Frustratingly Easy Semi-Supervised Domain Adaptation

Hal Daumé III
School Of Computing
University of Utah
hal@cs.utah.edu

Abhishek Kumar
School Of Computing
University of Utah
abhik@cs.utah.edu

Avishek Saha
School Of Computing
University of Utah
avishek@cs.utah.edu

Abstract

In this work, we propose a semi-supervised extension to a well-known supervised domain adaptation approach (EA) (Daumé III, 2007). Our proposed approach (EA++) builds on the notion of augmented space (introduced in EA) and harnesses unlabeled data in target domain to ameliorate the transfer of information from *source* to *target*. This semi-supervised approach to domain adaptation is extremely simple to implement, and can be applied as a pre-processing step to any supervised learner. Experimental results on sequential labeling tasks demonstrate the efficacy of the proposed method.

1 Introduction

A domain adaptation approach for sequential labeling tasks in NLP was proposed in (Daumé III, 2007). The proposed approach, termed EASYADAPT (EA), augments the *source domain* feature space using features from labeled data in *target domain*. EA is simple, easy to extend and implement as a preprocessing step and most importantly is agnostic of the underlying classifier. However, EA requires labeled data in the target and hence applies to *fully supervised* (labeled data in *source* and *target*) domain adaptation settings *only*. In this paper, we propose a *semi-supervised*[1] (labeled data in *source*, and both labeled and unlabeled data in *target*) approach to leverage unlabeled data for EASYADAPT (which we call EA++) and empirically demonstrate its superior performance over EA as well as few other existing approaches.

[1]We refer, labeled data in source and *only* unlabeled data in target, as the *unsupervised* domain adaptation setting.

There exists prior work on supervised domain adaptation (or multi-task learning) that can be related to EASYADAPT. An algorithm for multi-task learning using shared parameters was proposed (Evgeniou and Pontil, 2004) for multi-task regularization where each task parameter was represented as sum of a mean parameter (that stays same for all tasks) and its deviation from this mean. SVM was used as the base classifier and the algorithm was formulated in the standard SVM dual optimization setting. Subsequently, this framework (Evgeniou and Pontil, 2004) was extended (Dredze et al., 2010) to online multi-domain setting. Prior work on semi-supervised approaches to domain adaptation also exists in literature. Extraction of specific features from the available dataset was proposed (Arnold and Cohen, 2008; Blitzer et al., 2006) to facilitate the task of domain adaptation. Co-adaptation (Tur, 2009), a combination of co-training and domain adaptation, can also be considered as a semi-supervised approach to domain adaptation. A semi-supervised EM algorithm for domain adaptation was proposed in (Dai et al., 2007). Similar to graph based semi-supervised approaches, a label propagation method was proposed (Xing et al., 2007) to facilitate domain adaptation. The recently proposed Domain Adaptation Machine (DAM) (Duan et al., 2009) is a semi-supervised extension of SVMs for domain adaptation and presents extensive empirical results. However, in almost all of the above cases, the proposed methods either use specifics of the datasets or are customized for some particular base classifier and hence it is not clear how the proposed methods can be extended to other existing classifiers.

EA, on the other hand, is remarkably general in the sense that it can be used as a pre-processing

Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing, ACL 2010, pages 53–59,
Uppsala, Sweden, 15 July 2010.

step in conjunction with any base classifier. However, one of the prime limitations of EA is its incapability to leverage unlabeled data. Given its simplicity and generality, it would be interesting to extend EA to semi-supervised settings. In this paper we propose EA++, a co-regularization based semi-supervised extension to EA. We present our approach and results for a single pair of source and target domain. However, we note that EA++ can also be extended to multiple source settings. If we have k sources and a single target domain then we can introduce a co-regularizer for each source-target pair. Due to space constraints, we defer details to a full version.

2 Background

2.1 Problem Setup and Notations

Let $\mathcal{X} \subset \mathbb{R}^d$ denote the instance space and $\mathcal{Y} = \{-1, +1\}$ denote the label space. We have a set of source labeled examples $L_s(\sim \mathcal{D}_s(x, y))$ and a set of target labeled examples $L_t(\sim \mathcal{D}_t(x, y))$, where $|L_s| = l_s \gg |L_t| = l_t$. We also have target unlabeled data denoted by $U_t(\sim \mathcal{D}_t(x))$, where $|U_t| = u_t$. Our goal is to learn a hypothesis $h : \mathcal{X} \mapsto \mathcal{Y}$ having low expected error with respect to the target domain. In this paper, we consider *linear hypotheses* only. However, the proposed techniques extend to non-linear hypotheses, as mentioned in (Daumé III, 2007). Source and target empirical errors for hypothesis h are denoted by $\hat{\epsilon}_s(h, f_s)$ and $\hat{\epsilon}_t(h, f_t)$ respectively, where f_s and f_t are source and target labeling functions. Similarly, the corresponding expected errors are denoted by $\epsilon_s(h, f_s)$ and $\epsilon_t(h, f_t)$. Shorthand notions of $\hat{\epsilon}_s$, $\hat{\epsilon}_t$, ϵ_s and ϵ_t have also been used.

2.2 EasyAdapt (EA)

In this section, we give a brief overview of EASYADAPT proposed in (Daumé III, 2007). Let us denote $\mathbb{R}^d$ as the *original* space. EA operates in an *augmented* space denoted by $\breve{\mathcal{X}} \subset \mathbb{R}^{3d}$ (for a single pair of source and target domain). For k domains, the *augmented* space blows up to $\mathbb{R}^{(k+1)d}$. The augmented feature maps $\Phi^s, \Phi^t : \mathcal{X} \mapsto \breve{\mathcal{X}}$ for source and target domains are defined as,

$$\begin{aligned} \Phi^s(\mathbf{x}) &= \langle \mathbf{x},\, \mathbf{x},\, \mathbf{0} \rangle \\ \Phi^t(\mathbf{x}) &= \langle \mathbf{x},\, \mathbf{0},\, \mathbf{x} \rangle \end{aligned} \tag{2.1}$$

where $\mathbf{x}$ and $\mathbf{0}$ are vectors in $\mathbb{R}^d$, and $\mathbf{0}$ denotes a zero vector of dimension d. The first d-dimensional segment corresponds to commonality between source and target, second d-dimensional segment corresponds to the source domain while the last segment corresponds to the target domain. Source and target domain features are transformed using these feature maps and the augmented feature space so constructed is passed onto the underlying supervised classifier. One of the most appealing properties of EASYADAPT is that it is agnostic of the underlying supervised classifier being used to learn in the *augmented* space. Almost any *standard supervised learning approach for linear classifiers* (for e.g., SVMs, perceptrons) can be used to learn a *linear hypothesis* $\breve{\mathbf{h}} \in \mathbb{R}^{3d}$ in the augmented space. As mentioned earlier, this work considers linear hypotheses only and the the proposed techniques can be extended (Daumé III, 2007) to non-linear hypotheses. Let us denote $\breve{\mathbf{h}} = \langle \mathbf{h_c},\, \mathbf{h_s},\, \mathbf{h_t} \rangle$, where each of $\mathbf{h_c}$, $\mathbf{h_s}$, $\mathbf{h_t}$ is of dimension d and represent the *common*, *source-specific* and *target-specific* components of $\breve{\mathbf{h}}$, respectively. During prediction on target data, the incoming target feature $\mathbf{x}$ is transformed to obtain $\Phi^t(\mathbf{x})$ and $\breve{\mathbf{h}}$ is applied on this transformed feature. This is equivalent to applying $(\mathbf{h_c} + \mathbf{h_t})$ on $\mathbf{x}$.

A good intuitive insight into why this simple algorithm works so well in practice and outperforms most state-of-the-art algorithms is given in (Daumé III, 2007). Briefly, it can be thought to be simultaneously training two hypotheses: $\mathbf{w_s} = (\mathbf{h_c} + \mathbf{h_s})$ for source domain and $\mathbf{w_t} = (\mathbf{h_c} + \mathbf{g_t})$ for target domain. The commonality between the domains is represented by $\mathbf{h_c}$ whereas the source and target domain specific information is captured by $\mathbf{h_s}$ and $\mathbf{h_t}$, respectively. This technique can be easily extended to a multi-domain scenario by making more copies of the original feature space ($(K + 1)$ copies in case of K domains). A kernelized version of the algorithm has also been presented in (Daumé III, 2007).

3 Using Unlabeled data

As discussed in the previous section, the EASYADAPT algorithm is attractive because it performs very well empirically and can be used in conjunction with any underlying supervised clas-

sifier. One drawback of EASYADAPT is that it does not make use of unlabeled target data which is generally available in large quantity in most practical problems. In this section, we propose a semi-supervised extension of this algorithm while maintaining the desirable classifier-agnostic property.

3.1 Motivation

In multi-view approach for semi-supervised learning algorithms (Sindhwani et al., 2005), different hypotheses are learned in different *views*. Thereafter, unlabeled data is utilized to co-regularize these learned hypotheses by making them agree on unlabeled samples. In domain adaptation, the source and target data come from two different distributions. However, if the source and target domains are *reasonably close* to each other, we can employ a similar form of regularization using unlabeled data. A similar co-regularizer based approach for unlabeled data was previously shown (Duan et al., 2009) to give improved empirical results for domain adaptation task. However, their technique applies for the particular base classifier they consider and hence does not extend to EASYADAPT.

3.2 EA++: EASYADAPT with unlabeled data

In our proposed semi-supervised extension to EASYADAPT, the source and target hypothesis are made to agree on unlabeled data. We refer to this algorithm as EA++. Recall that EASYADAPT learns a linear hypothesis $\breve{\mathbf{h}} \in \mathbb{R}^{3d}$ in the *augmented* space. The hypothesis $\breve{\mathbf{h}}$ contains common, source and target sub-hypotheses and is expressed as $\breve{\mathbf{h}} = \langle \mathbf{h_c}, \mathbf{h_s}, \mathbf{h_t} \rangle$. In *original* space (ref. section 2.2), this is equivalent to learning a source specific hypothesis $\mathbf{w_s} = (\mathbf{h_c} + \mathbf{h_s})$ and a target specific hypothesis $\mathbf{w_t} = (\mathbf{h_c} + \mathbf{h_t})$.

In EA++, we want source hypothesis $\mathbf{w_s}$ and target hypothesis $\mathbf{w_t}$ to agree on unlabeled data. For some unlabeled target sample $\mathbf{x_i} \in \mathcal{U}_t \subset \mathbb{R}^d$, EA++ would implicitly want to make the predictions of $\mathbf{w_t}$ and $\mathbf{w_t}$ on $\mathbf{x_i}$ to agree. Formally, it aims to achieve the following condition:

$$\begin{aligned} & \mathbf{w_s} \cdot \mathbf{x_i} \approx \mathbf{w_t} \cdot \mathbf{x_i} \\ \iff & (\mathbf{h_c} + \mathbf{h_s}) \cdot \mathbf{x_i} \approx (\mathbf{h_c} + \mathbf{h_t}) \cdot \mathbf{x_i} \\ \iff & (\mathbf{h_s} - \mathbf{h_t}) \cdot \mathbf{x_i} \approx 0 \\ \iff & \langle \mathbf{h_c}, \mathbf{h_s}, \mathbf{h_t} \rangle \cdot \langle \mathbf{0}, \mathbf{x_i}, -\mathbf{x_i} \rangle \approx 0. \end{aligned} \tag{3.1}$$

We define another feature map $\Phi^u : X \mapsto \tilde{X}$ for unlabeled data as below:

$$\Phi^u(\mathbf{x}) = \langle \mathbf{0}, \mathbf{x}, -\mathbf{x} \rangle. \tag{3.2}$$

Every unlabeled sample is transformed using the map $\Phi^u(.)$. The augmented feature space that results from the application of three feature maps, namely, $\Phi^s : X \mapsto \breve{X}, \Phi^t : X \mapsto \breve{X}, \Phi^u : X \mapsto \breve{X}$, on source labeled samples, target labeled sampled and target unlabeled samples is summarized in Figure 1.

As shown in Eq. 3.1, during the training phase, EA++ assigns a predicted value close to 0 for each unlabeled sample. However, it is worth noting that, during the test phase, EA++ predicts labels from two classes: $+1$ and -1. This warrants further exposition of the implementation specifics which is deferred until the next subsection.

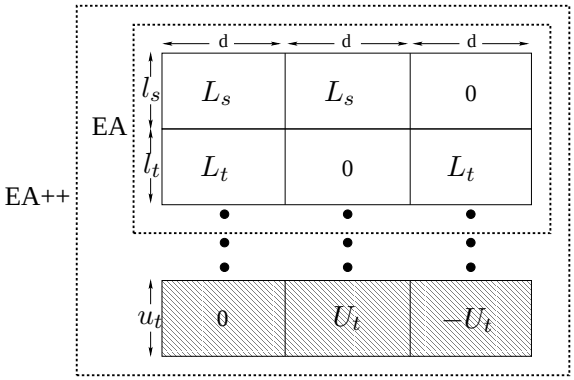


Figure 1: Diagrammatic representation of feature augmentation in EA and EA++

Algorithm 1 presents the EA++ approach in detail.

3.3 Implementation

In this section, we present implementation specific details of EA++. We consider SVM as our base supervised learner ($\mathcal{LEARN}$ in Algorithm 1). However, these details hold for other supervised

Algorithm 1 EA++

Input: L_s; L_t; U_t; $\mathcal{LEARN}$: supervised classifier
Output: $\breve{h}$: classifier learned in augmented space
/* initialize augmented training set */
1: $P := \{\}$
/* construct augmented training set */
2: $\forall(\mathbf{x}, y) \in L_s$, $P := P \cup \{\Phi^s(\mathbf{x}), y\}$
3: $\forall(\mathbf{x}, y) \in L_t$, $P := P \cup \{\Phi^t(\mathbf{x}), y\}$
4: $\forall\mathbf{x} \in U_t$, $P := P \cup \{\Phi^u(\mathbf{x}), 0\}$
/* output learned classifier */
5: $\breve{h} = \mathcal{LEARN}(P)$

classifiers too. In the dual form of SVM optimization function, the labels are multiplied with the inner product of features. This can make the unlabeled samples redundant since we want their labels to be 0 according to Eq. 3.1. To avoid this, we create as many copies of $\Phi^u(\mathbf{x})$ as there are labels and assign each label to one copy. For the case of binary classification, we create two copies of every augmented unlabeled sample, and assign $+1$ label to one copy and -1 to the other. The learner attempts to balance the loss of the two copies, and tries to make the prediction on unlabeled sample equal to 0. Figure 2 shows the curves of the hinge loss for class $+1$, class -1 and their sum. The effective loss for each unlabeled sample is similar to the sum of losses for $+1$ and -1 classes (shown in Figure 2c).

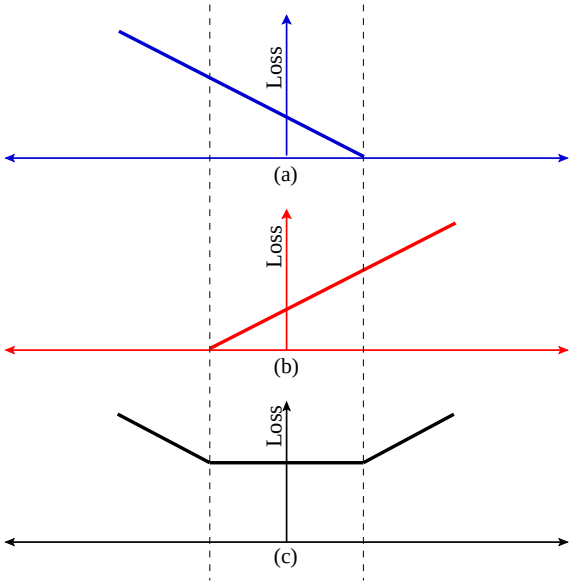


Figure 2: Loss functions for class $+1$, class -1 and unlabeled samples.

4 Experiments

In this section, we demonstrate the empirical performance of EA augmented with unlabeled data.

4.1 Setup

We follow the same experimental setup used in (Daumé III, 2007) and perform two sequence labelling tasks (a) named-entity-recognition (NER), and (b) part-of-speech-tagging (POS)on the following datasets:

PubMed-POS: Introduced by (Blitzer et al., 2006), this dataset consists of two domains. The WSJ portion of the Penn Treebank serves as the source domain and the PubMed abstracts serve as the target domain. The task is to perform part-of-speech tagging on unlabeled PubMed abstracts with a classifier trained on labeled WSJ and PubMed data.

Treebank-Brown. Treebank-Chunk data consists of the following domains: the standard WSJ domain (the same data as for CoNLL 2000), the ATIS switchboard domain and the Brown corpus. The Brown corpus consists of data combined from six subdomains. Treebank-Chunk is a shallow parsing task based on the data from the Penn Treebank. Treebank-Brown is identical to the Treebank-Chunk task, However, in Treebank-Brown we consider all of the Brown corpus to be a single domain.

Table 1 presents a summary of the datasets used. All datasets use roughly the same feature set which are lexical information (words, stems, capitalization, prefixes and suffixes), membership on gazetteers, etc. We use an averaged perceptron classifier from the Megam framework (implementation due to (Daumé III, 2004)) for all the aforementioned tasks. The training sample size varies from $1k$ to $16k$. In all cases, the amount of unlabeled target data was equal to the total amount of labeled source and target data.

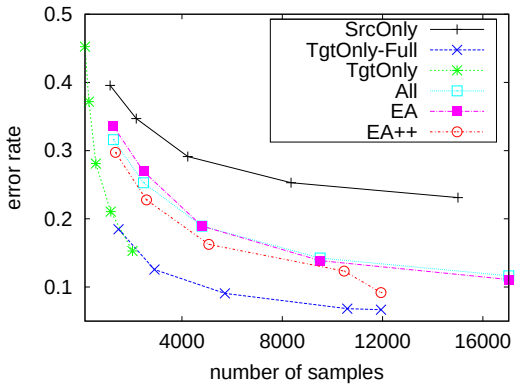
Task	Dom	#Tr	#De	#Te	#Ft
PubMed	src	950,028	-	-	571k
POS	tgt	11,264	1,987	14,554	39k
	wsj	191,209	29,455	38,440	94k
	swbd3	45,282	5,596	41,840	55k
	br-cf	58,201	8,307	7,607	144k
Tree	br-cg	67,429	9,444	6,897	149k
bank-	br-ck	51,379	6,061	9,451	121k
Chunk	br-cl	47,382	5,101	5,880	95k
	br-cm	11,696	1,324	1,594	51k
	br-cn	56,057	6,751	7,847	115k
	br-cp	55,318	7,477	5,977	112k
	br-cr	16,742	2,522	2,712	65k

Table 1: Summary of Datasets. The columns denote task, domain, size of training, development and test data sets, and the number of unique features in the training data.

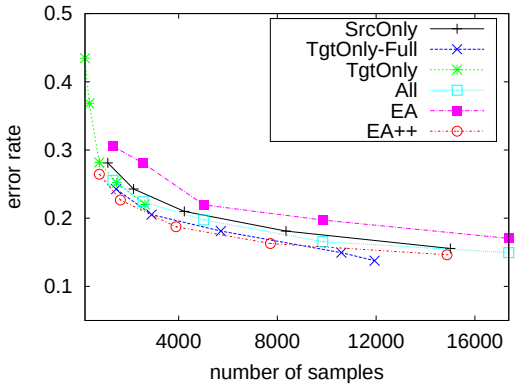
4.2 Results

We compare the empirical performance of EA++ with a few other baselines, namely, (a) SOURCEONLY (classifier trained on source labeled samples), (b) TARGETONLY-FULL (classifier trained on the same number of target labeled samples as the number of source labeled samples in SOURCEONLY), (c) TARGETONLY (classifier trained on small amount of target labeled samples, roughly one-tenth of the amount of source labeled samples in SOURCEONLY), (d) ALL (classifier trained on combined labeled samples of SOURCEONLY and TARGETONLY), (e) EA (classifier trained in *augmented feature space* on the same input training set as ALL), (f) EA++ (classifier trained in *augmented feature space* on the same input training set as EA and an equal amount of unlabeled *target* data). All these approaches were tested on the entire amount of available *target* test data.

Figure 3 presents the learning curves for (a) SOURCEONLY, (b) TARGETONLY-FULL, (c) TARGETONLY, (d) ALL, (e) EA, and (f) EA++ (EA with unlabeled data). The x-axis represents the number of training samples on which the predictor has been trained. At this point, we note that the number of training samples vary depending on the particular approach being used. For SOURCEONLY, TARGETONLY-FULL and TARGETONLY, it is just the corresponding number of labeled source or target samples, respectively. For ALL and EA, it is the summation of labeled source and target samples. For



(a)



(b)

Figure 3: Test accuracy of (a) PubMed-POS and (b) Treebank-Brown for, SOURCEONLY, TARGETONLY-FULL, TARGETONLY, ALL, EA and EA++.

EA++, the x-value plotted denotes the amount of unlabeled target data used (in addition to an equal amount of source+target labeled data, as in ALL or EA). We plot this number for EA++, just to compare its improvement over EA when using an additional (and equal) amount of unlabeled target data. This accounts for the different x values plotted for the different curves. In all cases, the y-axis denotes the error rate.

As can be seen in Figure 3(a), EA++ performs better than the normal EA (which uses labeled data only). The labeled and unlabeled case start together but with increase in number of samples their gap increases with the unlabeled case resulting in much lower error as compared to the labeled case. Similar trends were observed in other data sets as can be seen in Figure 3(b). We also note that EA performs poorly for some cases, as was

shown (Daumé III, 2007) earlier.

5 Summary

In this paper, we have proposed a semi-supervised extension to an existing domain adaptation technique (EA). Our approach EA++, leverages the unlabeled data to improve the performance of EA. Empirical results demonstrate improved accuracy for sequential labeling tasks performed on standardized datasets. The previously proposed EA could be applied exclusively to *fully supervised* domain adaptation problems only. However, with the current extension, EA++ applies to both *fully supervised* and *semi-supervised* domain adaptation problems.

6 Future Work

In both EA and EA++, we use features from source and target space to construct an augmented feature space. In other words, we are sharing features across source and target *labeled* data. We term such algorithms as *Feature Sharing Algorithms*. Feature sharing algorithms are effective for domain adaptation because they are simple, easy to implement as a preprocessing step and outperform many existing state-of-the-art techniques (shown previously for domain adaptation (Daumé III, 2007)). However, despite their simplicity and empirical success, it is not theoretically apparent why these algorithms perform so well. Prior work provides some intuitions but is mostly empirical and a formal theoretical analysis to justify FSAs (for domain adaptation) is clearly missing. Prior work (Maurer, 2006) analyzes the multi-task regularization approach (Evgeniou and Pontil, 2004) (which is related to EA) but they consider a cumulative loss in multi-task (or multi-domain) setting. This does not apply to domain adaptation setting where we are mainly interested in loss in the target domain *only*.

Theoretically analyzing the superior performance of EA and EA++ and providing generalization guarantees is an interesting line of future work. One approach would be to model the feature sharing approach in terms of co-regularization; an idea that originated in the context of multiview learning and for which some theoretical analysis has already been done (Rosenberg and Bartlett, 2007; Sindhwani and Rosenberg, 2008). Additionally, the aforementioned techniques, namely, SOURCEONLY, TARGETONLY, ALL have been empirically compared to EA and EA++. It would be interesting to formally frame these approaches and see whether their empirical performance can be justified within a theoretical framework.

References

Andrew Arnold and William W. Cohen. 2008. Intra-document structural frequency features for semi-supervised domain adaptation. In *CIKM'08*, pages 1291–1300, Napa Valley, California, USA.

John Blitzer, Ryan Mcdonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP'06*, pages 120–128, Sydney, Australia.

Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. 2007. Transferring Naive Bayes classifiers for text classification. In *AAAI'07*, pages 540–545, Vancouver, B.C.

Hal Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. August.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *ACL'07*, pages 256–263, Prague, Czech Republic.

Mark Dredze, Alex Kulesza, and Koby Crammer. 2010. Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79.

Lixin Duan, Ivor W. Tsang, Dong Xu, and Tat-Seng Chua. 2009. Domain adaptation from multiple sources via auxiliary classifiers. In *ICML'09*, pages 289–296, Montreal, Quebec.

Theodoros Evgeniou and Massimiliano Pontil. 2004. Regularized multitask learning. In *KDD'04*, pages 109–117, Seattle, WA, USA.

Andreas Maurer. 2006. The Rademacher complexity of linear transformation classes. In *COLT'06*, pages 65–78, Pittsburgh, Pennsylvania.

D. S. Rosenberg and P. L. Bartlett. 2007. The Rademacher complexity of co-regularized kernel classes. In *AISTATS'07*, San Juan, Puerto Rico.

Vikas Sindhwani and David S. Rosenberg. 2008. An RKHS for multi-view learning and manifold co-regularization. In *ICML'08*, pages 976–983, Helsinki, Finland.

Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. 2005. A co-regularization approach to semi-supervised learning with multiple views. In *ICML Workshop on Learning with Multiple Views*, pages 824–831, Bonn, Germany.

Gokhan Tur. 2009. Co-adaptation: Adaptive co-training for semi-supervised learning. In *ICASSP'09*, pages 3721–3724, Taipei, Taiwan.

Dikan Xing, Wenyuan Dai, Gui-Rong Xue, and Yong Yu. 2007. Bridged refinement for transfer learning. In *PKDD'07*, pages 324–335, Warsaw, Poland.