where the female dataset has the lowest mse with the TGTONLY-NN model, and male and mixed with the SRCONLY-Lasso model. Joining the findings in the EDA, together with these results, it can be concluded that the domains, having similar distributions, can become a good replacement for the other two domains as training data in the absence of the adequate amount of the target domain. That is why the augmented model could not improve the performance of baselines, which is intended to be applied to data sets with significant differences, as stated by (Daumé-III, 2009) when

concluding that his algorithm works so well on domains that are actually quite well separated.

C. Hyper-parameter sensitivity

For each of the implemented models, the most relevant hyper-parameter(s) are chosen and a sensitivity analysis is performed based on the performed iterations.

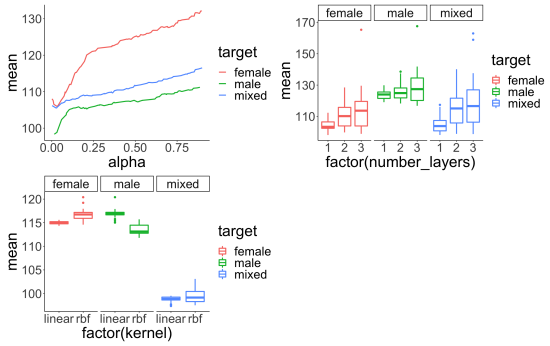

Fig. 4. 1. Lasso - alpha, 2. NN - hidden layer, 3. SVM - kernel

In Fig. 4, the first plot evidences that the three domains a small alpha can be set to a small value or in fact suggest to change the model to a Linear Regression. The second plot, indicates that for the Neural Network, one layer will be enough for the three domains and also shows that 'female' and 'mixed' domains manages smaller mse than the 'male' domain. Finally, the third plot for the SVM, shows that for 'female' domain a linear kernel will be the best option while for 'male' and 'mixed' domains an rbf configuration should be used. Additionally, the batch size and the learning rate are analyzed separately from the above hyperparameters as below.In general, a learning rate of 0.1 and a batch size between 200 and 400 are the ones who offer the best performance.

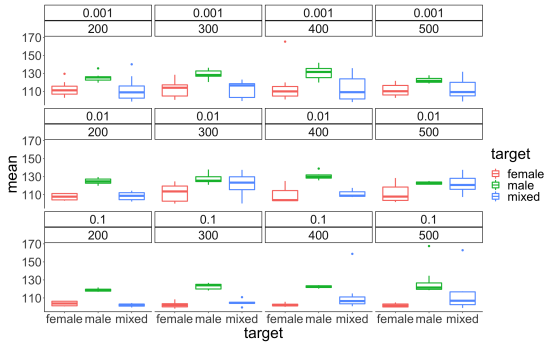

Fig. 5. Neural Network - learning rate and batch size

Observing the two graphs above Fig. 4 and Fig. 5, it can be concluded that by adjusting the appropriate hyperparameters, the configuration that best suits each domain is as follows

- Female: Lasso / Linear Regression (alpha = 0) or Neural Network with one layer, learning rate 0.1 and batch size between 200 and 400
- Male: Lasso / Linear Regression (alpha = 0)
- Mixed: SVM with linear kernel or Neural Network with one layer, learning rate 0.1 and batch size between 200 and 400

D. Ratio Target-Domain sensitivity

In theory, if we add more information from the target domain to the training set, we must have a more adequate fit, allowing us to obtain more precise estimates and predictions. Therefore, a sensitivity analysis of the percentage of data corresponding to the target domain with respect to the total data is performed.

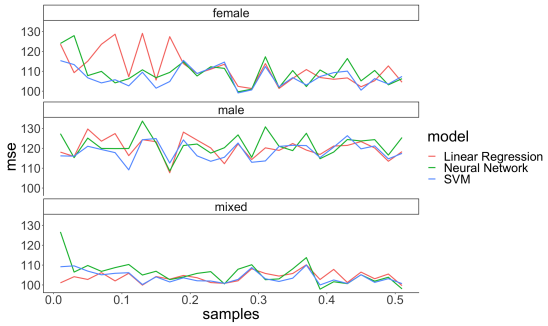

Fig. 6. FEDA: mse vs ratio target/source domains

For the mixed and male domains, there is not much change when more data is added to the training data meaning that the source domains are enough for training the models. In the female domain a change is noticed after the proportion of data is more than the 0.2 of the whole dataset.

E. TrAdaBoost

The code implemented in Github by (Ren, 2018) is used to evaluate the performance of the model TwoStageTrAdaBoostR2 developed by (Pardoe & Stone, 2010) in the datasets.

Since this method depends on a learner base, the same cross validation approach is used to determine the most suitable hyperparameters for the learner base. Moreover, a 10-fold cross validation is used for the estimation of the weights of the instances in the algorithm and a learning rate of 0.01.

The initialization of the weights is of utmost importance for the method, since with the initial weights each instance will reflect the amount of data that each domain has in the training set, and will help the method converge to the optimal weight for each instance.

The performance of TrAdaBoost in Fig. 4, shows that it did bot did not improve the results of the baseline or the augmented method, in fact it was one of the worst performing models. These results can be given for multiple reasons, such as: the base learners were not tuned properly, the initial weights were not adequate, among others. It is also important to mention that the authors used datasets with notable differences between the domains, while in this document the dataset has quite similarities between them.

V. Conclusion and Next Steps

The three datasets have a similar distribution allowing to be source domains for the other two datasets in case these do not have enough information, without having to resort to a complex model in order to obtain a good estimate. The FEDA model stands out in those cases where the datasets differ markedly and therefore in this case it could not stand out.

REFERENCES

Dai, W., Yang, Q., Xue, G.-R., & Yu, Y. (2007). Boosting for transfer learning. In *Proceedings of the 24th international conference on machine learning* (p. 193–200). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/1273496.1273521 doi: 10.1145/1273496.1273521

Daumé-III, H. (2009). *Frustratingly easy domain adaptation.*

Evgeniou, T., Micchelli, C. A., & Pontil, M. (2005, December). Learning multiple tasks with kernel methods. *J. Mach. Learn. Res.*, *6*, 615–637.

Freund, Y., & Schapire, R. E. (1997, August). A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, *55*(1), 119–139. Retrieved from https://doi.org/10.1006/jcss.1997.1504 doi: 10.1006/jcss.1997.1504

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345-1359.

Pardoe, D., & Stone, P. (2010). Boosting for regression transfer. In *Proceedings of the 27th international conference on international conference on machine learning* (p. 863–870). Madison, WI, USA: Omnipress.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Ren, J. (2018, June). *Two-stage TrAdaBoost.R2 algorithm from Pardoe's paper "Boosting for Regression Transfer (ICML 2010)".* GitHub. Retrieved from https://github.com/jay15summer/Two-stage-TrAdaboost.R2 doi: jay15summer

Yao, Y., & Doretto, G. (2010). Boosting for transfer learning with multiple sources. In *2010 ieee computer society conference on computer vision and pattern recognition* (p. 1855-1862).