Cross-domain Text Classification through Iterative Refining of Target Categories Representations

Giacomo Domeniconi[1], Gianluca Moro[1], Roberto Pasolini[1] and Claudio Sartori[2]

[1]DISI, Università degli Studi di Bologna, Via Venezia 52, Cesena, Italy

[2]DISI, Università degli Studi di Bologna, Viale del Risorgimento 2, Bologna, Italy

Keywords: Text Mining, Text Classification, Transfer Learning, Cross-domain Classification.

Abstract: Cross-domain text classification deals with predicting topic labels for documents in a target domain by leveraging knowledge from pre-labeled documents in a source domain, with different terms or different distributions thereof. Methods exist to address this problem by re-weighting documents from the source domain to transfer them to the target one or by finding a common feature space for documents of both domains; they often require the combination of complex techniques, leading to a number of parameters which must be tuned for each dataset to yield optimal performances. We present a simpler method based on creating explicit representations of topic categories, which can be compared for similarity to the ones of documents. Categories representations are initially built from relevant source documents, then are iteratively refined by considering the most similar target documents, with relatedness being measured by a simple regression model based on cosine similarity, built once at the begin. This expectedly leads to obtain accurate representations for categories in the target domain, used to classify documents therein. Experiments on common benchmark text collections show that this approach obtains results better or comparable to other methods, obtained with fixed empirical values for its few parameters.

1 INTRODUCTION

Text classification (or *categorization*) generally entails the automatic organization of text documents into a user-defined taxonomy of *classes* or *categories*, which typically correspond to topics discussed in the documents, such as *science*, *arts*, *history* and so on. This general task is useful to organize many types of documents like news stories, books and mail messages and may be applied within several contexts, including spam filtering, sentiment analysis, ad-hoc advertising, etc.

Classic text classification methods require a *training set* of documents, which must be already labeled with correct classes, to infer a knowledge model which is then able to classify further unseen documents under the same classes: this general approach is used by many different works, shown to be highly effective in organizing documents among several classes (Sebastiani, 2002).

However, a usable training set, other than being reasonably sized, should reflect quite precisely the characteristics of the documents to be classified: this generally assumes having documents classified under the very same categories of interest and basically containing equal or correlated terms. In other words, denoting a set of classes to be recognized together with the distribution of terms across them as a *domain*, we would need a training set of labeled documents falling within the very same domain. Such a training set, in some real contexts, may be unavailable or may require unfeasible human efforts or costs.

In some cases, although, we have at our disposal a set of labeled documents in a domain which is only slightly different from the one where we want to classify documents. For example, we may have a set of labeled documents with topics which are similar to those of interest, such that each topic on one side may be mapped to one on the other. On the other hand, we may have the same topics, but treated with some different terms, as may happen if we want to leverage a training set of outdated documents to classify newer ones. On a theoretical point of view, we usually have the same class labels equally conditioned by the input data, but the data itself has different distributions.

At this extent, methods for *cross-domain text clas-*

This work was partially supported by the project GenData, funded by the Italian MIUR.

Domeniconi G., Moro G., Pasolini R. and Sartori C..
Cross-domain Text Classification through Iterative Refining of Target Categories Representations.
DOI: 10.5220/0005069400310042
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (KDIR-2014), pages 31-42
ISBN: 978-989-758-048-2

sification exist, which can be used to classify unlabeled documents of a *target* domain by exploiting the knowledge obtained from labeled documents of a *source* domain. These methods fall into the wider class of *transfer learning* approaches, as they generally involve the transfer of knowledge obtained from the source domain to the target (Pan and Yang, 2010).

Different approaches exist for this task: some methods are generally based on adapting source data to the target domain, while others rely on bringing data of both domains in a common feature space to spot similarities. These methods are usually based on advanced statistical concepts and techniques, generally making their exact implementation difficult. Moreover, the outcome of these methods is often heavily influenced by respective parameters: while for each possible dataset there are ranges of parameters values yielding optimal accuracy, these values are generally different for each dataset, thus requiring to discover a combination of parameters' values that produces acceptable results, following generally a poor and unpredictable trial-and-error approach in a search space whose largeness grows exponentially with respect to the number of parameters.

In other words, solutions that need a high number of parameters often achieve good results in controlled environments with known test sets, thanks to repeated try–and–error cycles for parameter tuning, but in the real world sometimes they are not robust enough.

To alleviate the problem of parameter settings, we present in this work a simple novel method for cross-domain text classification based on building and iteratively improving structured representations for the categories in the target domain. In practice, the method starts from typical *bag-of-words* representations for single documents from source and target domains and combines those from the source domain to build preliminary representations for the top-level categories shared between the two domains; these are then refined by iteratively making them "closer" to documents of the target domain, to finally obtain fairly accurate representations of the corresponding categories. This works by comparing these representations of documents and categories by means of an univariate logistic regression model, built once before the iterative phase and based on the standard cosine similarity measure: this is used to pick documents which are most similar to each category, from which new representations are built each time, until they become as consistent as possible with the target domain.

We performed experiments on text collections commonly used as benchmarks, showing that this approach can achieve the same performances of the best known methods with good efficiency, despite a simple and compact implementation. We also show that these results are obtained by always using the same values for the two parameters: this eliminates the need of tuning, thus making the method more practically usable in real scenarios.

The rest of the paper is organized as follows. Section 2 reports an overview of other works about cross-domain text classification. Section 3 exposes in detail the method used to classify documents. Section 4 describes the experiments performed and reports their results, compared with those of other works. Finally, Section 5 sums up the the contribute and discusses possible future developments.

2 RELATED WORK

Supervised machine learning-based methods for text classification are largely diffused and have proven to be fairly effective in classifying documents across large taxonomies of categories, either flat or hierarchical, provided that suitable training sets of prelabeled documents are given (Dumais et al., 1998; Joachims, 1998; Yang and Liu, 1999; Sebastiani, 2002). Unsupervised approaches also exist, which are able to some extent to isolate previously unknown groups (*clusters*) of correlated documents, but generally cannot reach the accuracy of supervised approaches (Merkl, 1998; Kohonen et al., 2000).

A common approach is to represent documents as vectors of numeric features, computed according to their content. Words are often used as features, with each document represented by the number of occurrencies of each or by some derived measure: this is known as the *bag of words* approach (Sebastiani, 2002). Some later methods make use of statistical techniques like Latent Semantic Indexing (Weigend et al., 1999) or Latent Dirichlet Allocation (Blei et al., 2003) to discover hidden correlations between words and consequently improve representations of documents. More recent methods extract semantic information carried by terms by leveraging external knowledge bases such as the WordNet database (Scott and Matwin, 1998) or Wikipedia (Gabrilovich and Markovitch, 2007).

While in most text classification methods standard machine learning algorithms are used on bags of words, a somewhat distinct approach is the *Rocchio method*, where bags obtained from training documents are averaged to build similar representations for categories and each new document is assigned to the category having the representation which is most similar to it (Joachims, 1997): our method is similarly based on the idea of explicitely representing categories as averages of relevant documents.

Text categorization is one of the most relevant applications for cross-domain classification, also referred to as *domain adaptation*. According to the scheme proposed in (Pan and Yang, 2010), cross-domain classification is a case of *transductive transfer learning*, where knowledge must be transferred across two domains which are generally different while having the same labels $\mathcal{Y}$ for data. In many cases, including text classification, the two domains share (or are trivially represented in) a common feature space $\mathcal{X}$.

It is also often assumed that labels in source and target domains are equally conditioned by the input data, which though is distributed differently between the two; denoting with X_S and Y_S data and labels for the source domain and with X_T and Y_T those for the target domain, we have $P(Y_S|X_S) = P(Y_T|X_S)$, but $P(X_S) \neq P(X_T)$: this condition is known as *covariate shift* (Shimodaira, 2000).

Often, two major approaches to transductive transfer learning are distinguished: (Pan and Yang, 2010) and other works refer to them as *instance-transfer* and *feature-representation-transfer*.

Instance-transfer-based approaches generally work by re-weighting instances (data samples) from the source domain to adapt them to the target domain, in order to compensate the discrepancy between $P(X_S)$ and $P(X_T)$: this generally involves estimating an *importance* $\frac{P(x_S)}{P(x_T)}$ for each source instance x_S to reuse it as a training instance x_T under the target domain.

Some works mainly address the related problem of sample selection bias, where a classifier must be learned from a training set with a biased data distribution. (Zadrozny, 2004) analyzes the bias impact on various learning methods and proposes a correction method using knowledge of selection probabilities.

The *kernel mean matching* method (Huang et al., 2007) learns re-weighting factors by matching the means between the domains data in a reproducing kernel Hilbert space (RKHS); this is done without estimating $P(X_S)$ and $P(X_T)$ from a possibly limited quantity of samples. Among other works operating under this restriction there is the *Kullback-Liebler importance estimation procedure* (Sugiyama et al., 2007), a model to estimate importance based on minimization of the Kullback-Liebler divergence between real and expected $P(X_T)$.

Among works specifically considering text classification, (Dai et al., 2007b) trains a Naïve Bayes classifier on the source domain and transfers it to the target domain through an iterative Expectation-Maximization algorithm. In (Gao et al., 2008) multiple classifiers are trained on possibly multiple source domains and combined in a *locally weighted ensemble* based on similarity to a clustering of the target documents to classify them.

On the other side, feature-representation-transfer-based approaches generally work by finding a new feature space to represent instances of both source and target domains, where their differences are reduced and standard learning methods can be applied.

The *structural correspondence learning* method (Blitzer et al., 2006) works by introducing *pivot* features and learning linear predictors for them, whose resulting weights are transformed through Singular Value Decomposition and then used to augment training data instances. The paper (Daumé III, 2007) presents a simple method based on augmenting instances with features differentiating source and target domains, possibly improvable through nonlinear kernel mapping. In (Ling et al., 2008a) a spectral classification-based framework is introduced, using an objective function which balances the source domain supervision and the target domain structure. With the *Maximum Mean Discrepancy (MMD) Embedding* method (Pan et al., 2008), source and target instances are brought to a common low-dimensional space where differences between data distributions are reduced; *transfer component analysis* (Pan et al., 2011) improves this approach in terms of efficiency and generalization to unseen target data.

The following works are focused on text classification. In (Dai et al., 2007a) an approach based on co-clustering of words and documents is used, where labels are transferred across domain using word clusters as a bridge. The *topic-bridged PLSA* method (Xue et al., 2008) is instead based on Probabilistic Latent Semantic Analysis, which is extended to accept unlabeled data. In (Zhuang et al., 2011) is proposed a framework for joint non-negative matrix tri-factorization of both domains. *Topic correlation analysis* (Li et al., 2012) extracts both shared and domain-specific latent features and groups them, to support higher distribution gaps between domains.

Within the distinction between instance-transfer and feature-representation-transfer approaches, our method could be regarded as following the former, as no latent common space is learned. Instead, source documents are brought to the target domain, although in aggregated form and with no adaptation: they just serve to train a knowledge model and to bootstrap the iterative phase, as detailed in the next section.

Likely to traditional text classification, some methods leverage external knowledge bases: these can be helpful to link knowledge across domains. The method presented in (Wang et al., 2008) improves the cited co-clustering-based approach (Dai et al., 2007a) by representing documents with concepts extracted from Wikipedia. The *bridging information gap* method (Xiang et al., 2010) exploits instead an auxiliary domain acting as a bridge between source and target, using Wikipedia articles as a practical example. These methods usually offer very high performances, but need a suitable knowledge base for the context of the analyzed documents, which might not be easily available for overly specialized domains.

Beyond the presented works where domains differ only in the distribution of terms, methods for *cross-language* text classification exist, where source and target documents are written in different languages, so that there are few or no common words between the two domains. This scenario generally requires either some labeled documents for the target domain or an external knowledge base to be available: a dictionary for translation of single terms is often used. As examples, in (Ling et al., 2008b) is presented an approach based on *information bottleneck* where Chinese texts are translated into English to be classified, while the method in (Prettenhofer and Stein, 2010) is based on the structural correspondence learning cited above (Blitzer et al., 2006).

Other than text classification by topic, another related task on which domain adaptation is frequently used is *sentiment analysis*, where positive and negative opinions about specific objects (products, brands, etc.) must be distinguished: a usual motivating example is the need to extract knowledge from labeled reviews for some products to classify reviews for products of a different type, with possibly different terminologies. *Spectral feature alignment* (Pan and Yang, 2010) works by clustering together words specific for different domains leveraging the more general terms. In (Bollegala et al., 2013) a sentiment-sensitive thesaurus is built from possibly multiple source domains. In (Cheeti et al., 2013) a Naïve Bayes classifier on syntax trees-based features is used.

3 CROSS-DOMAIN LEARNING METHOD

This section describes in detail our method to classify documents in a target domain exploiting the knowledge of a source domain.

Inputs to the method are a set $\mathcal{D}_S$ of *source* or *in-domain* documents, which constitute the *source domain* and a disjoint set $\mathcal{D}_T$ of *target* or *out-of-domain* documents, making up the *target domain*; we denote with $\mathcal{D} = \mathcal{D}_S \cup \mathcal{D}_T$ their union. Each document in $\mathcal{D}$ is labeled with a single class from a set $\mathcal{C}$, according to two functions $C_S : \mathcal{D}_S \to \mathcal{C}$ and $C_T : \mathcal{D}_T \to \mathcal{C}$. As in any cross-domain classification method, we assume to have prior knowledge of C_S, while C_T is not known: the goal is to infer a function $\hat{C}_T : \mathcal{D}_T \to \mathcal{C}$ with maximal similarity to C_T.

The following subsections give details about the steps of the method: pre-processing of documents and feature extraction according to standard procedures, creation of initial representations for categories, training of a function to predict similarity between representations and iterative refining of categories representations. A discussion about time computational complexity is given thereafter.

3.1 Text Pre-processing

The method initially performs typical pre-processing operations on documents to transform each unstructured text into a structured representation.

A common tokenization process extracts single words from each document d, discarding punctuation, words shorter than 3 letters and all those found in a predefined list of stopwords; then the common Porter stemming algorithm is applied to group words with common stems (Porter, 1980). In the end, a set $W(d)$ of the processed words extracted from d is obtained, along with the number of occurrencies $f(d,t)$, also known as (*raw*) *frequency*, for each word (or *term*, equivalently) t.

The usual *bag of words* representation is used: each document d is reduced to a vector $\mathbf{w}_d$ of weights for each term t in a global feature set $\mathcal{W}$. As in other papers, features are filtered by Document Frequency (DF) thresholding, discarding all terms appearing in less than 3 documents, to trivially reduce complexity with negligible effects on accuracy. The remaining terms constitute the set $\mathcal{W}$ of features considered in all bags of words.

The weight of each term in each document is based on the numbers of occurrencies and determined by a defined weighting scheme. We use a slight variant of the common *tf-idf* (*Term Frequency, Inverse Document Frequency*) scheme (Salton and Buckley, 1988), computing the product between the *relative* frequency of a term in a document (instead of raw frequency, to avoid overweighting terms in longer documents) and the logarithm of the inverse frequency of the term across all the documents (to give less bias to overly common terms).

$$w_{d,t} = \underbrace{\frac{f(d,t)}{\sum_{\tau\in\mathcal{W}} f(d,\tau)}}_{tf} \cdot \underbrace{\log \frac{|\mathcal{D}|}{|\{\delta \in \mathcal{D} : t \in W(\delta)\}|}}_{idf} \quad (1)$$

Each document $d \in \mathcal{D}$ will be then represented by its weighted bag of words $\mathbf{w}_d$.

3.2 Initial Representation of Categories

Likely to single documents, whole categories are represented as bags of words.

For each category $c \in \mathcal{C}$, a bag of words can intuitively be built by averaging those of documents which are *representative* for it, i.e. those labeled with c. As no prior knowledge of how documents in $\mathcal{D}_T$ are labeled is available, documents in $\mathcal{D}_S$ are used instead, as the labeling function C_S is known. Each category c is then represented by the set $R_c^0 = \{d \in \mathcal{D}_S : C_S(d) = c\}$ of in-domain documents labeled with it. It is then sufficient to compute the mean weight of each term in each category, thus obtaining a representation $\mathbf{w}_c^0$ for each category $c \in \mathcal{C}$.

$$\mathbf{w}_c^0 = \frac{1}{|R_c^0|} \sum_{d \in R_c^0} \mathbf{w}_d \qquad (2)$$

The “0” indices denote that these are *initial* representations, which constitute the starting point for the subsequent iterative phase.

3.3 Text Similarity Measure

We need a function $\Phi : \mathbb{R}^n \times \mathbb{R}^n \to [0,1]$ which, given two bags of words with $n = |\mathcal{W}|$ features each, computes a *relatedness score* between the two of them. Specifically, given a document d and a category c, we refer to $\Phi(\mathbf{w}_d, \mathbf{w}_c^0)$ as the *absolute* likelihood of d being labeled with c, which is independent from other documents and categories.

A basic function commonly used to determine the relatedness of two bag of words is the *cosine similarity*, defined for two generic vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ as:

$$\cos(\mathbf{a}, \mathbf{b}) = \frac{\sum_{i=1}^n a_i \cdot b_i}{\sqrt{\sum_{i=1}^n a_i^2} \cdot \sqrt{\sum_{i=1}^n b_i^2}} \qquad (3)$$

This measure is widely used to compare documents in form of bags of words against each other, as it effectively spots similar distributions of terms in the documents. So, when computing the similarity $\cos(\mathbf{w}_d, \mathbf{w}_c^0)$ between bags representing a document d and a category c, we expect it to be significantly higher if they are effectively *related*, i.e. if c is the correct label for d. Assuming that values of the cosine similarity for couples of related bags are distributed according to a random variable Y_+ and that values for couples of unrelated bags are distributed in another random variable Y_-, then we predict that $E(Y_+) > E(Y_-)$ holds.

However, no prior knowledge is available of “how high” and “how low” should be the cosine similarity for pairs of related and unrelated bags, respectively. More formally, distributions of Y_+ and Y_- are unknown and we are not allowed to suppose that they are the same across different contexts.

To address this issue, suitable knowledge can be extracted from the source domain, whose labeling of documents is known: the values of cosine similarity between in-domain documents and categories can be measured by using the previously extracted bags of words. In practice, all the possible couples $(d,c) \in \mathcal{D}_S \times \mathcal{C}$ made of an in-domain document and a category are considered, computing for each the cosine similarity between respective bags: these values are used as samples from the Y_+ and Y_- distributions.

To extract knowledge from these samples, we fit a univariate logistic regression model (Hosmer Jr and Lemeshow, 2004): this procedure returns a function π returning the absolute likelihood for two bags of being related, given their cosine similarity.

$$\pi(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \qquad (4)$$

Considering, for each $(d,c) \in \mathcal{D}_S \times \mathcal{C}$, $x_{d,c} = \cos(\mathbf{w}_d, \mathbf{w}_c^0)$ and $y_{d,c}$ equal to 1 if $C_S(d) = c$ and to 0 otherwise, logistic regression is used to find values of β_0 and β_1 which maximize

$$\prod_{(d,c) \in \mathcal{D}_S \times C} \pi(x_{d,c})^{y_{d,c}} (1 - \pi(x_{d,c}))^{1 - y_{d,c}} \qquad (5)$$

The general function $\Phi(\mathbf{w}_d, \mathbf{w}_c^0) = \pi(\cos(\mathbf{w}_d, \mathbf{w}_c^0))$ obtained indicatesg the absolute likelihood of c being the correct label for d.

3.4 Iterative Refining of Target Categories

The Φ function can be used to classify out-of-domain documents by comparing their representation against those extracted from in-domain documents for the categories in $\mathcal{C}$: simply, for each document $d \in \mathcal{D}_T$, the predicted label is the category with the highest relatedness likelihood.

$$\hat{C}_T^0(d) = \operatorname*{argmax}_{c \in \mathcal{C}} \Phi(\mathbf{w}_d, \mathbf{w}_c^0) \qquad (6)$$

In the common case where the source and target domains are similar, yet somehow different, this does not yield optimal results. Infact, the representations used for categories are extracted from the source domain and thus reflect the distributions of words measured in it, while out-of-domain documents may refer to the same categories with different distributions of terms and even with some different terms.

However, we expect that this approximate classification does still yield the correct labeling for some amount of out-of-domain documents. Moreover, as the used function returns a likelihood for each document-category couple, we can weight the confidence of the classification for each document and we expect that documents classified with a very high degree of confidence almost surely are correctly labeled. For each document $d \in \mathcal{D}_T$ and each category $c \in \mathcal{C}$, we define the *relative* confidence or probability $p^0(d,c)$ of d being labeled with c as the normalization of the absolute likelihood across categories. In practice, for any d, values of $p^0(d,c)$ for each $c \in \mathcal{C}$ constitute a probability distribution (their sum is one).

$$p^0(d,c) = \frac{\Phi(\mathbf{w}_d, \mathbf{w}_c^0)}{\sum_{\gamma \in \mathcal{C}} \Phi(\mathbf{w}_d, \mathbf{w}_\gamma^0)} \qquad (7)$$

The formula implies that, in order for d to have an high probability $p^0(d,c)$ of being labeled with c, its representation must be very similar to that for c, but is also important that it is largely unrelated to different categories: if a document seems highly related to more than one category, none of them can be assigned with high relative confidence.

Having computed the probability distributions among categories for all documents in $\mathcal{D}_T$, those having high confidence of belonging to a specific category can be distinguished. Fixed a threshold ρ, we define for each category $c \in \mathcal{C}$ a set R_c^1 of documents having a probability of belonging to c superior either to this threshold and to probabilities for other categories: these documents are considered to be "surely enough" labeled with c. We impose $\rho \geq \frac{1}{|\mathcal{C}|}$, as any lower threshold would cause all documents to always be considered for their most probable category.

$$R_c^1 = \{d \in \mathcal{D}_T : p_0(d,c) > \rho \wedge \hat{C}_T^0(d) = c\} \qquad (8)$$

As sets of out-of-domain documents representing each category have been created, they can be exploited to build new representations for the categories. For each category c, similarly to how $\mathbf{w}_c^0$ was built by averaging bags of in-domain documents labeled with c, a new representation $\mathbf{w}_c^1$ is built by averaging documents in R_c^1. Documents of the source domain are no more used, as we experienced no significant accuracy improvement retaining them and because these bags should represent only the target domain.

$$\mathbf{w}_c^1 = \frac{1}{|R_c^0|} \sum_{d \in R_c^1} \mathbf{w}_d \qquad (9)$$

Having these new representations, the process can now return to the classification phase described at the beginning of this subsection and execute it again by substituting for each category c its initial representation $\mathbf{w}_c^0$ with its newly built one $\mathbf{w}_c^1$. We expect that, as new bags for categories better represent the target domain, the obtained classification gets closer to the real one. Moreover, we expect that documents which were classified with high confidence in the first run retain this distinction in the new run, as they contributed to build the new representation for their respective category, and even that new documents pass the confidence threshold for respective categories.

So, from the new categories representations, new probabilities $p^1(d,c)$ for each $(d,c) \in \mathcal{D}_T \times \mathcal{C}$ can be computed and, still considering the ρ confidence threshold, new sets R_c^2 of "sure" documents can be extracted for each category c, which in turn can be used to build further representations $\mathbf{w}_c^2$ for each $c \in \mathcal{C}$.

This cycle where bags of words for categories are progressively refined to better represent the target domain could be run indefinitely: we expect that the classification of these documents gets more accurate after each iteration.

Operationally, the method continues this iterative refining process until either a limit N_I of iterations is reached or the representations for all categories in $\mathcal{C}$ are identical to those from previous iteration. Infact, if in one iteration i the condition $R_c^i = R_c^{i-1}$ holds for each category $c \in \mathcal{C}$, then equal representations will be obtained through subsequent iterations ($\mathbf{w}_c^i = \mathbf{w}_c^{i-1}$). As a generalization of the second condition, where representations must be identical to those of the previous iteration, we may arrest the algorithm when, for each category, the cosine similarity between the latest representation and the previous one reaches a fixed threshold β, which should be slightly less than one (the default condition is equivalent to set $\beta = 1$). In this way, we may save some iterations where the representations have negligible variations.

Once a termination condition is met after a number $n_I \leq N_I$ of iterations, the final predicted label $\hat{C}_T(d)$ for each document $d \in \mathcal{D}_T$ is the one whose final representation $\mathbf{w}_c^{n_I}$ is most similar to its bag $\mathbf{w}_d$. In this step, as all target documents must be labeled, the most probable category for each is considered, even if its relative probability is not above ρ. In a likely case where new documents within the target domain must be classified after this training process without repeating it, we can compare each of them with all categories and assign it to the most similar one.

The pseudo-code for the whole described process (excluding the text pre-processing phase) is given in Figure 1: the equations given above for the first iteration are rewritten with an iteration counter i. Apart from the univariate logistic regression routine, for which there exist a number of implementations

```
Input: a bag of words $\mathbf{w}_d$ for each document $d \in \mathcal{D} = \mathcal{D}_S \cup \mathcal{D}_T$, set $\mathcal{C}$ of top categories, labeling $C_S : \mathcal{D}_S \to \mathcal{C}$ for source documents, confidence threshold $\rho$, maximum number $N_I$ of iterations
Output: predicted labeling $\hat{C}_T$ for documents of the target domain

for all $c \in \mathcal{C}$ do
    $R_c^0 \leftarrow \{d \in \mathcal{D}_S : C_S(d) = c\}$
    $\mathbf{w}_c^0 \leftarrow \frac{1}{|R_c^0|} \cdot \sum_{d \in R_c^0} \mathbf{w}_d$
end for
for all $(d,c) \in \mathcal{D}_S \times \mathcal{C}$ do
    $x_{d,c} \leftarrow \cos(\mathbf{w}_d, \mathbf{w}_c^0)$
    $y_{d,c} \leftarrow 1$ if $C_S(d) = c$, 0 otherwise
end for
$\pi \leftarrow$ LOGISTICREGRESSION($\mathbf{x}$,$\mathbf{y}$)
$\Phi(\mathbf{a},\mathbf{b}) \triangleq \pi(\cos(\mathbf{a},\mathbf{b}))$
$i \leftarrow 0$
while $i < N_I \wedge (i = 0 \vee \exists c \in \mathcal{C} : R_c^i \neq R_c^{i-1})$ do
    for all $(d,c) \in \mathcal{D}_T \times \mathcal{C}$ do
        $p^i(d,c) \leftarrow \frac{\Phi(\mathbf{w}_d,\mathbf{w}_c^i)}{\sum_{\gamma \in \mathcal{C}} \Phi(\mathbf{w}_d,\mathbf{w}_\gamma^i)}$
    end for
    for all $c \in \mathcal{C}$ do
        $A_c^i \leftarrow \{d \in \mathcal{D}_T : \operatorname{argmax}_{\gamma \in \mathcal{C}} p^i(d,\gamma) = c\}$
        $R_c^{i+1} \leftarrow \{d \in A_c^i : p^i(d,c) > \rho\}$
        $\mathbf{w}_c^{i+1} \leftarrow \frac{1}{|R_c^{i+1}|} \cdot \sum_{d \in R_c^{i+1}} \mathbf{w}_d$
    end for
    $i \leftarrow i + 1$
end while
for all $d \in \mathcal{D}_T$ do
    $\hat{C}_T(d) \leftarrow \operatorname{argmax}_{c \in \mathcal{C}} \Phi(\mathbf{w}_d, \mathbf{w}_c^i)$
end for
return $\hat{C}_T$
```

Figure 1: Pseudo-code for the iterative refining algorithm.

(Minka, 2003), the given code is self-contained and can be easily implemented in many languages.

3.5 Computational Complexity

The process performs many operations on vectors of length $|\mathcal{W}|$: while these operations would generally require a time linear in this length, given the prevalent sparsity of these vectors, we can use suitable data structures to bound both storage space and computation time linearly w.r.t. the mean number of non-zero elements. At this extent, we denote with l_D and l_C the mean number of non-zero elements in bags of words for documents and categories, respectively. By definition, we have $l_D \leq |\mathcal{W}|$ and $l_C \leq |\mathcal{W}|$; from our experiments (described in the next section) we also generally observed $l_D \ll l_C < |\mathcal{W}|$.

The construction of the initial representation for categories is done in $O(|\mathcal{D}_S| \cdot l_D)$ time, as all values of all documents representations must be summed up. Cosine similarities for vectors with l_D and l_C non-zero elements respectively can be computed in $O(l_D + l_C)$ time, which can be written as $O(l_C)$ given that $l_D < l_C$. To fit the logistic regression model, the cosine similarity for $N_S = |\mathcal{D}_S| \cdot |\mathcal{C}|$ pairs must be computed to acquire input data, which requires $O(l_c \cdot N_S)$ time; then the model can be fit with one of various optimization methods which are generally linear in the number N_S of data samples (Minka, 2003).

In each iteration of the refining phase, the method computes cosine similarity for $N_T = |\mathcal{D}_T| \cdot |\mathcal{C}|$ document-category pairs and normalizes them to obtain distribution probabilities in $O(N_T \cdot l_C)$ time; then, to build new bags of words for categories, up to $|\mathcal{D}_T|$ document bags must be summed up, which is done in $O(|\mathcal{D}_T| \cdot l_D)$ time. The sum of these two steps, always considering $l_D < l_C$, is $O(|\mathcal{D}_T| \cdot |\mathcal{C}| \cdot l_C)$, which must be multiplied by the final number n_I of iterations.

Summing up, the overall complexity of the method is $O(|\mathcal{D}_S| \cdot |\mathcal{C}| \cdot l_C + n_I \cdot |\mathcal{D}_T| \cdot |\mathcal{C}| \cdot l_C)$, which can be simplified as $O(n_I \cdot |\mathcal{D}| \cdot |\mathcal{C}| \cdot l_C)$, with $l_C \leq |\mathcal{W}|$. The complexity is therefore linear in the number $|\mathcal{D}|$ of documents, the number $|\mathcal{C}|$ of top categories (usually very small), the mean number l_C of mean terms per category (having $|\mathcal{W}|$ as an upper bound) and the number n_I of iterations in the final phase, which in our experiments is always less than 20. This complexity is comparable to the other methods which are considered in the upcoming experiments section.

4 EXPERIMENTS

To assess the performances of the method described above, we performed some experiments on sets of documents already used as a test bed for other cross-domain text classification methods, to be able to compare our results with them.

The method has been implemented in a software framework written in Java. To fit logistic regression models, we relied upon the Weka machine learning software (Hall et al., 2009).

4.1 Benchmark Datasets

For our experiments, we considered three text collections commonly used in cross-domain classification due to their classes taxonomy, exhibiting a shallow hierarchical structure. This allows to isolate a small set of *top categories*, each including a number of *subcategories* in which documents are organized.

Each possible input dataset is set up by choosing a small set of top categories of a collection constituting the set $\mathcal{C}$ and splitting documents of these categories into two groups: one contains documents of some branches of the top categories and is used as the source domain, the other one containing documents of different sub-categories is used as the target domain. By labeling each document in the two domains with its top-category, we obtain suitable datasets.

The ***20 Newsgroups*** collection[1] (or *20NG*) is a set of posts from 20 different Usenet discussion groups, which are arranged in a hierarchy, each represented by almost 1,000 posts. We consider the 4 most frequent top categories *comp*, *rec*, *sci* and *talk*, each represented by 4 sub-categories (5 for *comp*). Each test involves two top categories: the source domain is composed by documents of 2 or 3 sub-categories for each of them, the target domain is composed by the remaining 2 or 3 sub-categories each. We considered 6 different problems with different pairs of top-categories, following the same sub-categories split used in other works (see e.g. (Dai et al., 2007a) for a table). We also considered four problems with documents drawn from three top categories, which are less commonly tested among other works.

The ***SRAA*** text collection[2] is also drawn from Usenet: it consists of 73,218 posts from discussion groups about simulated autos, simulated aviation, real autos and real aviation. With this setting, we can perform tests using two different sets of top categories: {*real*, *simulated*} and {*auto*, *aviation*}. In the first case, we used documents about aviation of both types for the source domain and about autos of both types for the target domain; the second case is similar, with simulated vehicles as the source domain and real vehicles as the target one. As the four groups are highly unbalanced in the collection as is, tests are performed on a selection of 16,000 documents, 4,000 for each group, likely to other works.

The ***Reuters-21578*** collection[3] contains 21,578 newswire stories about economy and finance collected from Reuters in 1992. In this collection, documents are labeled with 5 types of labels, among which *orgs*, *people* and *places* are commonly used as top categories: we considered the three possible pairs of them, using the same split between source and target employed by other works where sub-categories are evenly divided.

4.2 Setup and Evaluation

We performed tests on the datasets described above. The only parameters to be configured are the maximum number N_I of iterations, which we fixed at 20 and was rarely reached in our runs, and the confidence threshold ρ, for which we tested multiple values.

In each test run, to evaluate the goodness of the predicted labeling $\hat{C}_T$ with respect to the correct one C_T, likely to other works, we measure the *accuracy* as the ratio of documents in the target domain for which the correct label was predicted: as almost all target domains have evenly distributed documents across categories, this is a fairly valid measure.

$$Acc(C_T, \hat{C}_T) = \frac{|\{d \in \mathcal{D}_T : \hat{C}_T(d) = C_T(d)\}|}{|\mathcal{D}_T|} \quad (10)$$

For each test, we also report two *baseline* results: the *minimal* accuracy obtained by simply classifying out-of-domain documents using categories representations extracted from the source domain (we would obtain this by setting $N_I = 0$, i.e. suppressing the iterative phase) and the *maximal* accuracy which would be reached by classifying the same documents using both the regression model and the categories representations extracted from the target domain itself, assuming prior knowledge of its labeling (in other words, we set $\mathcal{D}_S = \mathcal{D}_T$ and $N_I = 0$). We consider these baseline results as lower and upper bounds for the real accuracy.

4.3 Results

Table 1 summarizes some relevant results for each considered dataset: the accuracy baselines, the results reported in other works and our results with the threshold ρ set to 0.54, including the number of iterations needed to terminate the refining phase. Specifically, we reported the available results from the following works, also cited in Section 2:

CoCC co-clustering (Dai et al., 2007a),

TPLSA topic-bridged PLSA (Xue et al., 2008),

CDSC spectral classification (Ling et al., 2008a),

MTrick matrix trifactorization (Zhuang et al., 2011),

TCA topic correlation analysis (Li et al., 2012).

We can see from the table that our approach performs better than reported methods in most cases.

In the table, we picked $\rho = 0.54$ as we determined empirically by our experiments that it generally yields optimal results. Being close to 0.5, in the common case with two top categories, few documents are generally ignored in the iterative refining phase.

[1] http://qwone.com/∼jason/20Newsgroups/

[2] http://people.cs.umass.edu/∼mccallum/data/sraa.tar.gz

[3] http://www.cse.ust.hk/TL/dataset/Reuters.zip

Table 1: Results of our method (on rightmost columns) on selected test datasets, compared with those reported by other works: the results in bold are the best for each dataset (excluding baselines).

	Baselines		Other methods					ρ = 0.54	
Dataset	min	max	CoCC	TPLSA	CDSC	MTrick[a]	TCA	Acc.	Iters.
20 Newsgroups									
comp vs sci	0.760	0.989	0.870	**0.989**	0.902	-	0.891	0.976	16
rec vs talk	0.641	0.998	0.965	0.977	0.908	*0.950*	0.962	**0.992**	9
rec vs sci	0.824	0.991	0.945	0.951	0.876	*0.955*	0.879	**0.984**	11
sci vs talk	0.796	0.990	0.946	0.962	0.956	*0.937*	0.940	**0.974**	11
comp vs rec	0.903	0.992	0.958	0.951	0.958	-	0.940	**0.980**	10
comp vs talk	0.966	0.995	0.980	0.977	0.976	-	0.967	**0.990**	8
comp vs rec vs sci	0.682	0.975	-	-	-	*0.932*	-	**0.940**	16
rec vs sci vs talk	0.486	0.991	-	-	-	*0.936*	-	**0.977**	15
comp vs sci vs talk	0.722	0.986	-	-	-	*0.921*	-	**0.971**	14
comp vs rec vs talk	0.917	0.991	-	-	-	*0.955*	-	**0.980**	9
SRAA									
real vs simulated	0.570	0.976	0.880	0.889	0.812	-	-	**0.936**	13
auto vs aviation	0.809	0.983	0.932	0.947	0.880	-	-	**0.962**	18
Reuters-21578									
orgs vs places	0.736	0.909	0.680	0.653	0.682	**0.768**	0.730	0.724	16
orgs vs people	0.779	0.918	0.764	0.763	0.768	0.808	0.792	**0.820**	13
people vs places	0.612	0.926	**0.826**	0.805	0.798	0.690	0.626	0.693	13

[a] Values for 20 Newsgroups collection reported by "MTrick" (in italic) actually are not computed on single runs, but are averages of multiple runs, each with an equal set of top categories, where a baseline document classifier trained on source domain and tested on target got an accuracy higher than 65%

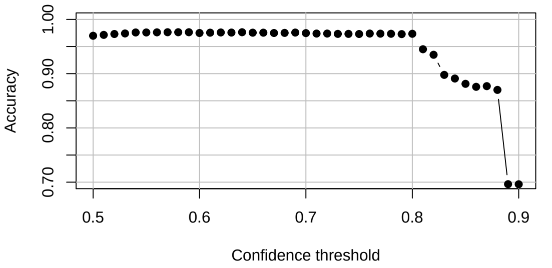

Figure 2: Accuracy on the *comp vs sci* dataset for different values of the ρ threshold.

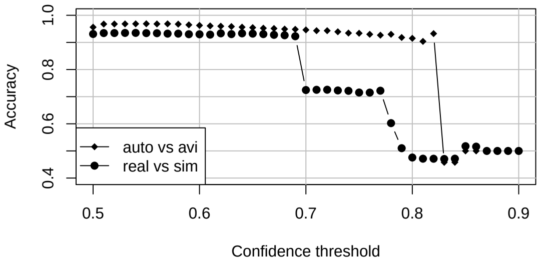

Figure 3: Accuracy on the two *SRAA* datasets for different values of the ρ threshold.

However, we observed that in many cases the threshold parameter ρ has little influence on the final accuracy, as long as it stays within a reasonable range of values: we show some examples. Figure 2 reports the accuracy on the *comp vs sci* dataset with different threshold values: it can be noted that the result scarcely varies for threshold values between 0.5 and 0.8; the same trend holds even for the other datasets with two top categories of 20 Newsgroups. Figure 3 shows the same plot for the two SRAA problems, showing just a different range for *real vs sim.* Instead, tests on 20NG with three top categories, where the minimum value for ρ is 1/3, generally yield high accuracies for thresholds between 0.45 and 0.6, as shown for example in Figure 4 for *comp vs rec vs sci.* On the Reuters collection, accuracy has a more unpredictable behavior as the threshold varies: this is probably due to the higher difficulty of distinguishing its top categories, as also appears from results of other works.

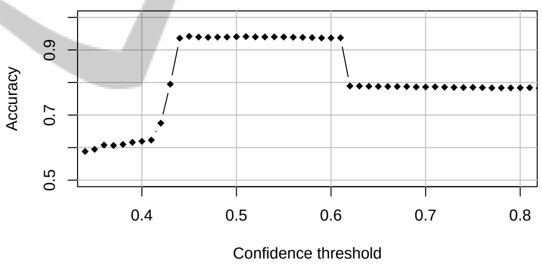

Figure 4: Accuracy on the *comp vs rec vs sci* dataset for different values of the ρ threshold.

In the results, we reported the number of iterations needed for the algorithm to reach the convergence condition, where categories representations

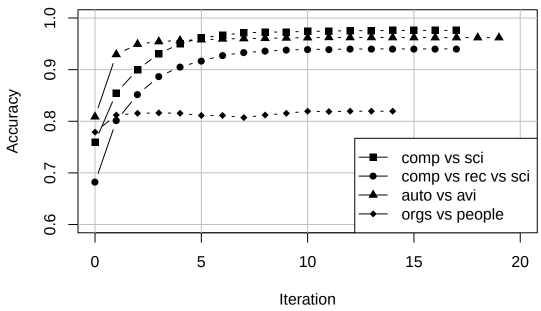

Figure 5: Intermediate accuracy at each iteration on different datasets.

stop changing between successive iterations. It is interesting to check what accuracy would be obtained with an anticipated termination of the algorithm, obtained by setting a lower value for the maximum number N_I of iterations. We report in Figure 5 the *intermediate* accuracy obtained on various datasets by limiting the N_I parameter. As stated above, for $N_I = 0$ the minimal accuracy is obtained. Accuracy generally grows faster in the first iterations and only has minor fluctuations in the successive iterations: generally, the result is that 5 iterations grant an accuracy at most 3% below the convergence value, while with 10 iterations the optimal result is within 1%. The parameter can then set as a tradeoff between accuracy and running time, while setting an high value (empirically, 20 or more) would still yield optimal accuracy with a reasonable running time.

Specifically, letting the algorithm reach the convergence condition, our running times for single tests, each run on two processing cores on virtualized hardware, have been within 2 minutes for the smaller Reuters-based datasets, between 5 and 7 minutes for problems with two top categories of 20NG and between 15 and 20 minutes for the remaining datasets with more documents. These times are roughly proportional to the number of iterations: the initial training phase on the source domain takes about the time of one iteration to compute the needed similarity values and few seconds to fit the regression model.

As said above, an alternative termination condition to reduce the number of iterations and consequently the running time, without compromising the accuracy, is to stop them when the cosine similarities of all categories between their own current and previous representations reach a given threshold β. Results of this variant with two different values of the threshold, compared to default results with $\beta = 1$, are given in Table 2. With the two picked values, we generally have a strong reduction of the number of iterations while maintaining the accuracy very close to the convergence value: in the tests with $\beta = 0.999$, the number of iterations is at least halved down in most cases and always drops below 10 with an accuracy which, excluding tests on Reuters, is at most 0.5% lower than the one obtained normally. A lower number of iterations directly impacts on the running time, which with $\beta = 0.999$ stayed within about 10 minutes even for larger datasets. The downside of this variant is the introduction of a new parameter, altough is shown that the two values given in the table generally work fine on all tested datasets.

Table 2: Accuracy (A, in thousandths) and number of iterations (I) for all datasets with different settings for the β similarity threshold for termination ($\beta = 1$ corresponds to the default termination condition).

$\beta \rightarrow$	1 (def.)		0.9999		0.999	
Dataset	A	I	A	I	A	I
20 Newsgroups						
comp vs sci	976	16	974	9	973	7
rec vs talk	992	9	992	8	990	4
rec vs sci	984	11	984	6	984	4
sci vs talk	974	11	974	6	970	4
comp vs rec	980	10	980	6	979	3
comp vs talk	990	8	990	4	990	2
comp rec sci	940	16	940	12	938	8
rec sci talk	977	15	976	10	976	9
comp sci talk	971	14	971	11	967	6
comp rec talk	980	9	979	4	979	3
SRAA						
real vs sim	936	13	939	9	936	4
auto vs avi	962	18	961	8	957	3
Reuters-21578						
orgs places	724	16	727	10	731	6
orgs people	820	13	820	12	812	7
people places	693	13	693	12	666	5

Summing up, the fixed values for the two settable parameters $\rho = 0.54$ and $N_I = 10$ seem to yield good results in almost any dataset, with the possibility of reducing N_I to trade off some accuracy for a lower running time. The alternative termination condition is a further possibility to limit the number of iterations with a parameter for which, likely to the other ones, globally valid values seem to exist.

While up to now we assumed to know in advance all documents of the target domain, in many real cases there is the need to classify documents which are not known while training the knowledge model. As stated before, in this case, we can simply compare new documents with the final categories representations and check which of them is the most similar for each document. To verify this, we performed additional tests on 20NG where in the iterative phase only a subset of documents in the target domain is known, while the final accuracy is evaluated as before on all of them.

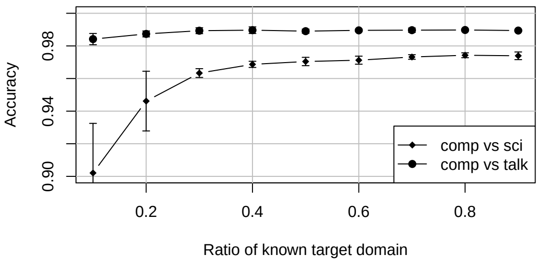

Figure 6: Average accuracy on *comp vs sci* and *comp vs talk* datasets when only a given ratio of the target documents is known during the iterative phase; each point is an average on 5 tests, with error bars indicating standard deviation.

Figure 6 reports the results for tests performed on 20 Newsgroups with $\rho = 0.54$ and $N_I = 20$ where only a fixed ratio of out-of-domain documents is considered to be known in the iterative phase: each point gives the average accuracy on five runs with different subsets randomly drawn from the target domain. For ease of readability, we just reported results for the two datasets with respectively the lowest and highest overall average accuracy: curves for other two-categories problems on 20NG follow the same trend and mostly lie between the two. Average accuracy is above 90% even when just the 10% of the target domain (less than 500 documents) is known, while using the 30% or more of the out-of-domain documents generally guarantees an accuracy of 96% at least.

5 CONCLUSION AND FUTURE WORK

We presented a conceptually simple and fairly efficient method for cross-domain text classification based on explicitly representing categories and computing their similarity to documents. The method works by initially creating representations from the source domain to start collecting sure classifications in the target domain, which are then used to progressively build better representations for it.

We tested the method on text collections commonly used as benchmarks for transfer learning tasks, obtaining fairly good performances with respect to other approaches. While the algorithm has few parameters to be set, they are shown to often have little influence on the result and that some fixed empirical values often yield (nearly) optimal accuracy.

The method can be extended to consider semantics of terms, either leveraging external knowledge (as in (Wang et al., 2008)) or statistical techniques like (P)LSA. Regarding applications, with suitable adaptations, we are considering testing it on related problems such as cross-language classification and sentiment analysis.

REFERENCES

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *The Journal of Machine Learning research*, 3:993–1022.

Blitzer, J., McDonald, R., and Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128. Association for Computational Linguistics.

Bollegala, D., Weir, D., and Carroll, J. (2013). Cross-domain sentiment classification using a sentiment sensitive thesaurus. *IEEE Transactions on Knowledge and Data Engineering*, 25(8):1719–1731.

Cheeti, S., Stanescu, A., and Caragea, D. (2013). Cross-domain sentiment classification using an adapted nive bayes approach and features derived from syntax trees. In *Proceedings of KDIR 2013, 5th International Conference on Knowledge Discovery and Information Retrieval*, pages 169–176.

Dai, W., Xue, G.-R., Yang, Q., and Yu, Y. (2007a). Co-clustering based classification for out-of-domain documents. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 210–219. ACM.

Dai, W., Xue, G.-R., Yang, Q., and Yu, Y. (2007b). Transferring naive bayes classifiers for text classification. In *Proceedings of the AAAI '07, 22nd national conference on Artificial intelligence*, pages 540–545.

Daumé III, H. (2007). Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263.

Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of CIKM '98, 7th International Conference on Information and Knowledge Management*, pages 148–155. ACM.

Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, volume 7, pages 1606–1611.

Gao, J., Fan, W., Jiang, J., and Han, J. (2008). Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 283–291. ACM.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Hosmer Jr, D. W. and Lemeshow, S. (2004). *Applied logistic regression*. John Wiley & Sons.

Huang, J., Smola, A. J., Gretton, A., Borgwardt, K. M., and Schölkopf, B. (2007). Correcting sample selection bias by unlabeled data. *Advances in neural information processing systems*, 19:601–608.

Joachims, T. (1997). A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *Proceedings of ICML '97, 14th International Conference on Machine Learning*, pages 143–151.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Proceedings of ECML-98, 10th European Conference on Machine Learning*, 1398:137–142.

Kohonen, T., Kaski, S., Lagus, K., Salojarvi, J., Honkela, J., Paatero, V., and Saarela, A. (2000). Self organization of a massive document collection. *IEEE Transactions on Neural Networks*, 11(3):574–585.

Li, L., Jin, X., and Long, M. (2012). Topic correlation analysis for cross-domain text classification. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Ling, X., Dai, W., Xue, G.-R., Yang, Q., and Yu, Y. (2008a). Spectral domain-transfer learning. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 488–496. ACM.

Ling, X., Xue, G.-R., Dai, W., Jiang, Y., Yang, Q., and Yu, Y. (2008b). Can chinese web pages be classified with english data source? In *Proceedings of the 17th international conference on World Wide Web*, pages 969–978. ACM.

Merkl, D. (1998). Text classification with self-organizing maps: Some lessons learned. *Neurocomputing*, 21(1):61–77.

Minka, T. P. (2003). A comparison of numerical optimizers for logistic regression. http://research.microsoft.com/en-us/um/people/minka/papers/logreg/.

Pan, S. J., Kwok, J. T., and Yang, Q. (2008). Transfer learning via dimensionality reduction. In *Proceedings of the AAAI '08, 23rd national conference on Artificial intelligence*, pages 677–682.

Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q. (2011). Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210.

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

Porter, M. F. (1980). An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137.

Prettenhofer, P. and Stein, B. (2010). Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1127.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Scott, S. and Matwin, S. (1998). Text classification using wordnet hypernyms. In *Use of WordNet in natural language processing systems: Proceedings of the conference*, pages 38–44.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47.

Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244.

Sugiyama, M., Nakajima, S., Kashima, H., Von Buenau, P., and Kawanabe, M. (2007). Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems 20*, volume 7, pages 1433–1440.

Wang, P., Domeniconi, C., and Hu, J. (2008). Using Wikipedia for co-clustering based cross-domain text classification. In *ICDM '08, 8th IEEE International Conference on Data Mining*, pages 1085–1090. IEEE.

Weigend, A. S., Wiener, E. D., and Pedersen, J. O. (1999). Exploiting hierarchy in text categorization. *Information Retrieval*, 1(3):193–216.

Xiang, E. W., Cao, B., Hu, D. H., and Yang, Q. (2010). Bridging domains using world wide knowledge for transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(6):770–783.

Xue, G.-R., Dai, W., Yang, Q., and Yu, Y. (2008). Topic-bridged plsa for cross-domain text classification. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 627–634. ACM.

Yang, Y. and Liu, X. (1999). A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49. ACM.

Zadrozny, B. (2004). Learning and evaluating classifiers under sample selection bias. In *Proceedings of the 21st International Conference on Machine Learning*, page 114. ACM.

Zhuang, F., Luo, P., Xiong, H., He, Q., Xiong, Y., and Shi, Z. (2011). Exploiting associations between word clusters and document classes for cross-domain text categorization. *Statistical Analysis and Data Mining*, 4(1):100–114.