Learning to Predict Distributions of Words Across Domains

Danushka Bollegala
Department of Computer Science
University of Liverpool
Liverpool,
L69 3BX, UK
danushka.bollegala@liverpool.ac.uk

David Weir
Department of Informatics
University of Sussex
Falmer, Brighton,
BN1 9QJ, UK
d.j.weir@sussex.ac.uk

John Carroll
Department of Informatics
University of Sussex
Falmer, Brighton,
BN1 9QJ, UK
j.a.carroll@sussex.ac.uk

Abstract

Although the distributional hypothesis has been applied successfully in many natural language processing tasks, systems using distributional information have been limited to a single domain because the distribution of a word can vary between domains as the word's predominant meaning changes. However, if it were possible to *predict* how the distribution of a word changes from one domain to another, the predictions could be used to adapt a system trained in one domain to work in another. We propose an unsupervised method to predict the distribution of a word in one domain, given its distribution in another domain. We evaluate our method on two tasks: cross-domain part-of-speech tagging and cross-domain sentiment classification. In both tasks, our method significantly outperforms competitive baselines and returns results that are statistically comparable to current state-of-the-art methods, while requiring no task-specific customisations.

1 Introduction

The Distributional Hypothesis, summarised by the memorable line of Firth (1957) – *You shall know a word by the company it keeps* – has inspired a diverse range of research in natural language processing. In such work, a word is represented by the distribution of other words that co-occur with it. Distributional representations of words have been successfully used in many language processing tasks such as entity set expansion (Pantel et al., 2009), part-of-speech (POS) tagging and chunking (Huang and Yates, 2009), ontology learning (Curran, 2005), computing semantic textual similarity (Besançon et al., 1999), and lexical inference (Kotlerman et al., 2012).

However, the distribution of a word often varies from one domain[1] to another. For example, in the domain of portable computer reviews the word *lightweight* is often associated with positive sentiment bearing words such as *sleek* or *compact*, whereas in the movie review domain the same word is often associated with negative sentiment-bearing words such as *superficial* or *formulaic*. Consequently, the distributional representations of the word *lightweight* will differ considerably between the two domains. In this paper, given the distribution $\boldsymbol{w}_{\mathcal{S}}$ of a word w in the source domain $\mathcal{S}$, we propose an unsupervised method for *predicting* its distribution $\boldsymbol{w}_{\mathcal{T}}$ in a different target domain $\mathcal{T}$.

The ability to predict how the distribution of a word varies from one domain to another is vital for numerous adaptation tasks. For example, unsupervised cross-domain sentiment classification (Blitzer et al., 2007; Aue and Gamon, 2005) involves using sentiment-labeled user reviews from the source domain, and unlabeled reviews from both the source and the target domains to learn a sentiment classifier for the target domain. Domain adaptation (DA) of sentiment classification becomes extremely challenging when the distributions of words in the source and the target domains are very different, because the features learnt from the source domain labeled reviews might not appear in the target domain reviews that must be classified. By predicting the distribution of a word across different domains, we can find source domain features that are similar to the features in target domain reviews, thereby reducing the mismatch of features between the two domains.

We propose a two-step unsupervised approach to predict the distribution of a word across domains. First, we create two lower dimensional la-

[1] In this paper, we use the term *domain* to refer to a collection of documents about a particular topic, for example reviews of a particular kind of product.

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 613–623,
Baltimore, Maryland, USA, June 23-25 2014.

tent feature spaces separately for the source and the target domains using Singular Value Decomposition (SVD). Second, we learn a mapping from the source domain latent feature space to the target domain latent feature space using Partial Least Square Regression (PLSR). The SVD smoothing in the first step both reduces the data sparseness in distributional representations of individual words, as well as the dimensionality of the feature space, thereby enabling us to efficiently and accurately learn a prediction model using PLSR in the second step. Our proposed cross-domain word distribution prediction method is unsupervised in the sense that it does not require any labeled data in either of the two steps.

Using two popular multi-domain datasets, we evaluate the proposed method in two prediction tasks: (a) predicting the POS of a word in a target domain, and (b) predicting the sentiment of a review in a target domain. Without requiring any task specific customisations, systems based on our distribution prediction method significantly outperform competitive baselines in both tasks. Because our proposed distribution prediction method is unsupervised and task independent, it is potentially useful for a wide range of DA tasks such entity extraction (Guo et al., 2009) or dependency parsing (McClosky et al., 2010). Our contributions are summarised as follows:

- Given the distribution $\boldsymbol{w}_{\mathcal{S}}$ of a word w in a source domain $\mathcal{S}$, we propose a method for learning its distribution $\boldsymbol{w}_{\mathcal{T}}$ in a target domain $\mathcal{T}$.
- Using the learnt distribution prediction model, we propose a method to learn a cross-domain POS tagger.
- Using the learnt distribution prediction model, we propose a method to learn a cross-domain sentiment classifier.

To our knowledge, ours is the first successful attempt to learn a model that predicts the distribution of a word across different domains.

2 Related Work

Learning semantic representations for words using documents from a single domain has received much attention lately (Vincent et al., 2010; Socher et al., 2013; Baroni and Lenci, 2010). As we have already discussed, the semantics of a word varies across different domains, and such variations are not captured by models that only learn a single semantic representation for a word using documents from a single domain.

The POS of a word is influenced both by its context (*contextual bias*), and the domain of the document in which it appears (*lexical bias*). For example, the word *signal* is predominately used as a noun in MEDLINE, whereas it appears predominantly as an adjective in the Wall Street Journal (WSJ) (Blitzer et al., 2006). Consequently, a tagger trained on WSJ would incorrectly tag *signal* in MEDLINE. Blitzer et al. (2006) append the source domain labeled data with predicted pivots (i.e. words that appear in both the source and target domains) to adapt a POS tagger to a target domain. Choi and Palmer (2012) propose a cross-domain POS tagging method by training two separate models: a generalised model and a domain-specific model. At tagging time, a sentence is tagged by the model that is most similar to that sentence. Huang and Yates (2009) train a Conditional Random Field (CRF) tagger with features retrieved from a smoothing model trained using both source and target domain unlabeled data. Adding latent states to the smoothing model further improves the POS tagging accuracy (Huang and Yates, 2012). Schnabel and Schütze (2013) propose a training set filtering method where they eliminate shorter words from the training data based on the intuition that longer words are more likely to be examples of productive linguistic processes than shorter words.

The sentiment of a word can vary from one domain to another. In Structural Correspondence Learning (SCL) (Blitzer et al., 2006; Blitzer et al., 2007), a set of pivots are chosen using pointwise mutual information. Linear predictors are then learnt to predict the occurrence of those pivots, and SVD is used to construct a lower dimensional representation in which a binary classifier is trained. Spectral Feature Alignment (SFA) (Pan et al., 2010) also uses pivots to compute an alignment between domain specific and domain independent features. Spectral clustering is performed on a bipartite graph representing domain specific and domain independent features to find a lower-dimensional projection between the two sets of features. The cross-domain sentiment-sensitive thesaurus (SST) (Bollegala et al., 2011) groups together words that express similar sentiments in

different domains. The created thesaurus is used to expand feature vectors during train and test stages in a binary classifier. However, unlike our method, SCL, SFA, or SST do *not* learn a prediction model between word distributions across domains.

Prior knowledge of the sentiment of words, such as sentiment lexicons, has been incorporated into cross-domain sentiment classification. He et al. (2011) propose a joint sentiment-topic model that imposes a sentiment-prior depending on the occurrence of a word in a sentiment lexicon. Ponomareva and Thelwall (2012) represent source and target domain reviews as nodes in a graph and apply a label propagation algorithm to predict the sentiment labels for target domain reviews from the sentiment labels in source domain reviews. A sentiment lexicon is used to create features for a document. Although incorporation of prior sentiment knowledge is a promising technique to improve accuracy in cross-domain sentiment classification, it is complementary to our task of distribution prediction across domains.

The *unsupervised* DA setting that we consider does not assume the availability of labeled data for the target domain. However, if a small amount of labeled data is available for the target domain, it can be used to further improve the performance of DA tasks (Xiao et al., 2013; Daumé III, 2007).

3 Distribution Prediction

3.1 In-domain Feature Vector Construction

Before we tackle the problem of learning a model to predict the distribution of a word across domains, we must first compute the distribution of a word from a single domain. For this purpose, we represent a word w using unigrams and bigrams that co-occur with w in a sentence as follows.

Given a document H, such as a user-review of a product, we split H into sentences, and lemmatize each word in a sentence using the RASP system (Briscoe et al., 2006). Using a standard stop word list, we filter out frequent non-content unigrams and select the remainder as unigram features to represent a sentence. Next, we generate bigrams of word lemmas and remove any bigrams that consists only of stop words. Bigram features capture negations more accurately than unigrams, and have been found to be useful for sentiment classification tasks. Table 1 shows the unigram and bigram features we extract for a sentence using this procedure. Using data from a single domain, we construct a feature co-occurrence matrix $\mathbf{A}$ in which columns correspond to unigram features and rows correspond to either unigram or bigram features. The value of the element a_{ij} in the co-occurrence matrix $\mathbf{A}$ is set to the number of sentences in which the i-th and j-th features co-occur.

sentence	This is an interesting and well researched book
unigrams (surface)	this, is, an, interesting, and, well, researched, book
unigrams (lemma)	this, be, an, interest, and, well, research, book
unigrams (features)	interest, well, research, book
bigrams (lemma)	this+be, be+an, an+interest, interest+and, and+well, well+research, research+book
bigrams (features)	an+interest, interest+and, and+well, well+research, research+book

Table 1: Extracting unigram and bigram features.

Typically, the number of unique bigrams is much larger than that of unigrams. Moreover, co-occurrences of bigrams are rare compared to co-occurrences of unigrams, and co-occurrences involving a unigram and a bigram. Consequently, in matrix $\mathbf{A}$, we consider co-occurrences only between unigrams vs. unigrams, and bigrams vs. unigrams. We consider each row in $\mathbf{A}$ as representing the distribution of a feature (i.e. unigrams or bigrams) in a particular domain over the unigram features extracted from that domain (represented by the columns of $\mathbf{A}$). We apply Positive Pointwise Mutual Information (PPMI) to the co-occurrence matrix $\mathbf{A}$. This is a variation of the Pointwise Mutual Information (PMI) (Church and Hanks, 1990), in which all PMI values that are less than zero are replaced with zero (Lin, 1998; Bullinaria and Levy, 2007). Let $\mathbf{F}$ be the matrix that results when PPMI is applied to $\mathbf{A}$. Matrix $\mathbf{F}$ has the same number of rows, n_r, and columns, n_c, as the raw co-occurrence matrix $\mathbf{A}$.

Note that in addition to the above-mentioned representation, there are many other ways to represent the distribution of a word in a particular domain (Turney and Pantel, 2010). For example, one can limit the definition of co-occurrence to words that are linked by some dependency relation (Pado and Lapata, 2007), or extend the window of co-occurrence to the entire document (Baroni and Lenci, 2010). Since the method we propose in Section 3.2 to predict the distribution of a word across domains does not depend on the particular

feature representation method, any of these alternative methods could be used.

To reduce the dimensionality of the feature space, and create dense representations for words, we perform SVD on $\mathbf{F}$. We use the left singular vectors corresponding to the k largest singular values to compute a rank k approximation $\hat{\mathbf{F}}$, of $\mathbf{F}$. We perform truncated SVD using SVDLIBC[2]. Each row in $\hat{\mathbf{F}}$ is considered as representing a word in a lower k ($\ll n_c$) dimensional feature space corresponding to a particular domain. Distribution prediction in this lower dimensional feature space is preferrable to prediction over the original feature space because there are reductions in overfitting, feature sparseness, and the learning time. We created two matrices, $\hat{\mathbf{F}}_{\mathcal{S}}$ and $\hat{\mathbf{F}}_{\mathcal{T}}$ from the source and target domains, respectively, using the above mentioned procedure.

3.2 Cross-Domain Feature Vector Prediction

We propose a method to learn a model that can predict the distribution $\boldsymbol{w}_{\mathcal{T}}$ of a word w in the target domain $\mathcal{T}$, given its distribution $\boldsymbol{w}_{\mathcal{S}}$ in the source domain $\mathcal{S}$. We denote the set of features that occur in both domains by $\mathcal{W} = \{w^{(1)}, \ldots, w^{(n)}\}$. In the literature, such features are often referred to as *pivots*, and they have been shown to be useful for DA, allowing the weights learnt to be transferred from one domain to another. Various criteria have been proposed for selecting a small set of pivots for DA, such as the mutual information of a word with the two domains (Blitzer et al., 2007). However, we do not impose any further restrictions on the set of pivots $\mathcal{W}$ other than that they occur in both domains.

For each word $w^{(i)} \in \mathcal{W}$, we denote the corresponding rows in $\hat{\mathbf{F}}_{\mathcal{S}}$ and $\hat{\mathbf{F}}_{\mathcal{T}}$ by column vectors $\boldsymbol{w}_{\mathcal{S}}^{(i)}$ and $\boldsymbol{w}_{\mathcal{T}}^{(i)}$. Note that the dimensionality of $\boldsymbol{w}_{\mathcal{S}}^{(i)}$ and $\boldsymbol{w}_{\mathcal{T}}^{(i)}$ need not be equal, and we may select different numbers of singular vectors to approximate $\hat{\mathbf{F}}_{\mathcal{S}}$ and $\hat{\mathbf{F}}_{\mathcal{T}}$. We model distribution prediction as a multivariate regression problem where, given a set $\{(\boldsymbol{w}_{\mathcal{S}}^{(i)}, \boldsymbol{w}_{\mathcal{T}}^{(i)})\}_{i=1}^{n}$ consisting of pairs of feature vectors selected from each domain for the pivots in $\mathcal{W}$, we learn a mapping from the inputs ($\boldsymbol{w}_{\mathcal{S}}^{(i)}$) to the outputs ($\boldsymbol{w}_{\mathcal{T}}^{(i)}$).

We use Partial Least Squares Regression (PLSR) (Wold, 1985) to learn a regression model using pairs of vectors. PLSR has been applied in Chemometrics (Geladi and Kowalski, 1986), producing stable prediction models even when the number of samples is considerably smaller than the dimensionality of the feature space. In particular, PLSR fits a smaller number of latent variables ($10 - 100$ in practice) such that the correlation between the feature vectors for pivots in the two domains are maximised in this latent space.

Algorithm 1 Learning a prediction model.

Input: $\mathbf{X}$, $\mathbf{Y}$, L.
Output: Prediction matrix $\mathbf{M}$.

1: Randomly select $\boldsymbol{\gamma}_l$ from columns in $\mathbf{Y}_l$.
2: $\boldsymbol{v}_l = {\mathbf{X}_l}^{\top}\boldsymbol{\gamma}_l / \left|\left|{\mathbf{X}_l}^{\top}\boldsymbol{\gamma}_l\right|\right|$
3: $\boldsymbol{\lambda}_l = \mathbf{X}_l\boldsymbol{v}_l$
4: $\boldsymbol{q}_l = {\mathbf{Y}_l}^{\top}\boldsymbol{\lambda}_l / \left|\left|{\mathbf{Y}_l}^{\top}\boldsymbol{\lambda}_l\right|\right|$
5: $\boldsymbol{\gamma}_l = \mathbf{Y}_l\boldsymbol{q}_l$
6: If $\boldsymbol{\gamma}_l$ is unchanged go to Line 7; otherwise go to Line 2
7: $c_l = {\boldsymbol{\lambda}_l}^{\top}\boldsymbol{\gamma}_l / \left|\left|{\boldsymbol{\lambda}_l}^{\top}\boldsymbol{\gamma}_l\right|\right|$
8: $\boldsymbol{p}_l = {\mathbf{X}_l}^{\top}\boldsymbol{\lambda}_l / {\boldsymbol{\lambda}_l}^{\top}\boldsymbol{\lambda}_l$
9: $\mathbf{X}_{l+1} = \mathbf{X}_l - \boldsymbol{\lambda}_l{\boldsymbol{p}_l}^{\top}$ and $\mathbf{Y}_{l+1} = \mathbf{Y}_l - c_l\boldsymbol{\lambda}_l{\boldsymbol{q}_l}^{\top}$.
10: Stop if $l = L$; otherwise $l = l + 1$ and return to Line 1.
11: Let $\mathbf{C} = \mathrm{diag}(c_1, \ldots, c_L)$, and $\mathbf{V} = [\boldsymbol{v}_1 \ldots \boldsymbol{v}_L]$
12: $\mathbf{M} = \mathbf{V}(\mathbf{P}^{\top}\mathbf{V})^{-1}\mathbf{C}\mathbf{Q}^{\top}$
13: **return** $\mathbf{M}$

Let $\mathbf{X}$ and $\mathbf{Y}$ denote matrices formed by arranging respectively the vectors $\boldsymbol{w}_{\mathcal{S}}^{(i)}$s and $\boldsymbol{w}_{\mathcal{T}}^{(i)}$ in rows. PLSR decomposes $\mathbf{X}$ and $\mathbf{Y}$ into a series of products between rank 1 matrices as follows:

$$\mathbf{X} \approx \sum_{l=1}^{L} \boldsymbol{\lambda}_l{\boldsymbol{p}_l}^{\top} = \boldsymbol{\Lambda}\mathbf{P}^{\top} \quad (1)$$

$$\mathbf{Y} \approx \sum_{l=1}^{L} \boldsymbol{\gamma}_l{\boldsymbol{q}_l}^{\top} = \boldsymbol{\Gamma}\mathbf{Q}^{\top}. \quad (2)$$

Here, $\boldsymbol{\lambda}_l$, $\boldsymbol{\gamma}_l$, $\boldsymbol{p}_l$, and $\boldsymbol{q}_l$ are column vectors, and the summation is taken over the rank 1 matrices that result from the outer product of those vectors. The matrices, $\boldsymbol{\Lambda}$, $\boldsymbol{\Gamma}$, $\mathbf{P}$, and $\mathbf{Q}$ are constructed respectively by arranging $\boldsymbol{\lambda}_l$, $\boldsymbol{\gamma}_l$, $\boldsymbol{p}_l$, and $\boldsymbol{q}_l$ vectors as columns.

Our method for learning a distribution prediction model is shown in Algorithm 1. It is based on the two block NIPALS routine (Wold, 1975; Rosipal and Kramer, 2006) and iteratively discovers L pairs of vectors $(\boldsymbol{\lambda}_l, \boldsymbol{\gamma}_l)$ such that the covariances, $\mathrm{Cov}(\boldsymbol{\lambda}_l, \boldsymbol{\gamma}_l)$, are maximised under the constraint $||\boldsymbol{p}_l|| = ||\boldsymbol{q}_l|| = 1$. Finally, the prediction matrix, $\mathbf{M}$ is computed using $\boldsymbol{\lambda}_l, \boldsymbol{\gamma}_l, \boldsymbol{p}_l, \boldsymbol{q}_l$. The predicted distribution $\hat{\boldsymbol{w}}_{\mathcal{T}}$ of a word w in $\mathcal{T}$ is given by

$$\hat{\boldsymbol{w}}_{\mathcal{T}} = \mathbf{M}\boldsymbol{w}_{\mathcal{S}}. \quad (3)$$

[2] http://tedlab.mit.edu/~dr/SVDLIBC/

Our distribution prediction learning method is unsupervised in the sense that it does not require manually labeled data for a particular task from any of the domains. This is an important point, and means that the distribution prediction method is independent of the task to which it may subsequently be applied. As we go on to show in Section 6, this enables us to use the same distribution prediction method for both POS tagging and sentiment classification.

4 Domain Adaptation

The main reason that a model trained only on the source domain labeled data performs poorly in the target domain is the *feature mismatch* – few features in target domain test instances appear in source domain training instances. To overcome this problem, we use the proposed distribution prediction method to find those related features in the source domain that correspond to the features appearing in the target domain test instances.

We consider two DA tasks: (a) cross-domain POS tagging (Section 4.1), and (b) cross-domain sentiment classification (Section 4.2). Note that our proposed distribution prediction method can be applied to numerous other NLP tasks that involve sequence labelling and document classification.

4.1 Cross-Domain POS Tagging

We represent each word using a set of features such as capitalisation (whether the first letter of the word is capitalised), numeric (whether the word contains digits), prefixes up to four letters, and suffixes up to four letters (Miller et al., 2011). Next, for each word w in a source domain labeled (i.e. manually POS tagged) sentence, we select its neighbours $u^{(i)}$ in the source domain as additional features. Specifically, we measure the similarity, $\mathrm{sim}(\boldsymbol{u}_{\mathcal{S}}^{(i)}, \boldsymbol{w}_{\mathcal{S}})$, between the source domain distributions of $u^{(i)}$ and w, and select the top r similar neighbours $u^{(i)}$ for each word w as additional features for w. We refer to such features as *distributional features* in this work. The value of a neighbour $u^{(i)}$ selected as a distributional feature is set to its similarity score $\mathrm{sim}(\boldsymbol{u}_{\mathcal{S}}^{(i)}, \boldsymbol{w}_{\mathcal{S}})$. Next, we train a CRF model using all features (i.e. capitalisation, numeric, prefixes, suffixes, and distributional features) on source domain labeled sentences.

We train a PLSR model, $\mathbf{M}$, that predicts the target domain distribution $\mathbf{M}\boldsymbol{u}_{\mathcal{S}}^{(i)}$ of a word $u^{(i)}$ in the source domain labeled sentences, given its distribution, $\boldsymbol{u}_{\mathcal{S}}^{(i)}$. At test time, for each word w that appears in a target domain test sentence, we measure the similarity, $\mathrm{sim}(\mathbf{M}\boldsymbol{u}_{\mathcal{S}}^{(i)}, \boldsymbol{w}_{\mathcal{T}})$, and select the most similar r words $u^{(i)}$ in the source domain labeled sentences as the distributional features for w, with their values set to $\mathrm{sim}(\mathbf{M}\boldsymbol{u}_{\mathcal{S}}^{(i)}, \boldsymbol{w}_{\mathcal{T}})$. Finally, the trained CRF model is applied to a target domain test sentence.

Note that distributional features are always selected from the source domain during both train and test times, thereby increasing the number of overlapping features between the trained model and test sentences. To make the inference tractable and efficient, we use a first-order Markov factorisation, in which we consider all pairwise combinations between the features for the current word and its immediate predecessor.

4.2 Cross-Domain Sentiment Classification

Unlike in POS tagging, where we must individually tag each word in a target domain test sentence, in sentiment classification we must classify the sentiment for the entire review. We modify the DA method presented in Section 4.1 to satisfy this requirement as follows.

Let us assume that we are given a set $\{(\boldsymbol{x}_{\mathcal{S}}^{(i)}, y^{(i)})\}_{i=1}^{n}$ of n labeled reviews $\boldsymbol{x}_{\mathcal{S}}^{(i)}$ for the source domain $\mathcal{S}$. For simplicity, let us consider binary sentiment classification where each review $\boldsymbol{x}^{(i)}$ is labeled either as positive (i.e. $y^{(i)} = 1$) or negative (i.e. $y^{(i)} = -1$). Our cross-domain binary sentiment classification method can be easily extended to the multi-class setting as well. First, we lemmatise each word in a source domain labeled review $\boldsymbol{x}_{\mathcal{S}}^{(i)}$, and extract both unigrams and bigrams as features to represent $\boldsymbol{x}_{\mathcal{S}}^{(i)}$ by a binary-valued feature vector. Next, we train a binary classification model, $\boldsymbol{\theta}$, using those feature vectors. Any binary classification algorithm can be used to learn $\boldsymbol{\theta}$. In our experiments, we used L2 regularised logistic regression.

Next, we train a PLSR model, $\mathbf{M}$, as described in Section 3.2 using unlabeled reviews in the source and target domains. At test time, we represent a test target review H using a binary-valued feature vector $\boldsymbol{h}$ of unigrams and bigrams of lemmas of the words in H, as we did for source domain labeled train reviews. Next, for each feature $w^{(j)}$ extracted from H, we measure the similarity,

$\text{sim}(\mathbf{M}\boldsymbol{u}_{\mathcal{S}}^{(i)}, \boldsymbol{w}_{\mathcal{T}}^{(j)})$, between the target domain distribution of $w^{(j)}$, and each feature (unigram or bigram) $u^{(i)}$ in the source domain labeled reviews. We score each source domain feature $u^{(i)}$ for its relatedness to H using the formula:

$$\text{score}(u^{(i)}, \mathsf{H}) = \frac{1}{|\mathsf{H}|} \sum_{j=1}^{|\mathsf{H}|} \text{sim}(\mathbf{M}\boldsymbol{u}_{\mathcal{S}}^{(i)}, \boldsymbol{w}_{\mathcal{T}}^{(j)}) \quad (4)$$

where $|\mathsf{H}|$ denotes the total number of features extracted from the test review H. We select the top scoring r features $u^{(i)}$ as distributional features for H, and append those to $\boldsymbol{h}$. The corresponding values of those distributional features are set to the scores given by Equation 4. Finally, we classify $\boldsymbol{h}$ using the trained binary classifier $\boldsymbol{\theta}$. Note that given a test review, we find the distributional features that are similar to *all* the words in the test review from the source domain. In particular, we *do not* find distributional features independently for each word in the test review. This enables us to find distributional features that are consistent with all the features in a test review.

4.3 Model Choices

For both POS tagging and sentiment classification, we experimented with several alternative approaches for feature weighting, representation, and similarity measures using development data, which we randomly selected from the training instances from the datasets described in Section 5.

For feature weighting for sentiment classification, we considered using the number of occurrences of a feature in a review and tf-idf weighting (Salton and Buckley, 1983). For representation, we considered distributional features $u^{(i)}$ in descending order of their scores given by Equation 4, and then taking the inverse-rank as the values for the distributional features (Bollegala et al., 2011). However, none of these alternatives resulted in performance gains. With respect to similarity measures, we experimented with cosine similarity and the similarity measure proposed by Lin (1998); cosine similarity performed consistently well over all the experimental settings. The feature representation was held fixed during these similarity measure comparisons.

For POS tagging, we measured the effect of varying r, the number of distributional features, using a development dataset. We observed that setting r larger than 10 did not result in significant improvements in tagging accuracy, but only increased the train time due to the larger feature space. Consequently, we set $r = 10$ in POS tagging. For sentiment analysis, we used all features in the source domain labeled reviews as distributional features, weighted by their scores given by Equation 4, taking the inverse-rank. In both tasks, we parallelised similarity computations using BLAS[3] level-3 routines to speed up the computations. The source code of our implementation is publicly available[4].

5 Datasets

To evaluate DA for POS tagging, following Blitzer et al. (2006), we use sections $2 - 21$ from Wall Street Journal (WSJ) as the source domain labeled data. An additional $100,000$ WSJ sentences from the 1988 release of the WSJ corpus are used as the source domain unlabeled data. Following Schnabel and Schütze (2013), we use the POS labeled sentences in the SACNL dataset (Petrov and McDonald, 2012) for the five target domains: QA forums, Emails, Newsgroups, Reviews, and Blogs. Each target domain contains around 1000 POS labeled test sentences and around $100,000$ unlabeled sentences.

To evaluate DA for sentiment classification, we use the Amazon product reviews collected by Blitzer et al. (2007) for four different product categories: books (**B**), DVDs (**D**), electronic items (**E**), and kitchen appliances (**K**). There are 1000 positive and 1000 negative sentiment labeled reviews for each domain. Moreover, each domain has on average $17,547$ unlabeled reviews. We use the standard split of 800 positive and 800 negative labeled reviews from each domain as training data, and the remainder for testing.

6 Experiments and Results

For each domain $\mathcal{D}$ in the SANCL (POS tagging) and Amazon review (sentiment classification) datasets, we create a PPMI weighted co-occurrence matrix $\mathbf{F}_{\mathcal{D}}$. On average, $\mathbf{F}_{\mathcal{D}}$ created for a target domain in the SANCL dataset contains $104,598$ rows and $65,528$ columns, whereas those numbers in the Amazon dataset are $27,397$ and $35,200$ respectively. In cross-domain sentiment classification, we measure the binary sentiment classification accuracy for the target domain

[3] http://www.openblas.net/

[4] http://www.csc.liv.ac.uk/~danushka/software.html

test reviews for each pair of domains (12 pairs in total for 4 domains). On average, we have $40,176$ pivots for a pair of domains in the Amazon dataset.

In cross-domain POS tagging, WSJ is always the source domain, whereas the five domains in SANCL dataset are considered as the target domains. For this setting we have 9822 pivots on average. The number of singular vectors k selected in SVD, and the number of PLSR dimensions L are set respectively to 1000 and 50 for the remainder of the experiments described in the paper. Later we study the effect of those two parameters on the performance of the proposed method. The L-BFGS (Liu and Nocedal, 1989) method is used to train the CRF and logistic regression models.

6.1 POS Tagging Results

Table 2 shows the token-level POS tagging accuracy for *unseen* words (i.e. words that appear in the target domain test sentences but not in the source domain labeled train sentences). By limiting the evaluation to unseen words instead of all words, we can evaluate the gain in POS tagging accuracy solely due to DA. The **NA** (no-adapt) baseline simulates the effect of not performing any DA. Specifically, in POS tagging, a CRF trained on source domain labeled sentences is applied to target domain test sentences, whereas in sentiment classification, a logistic regression classifier trained using source domain labeled reviews is applied to the target domain test reviews. The $\mathcal{S}_{\mathbf{pred}}$ baseline directly uses the source domain distributions for the words instead of projecting them to the target domain. This is equivalent to setting the prediction matrix $\mathbf{M}$ to the unit matrix. The $\mathcal{T}_{\mathbf{pred}}$ baseline uses the target domain distribution $\boldsymbol{w}_{\mathcal{T}}$ for a word w instead of $\mathbf{M}\boldsymbol{w}_{\mathcal{S}}$. If w does not appear in the target domain, then $\boldsymbol{w}_{\mathcal{T}}$ is set to the zero vector. The $\mathcal{S}_{pred}$ and $\mathcal{T}_{pred}$ baselines simulate the two alternatives of using source and target domain distributions instead of learning a PLSR model. The DA method proposed in Section 4.1 is shown as the **Proposed** method. **Filter** denotes the training set filtering method proposed by Schnabel and Schütze (2013) for the DA of POS taggers.

From Table 2, we see that the **Proposed** method achieves the best performance in all five domains, followed by the $\mathcal{T}_{pred}$ baseline. Recall that the $\mathcal{T}_{pred}$ baseline cannot find source domain words that do not appear in the target domain as distributional features for the words in the target domain test reviews. Therefore, when the overlap between the vocabularies used in the source and the target domains is small, $\mathcal{T}_{pred}$ cannot reduce the mismatch between the feature spaces. Poor performance of the $\mathcal{S}_{pred}$ baseline shows that the distributions of a word in the source and target domains are different to the extent that the distributional features found using source domain distributions are inadequate. The two baselines $\mathcal{S}_{pred}$ and $\mathcal{T}_{pred}$ collectively motivate our proposal to learn a distribution prediction model from the source domain to the target. The improvements of **Proposed** over the previously proposed **Filter** are statistically significant in all domains except the Emails domain (denoted by † in Table 2 according to the Binomial exact test at 95% confidence). However, the differences between the $\mathcal{T}_{pred}$ and **Proposed** methods are not statistically significant.

Target	NA	$\mathcal{S}_{pred}$	$\mathcal{T}_{pred}$	Filter	Proposed
QA	67.34	68.18	68.75	57.08	$\mathbf{69.28}^{\dagger}$
Emails	65.62	66.62	67.07	65.61	**67.09**
Newsgroups	75.71	75.09	75.57	70.37	$\mathbf{75.85}^{\dagger}$
Reviews	56.36	54.60	56.68	47.91	$\mathbf{56.93}^{\dagger}$
Blogs	76.64	54.78	76.90	74.56	$\mathbf{76.97}^{\dagger}$

Table 2: POS tagging accuracies on SANCL.

6.2 Sentiment Classification Results

In Figure 1, we compare the **Proposed** cross-domain sentiment classification method (Section 4.2) against several baselines and the current state-of-the-art methods. The baselines **NA**, $\mathcal{S}_{\mathbf{pred}}$, and $\mathcal{T}_{\mathbf{pred}}$ are defined similarly as in Section 6.1. **SST** is the Sentiment Sensitive Thesaurus proposed by Bollegala et al. (2011). **SST** creates a single distribution for a word using both source and target domain reviews, instead of two separate distributions as done by the **Proposed** method. **SCL** denotes the Structural Correspondence Learning method proposed by Blitzer et al. (2006). **SFA** denotes the Spectral Feature Alignment method proposed by Pan et al. (2010). **SFA** and **SCL** represent the current state-of-the-art methods for cross-domain sentiment classification. All methods are evaluated under the same settings, including train/test split, feature spaces, pivots, and classification algorithms so that any differences in performance can be directly attributable to their domain adaptability. For each domain, the accuracy obtained by a classifier trained using labeled data from that

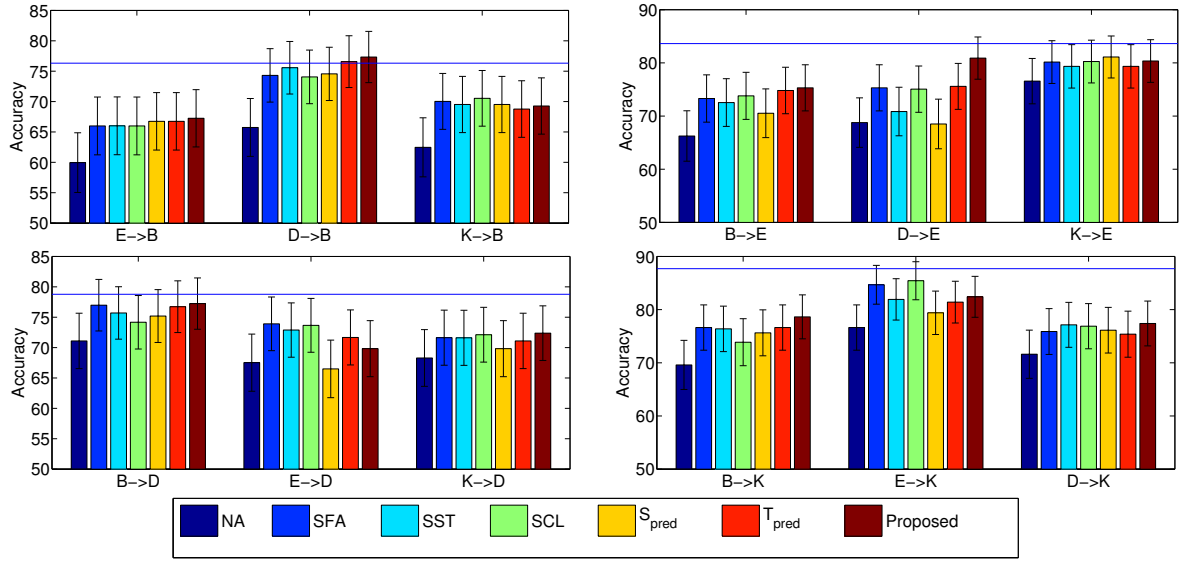

Figure 1: Cross-Domain sentiment classification.

domain is indicated by a solid horizontal line in each sub-figure. This upper baseline represents the classification accuracy we could hope to obtain if we were to have labeled data for the target domain. Clopper-Pearson 95% binomial confidence intervals are superimposed on each vertical bar.

From Figure 1 we see that the **Proposed** method reports the best results in 8 out of the 12 domain pairs, whereas **SCL**, **SFA**, and $\mathcal{S}_{\mathbf{pred}}$ report the best results in other cases. Except for the **D-E** setting in which **Proposed** method significantly outperforms both **SFA** and **SCL**, the performance of the **Proposed** method is not statistically significantly different to that of **SFA** or **SCL**.

The selection of pivots is vital to the performance of **SFA**. However, unlike **SFA**, which requires us to carefully select a small subset of pivots (ca. less than 500) using some heuristic approach, our **Proposed** method does not require any pivot selection. Moreover, **SFA** projects source domain reviews to a lower-dimensional latent space, in which a binary sentiment classifier is subsequently trained. At test time **SFA** projects a target review into this lower-dimensional latent space and applies the trained classifier. In contrast, our **Proposed** method predicts the distribution of a word in the target domain, given its distribution in the source domain, thereby explicitly *translating* the source domain reviews to the target. This property enables us to apply the proposed distribution prediction method to tasks other than sentiment analysis such as POS tagging where we must identify distributional features for individual words.

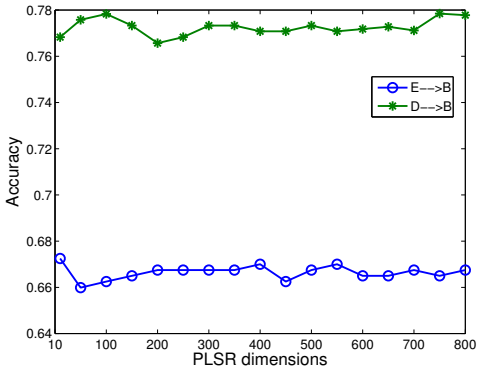

Figure 2: The effect of PLSR dimensions.

Unlike our distribution prediction method, which is unsupervised, **SST** requires labeled data for the source domain to learn a feature mapping between a source and a target domain in the form of a thesaurus. However, from Figure 1 we see that in 10 out of the 12 domain-pairs the **Proposed** method returns higher accuracies than **SST**.

To evaluate the overall effect of the number of singular vectors k used in the SVD step, and the number of PLSR components L used in Algorithm 1, we conduct two experiments. To evaluate the effect of the PLSR dimensions, we fixed $k = 1000$ and measured the cross-domain sentiment classification accuracy over a range of L values. As shown in Figure 2, accuracy remains stable across a wide range of PLSR dimensions. Because the time complexity of Algorithm 1 increases linearly with L, it is desirable that we select smaller L val-

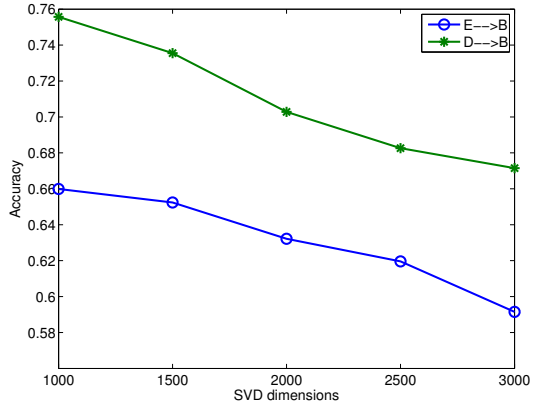

Figure 3: The effect of SVD dimensions.

Measure	Distributional features
$\mathrm{sim}(u_{\mathcal{S}}, w_{\mathcal{S}})$	thin (0.1733), digestible (0.1728), small+print (0.1722)
$\mathrm{sim}(u_{\mathcal{T}}, w_{\mathcal{T}})$	travel+companion (0.6018), snap-in (0.6010), touchpad (0.6016)
$\mathrm{sim}(u_{\mathcal{S}}, w_{\mathcal{T}})$	segregation (0.1538), participation (0.1512), depression+era (0.1508)
$\mathrm{sim}(\mathbf{M}u_{\mathcal{S}}, w_{\mathcal{T}})$	small (0.2794), compact (0.2641), sturdy (0.2561)

Table 3: Top 3 distributional features $u \in \mathcal{S}$ for the word *lightweight* (w).

ues in practice.

To evaluate the effect of the SVD dimensions, we fixed $L = 100$ and measured the cross-domain sentiment classification accuracy for different k values as shown in Figure 3. We see an overall decrease in classification accuracy when k is increased. Because the dimensionality of the source and target domain feature spaces is equal to k, the complexity of the least square regression problem increases with k. Therefore, larger k values result in overfitting to the train data and classification accuracy is reduced on the target test data.

As an example of the distribution prediction method, in Table 3 we show the top 3 similar distributional features u in the *books* (source) domain, predicted for the *electronics* (target) domain word w = *lightweight*, by different similarity measures. Bigrams are indicted by a + sign and the similarity scores of the distributional features are shown within brackets.

Using the source domain distributions for both u and w (i.e. $\mathrm{sim}(u_{\mathcal{S}}, w_{\mathcal{S}})$) produces distributional features that are specific to the books domain, or to the dominant adjectival sense of *having no importance or influence*. On the other hand, using target domain distributions for u and w (i.e. $\mathrm{sim}(u_{\mathcal{T}}, w_{\mathcal{T}})$) returns distributional features of the dominant nominal sense of *lower in weight* frequently associated with electronic devices. Simply using source domain distributions $u_{\mathcal{S}}$ (i.e. $\mathrm{sim}(u_{\mathcal{S}}, w_{\mathcal{T}})$) returns totally unrelated distributional features. This shows that word distributions in source and target domains are very different and some adaptation is required prior to computing distributional features.

Interestingly, we see that by using the distributions predicted by the proposed method (i.e. $\mathrm{sim}(\mathbf{M}u_{\mathcal{S}}, w_{\mathcal{T}})$) we overcome this problem and find relevant distributional features from the source domain. Although for illustrative purposes we used the word *lightweight*, which occurs in both the source and the target domains, our proposed method does not require the source domain distribution $w_{\mathcal{S}}$ for a word w in a target domain document. Therefore, it can find distributional features even for words occurring only in the target domain, thereby reducing the feature mismatch between the two domains.

7 Conclusion

We proposed a method to predict the distribution of a word across domains. We first create a distributional representation for a word using the data from a single domain, and then learn a Partial Least Square Regression (PLSR) model to predict the distribution of a word in a target domain given its distribution in a source domain. We evaluated the proposed method in two domain adaptation tasks: cross-domain POS tagging and cross-domain sentiment classification. Our experiments show that without requiring any task-specific customisations to our distribution prediction method, it outperforms competitive baselines and achieves comparable results to the current state-of-the-art domain adaptation methods.

References

Anthony Aue and Michael Gamon. 2005. Customizing sentiment classifiers to new domains: a case study. Technical report, Microsoft Research.

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673 – 721.

Romaric Besançon, Martin Rajman, and Jean-Cédric Chappelier. 1999. Textual similarities based on a

distributional approach. In *Proc. of DEXA*, pages 180 – 184.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proc. of EMNLP*, pages 120 – 128.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proc. of ACL*, pages 440 – 447.

Danushka Bollegala, David Weir, and John Carroll. 2011. Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification. In *Proc. of ACL/HLT*, pages 132 – 141.

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proc. of COLING/ACL Interactive Presentation Sessions*.

John A. Bullinaria and Jospeh P. Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510 – 526.

Jinho D. Choi and Martha Palmer. 2012. Fast and robust part-of-speech tagging using dynamic model selection. In *Proc. of ACL Short Papers*, volume 2, pages 363 – 367.

Kenneth W. Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22 – 29, March.

James Curran. 2005. Supersense tagging of unknown nouns using semantic similarity. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 26 – 33.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proc. of ACL*, pages 256 – 263.

John R. Firth. 1957. A synopsis of linguistic theory 1930-55. *Studies in Linguistic Analysis*, pages 1 – 32.

Paul Geladi and Bruce R. Kowalski. 1986. Partial least-squares regression: a tutorial. *Analytica Chimica Acta*, 185(0):1 – 17.

Honglei Guo, Huijia Zhu, Zhili Guo, Xiaoxun Zhang, Xian Wu, and Zhong Su. 2009. Domain adaptation with latent semantic association for named entity recognition. In *Proc. of NAACL*, pages 281 – 289.

Yulan He, Chenghua Lin, and Harith Alani. 2011. Automatically extracting polarity-bearing topics for cross-domain sentiment classification. In *Proc. of ACL/HLT*, pages 123 – 131.

Fei Huang and Alexander Yates. 2009. Distributional representations for handling sparsity in supervised sequence-labeling. In *ACL-IJCNLP'09*, pages 495 – 503.

Fei Huang and Alexander Yates. 2012. Biased representation learning for domain adaptation. In *Proc. of EMNLP/CoNLL*, pages 1313 – 1323.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2012. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(4):359 – 389.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proc. of ACL*, pages 768 – 774.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503 – 528.

David McClosky, Eugene Charniak, and Mark Johnson. 2010. Automatic domain adaptation for parsing. In *Proc. of NAACL/HLT*, pages 28 – 36.

John E. Miller, Manabu Torii, and K. Vijay-Shanker. 2011. Building domain-specific taggers without annotated (domain) data. In *Proc. of EMNLP/CoNLL*, pages 1103 – 1111.

Sebastian Pado and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161 – 199.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proc. of WWW*, pages 751 – 760.

Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *Proc. of EMNLP*, pages 938 – 947.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Notes of the 1st SANCL Workshop*.

Natalia Ponomareva and Mike Thelwall. 2012. Do neighbours help? an exploration of graph-based algorithms for cross-domain sentiment classification. In *Proc. of EMNLP*, pages 655 – 665.

Roman Rosipal and Nicole Kramer. 2006. Overview and recent advances in partial least squares. In C. Saunders et al., editor, *SLSFS'05*, volume 3940 of *LNCS*, pages 34 – 51, Berlin Heidelberg. Springer-Verlag.

G. Salton and C. Buckley. 1983. *Introduction to Modern Information Retreival*. McGraw-Hill Book Company.

Tobias Schnabel and Hinrich Schütze. 2013. Towards robust cross-domain domain adaptation for part-of-speech tagging. In *Proc. of IJCNLP*, pages 198 – 206.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*, pages 1631 – 1642.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Aritificial Intelligence Research*, 37:141 – 188.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antonie Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371 – 3408.

Herman Wold. 1975. Path models with latent variables: the NIPALS approach. In H. M. Blalock et al., editor, *Quantitative socialogy: international perspective on mathematical and statistical modeling*, pages 307 – 357. Academic.

Herman Wold. 1985. Partial least squares. In Samel Kotz and Norman L. Johnson, editors, *Encyclopedia of the Statistical Sciences*, pages 581 – 591. Wiley.

Min Xiao, Feipeng Zhao, and Yuhong Guo. 2013. Learning latent word representations for domain adaptation using supervised word clustering. In *Proc. of EMNLP*, pages 152 – 162.