Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach

Xavier Glorot(1) GLOROTXA@IRO.MONTREAL.CA
Antoine Bordes(2,1) BORDESAN@HDS.UTC.FR
Yoshua Bengio(1) BENGIOY@IRO.MONTREAL.CA
(1) Dept. IRO, Université de Montréal. Montréal (QC), H3C 3J7, Canada
(2) Heudiasyc, UMR CNRS 6599, Université de Technologie de Compiègne, 60205 Compiègne, France

Abstract

The exponential increase in the availability of online reviews and recommendations makes sentiment classification an interesting topic in academic and industrial research. Reviews can span so many different domains that it is difficult to gather annotated training data for all of them. Hence, this paper studies the problem of domain adaptation for sentiment classifiers, hereby a system is trained on labeled reviews from one source domain but is meant to be deployed on another. We propose a deep learning approach which learns to extract a meaningful representation for each review in an unsupervised fashion. Sentiment classifiers trained with this high-level feature representation clearly outperform state-of-the-art methods on a benchmark composed of reviews of 4 types of Amazon products. Furthermore, this method scales well and allowed us to successfully perform domain adaptation on a larger industrial-strength dataset of 22 domains.

1. Introduction

With the rise of social media such as blogs and social networks, reviews, ratings and recommendations are rapidly proliferating; being able to automatically filter them is a current key challenge for businesses looking to sell their wares and identify new market opportunities. This has created a surge of research in sentiment classification (or sentiment analysis), which aims to determine the judgment of a writer with respect to a given topic based on a given textual comment. Sentiment analysis is now a mature machine learning research topic, as illustrated with this review (Pang and Lee, 2008). Applications to many different domains have been presented, ranging from movie reviews (Pang *et al.*, 2002) and congressional floor debates (Thomas *et al.*, 2006) to product recommendations (Snyder and Barzilay, 2007; Blitzer *et al.*, 2007).

This large variety of data sources makes it difficult and costly to design a robust sentiment classifier. Indeed, reviews deal with various kinds of products or services for which vocabularies are different. For instance, consider the simple case of training a system analyzing reviews about only two sorts of products: *kitchen appliances* and *DVDs*. One set of reviews would contain adjectives such as "malfunctioning", "reliable" or "sturdy", and the other "thrilling", "horrific" or "hilarious", etc. Therefore, data distributions are different across domains. One solution could be to learn a different system for each domain. However, this would imply a huge cost to annotate training data for a large number of domains and prevent us from exploiting the information shared across domains. An alternative strategy, evaluated here, consists in learning a single system from the set of domains for which labeled and unlabeled data are available and then apply it to any target domain (labeled or unlabeled). This only makes sense if the system is able to discover intermediate abstractions that are shared and meaningful across domains. This problem of training and testing models on different distributions is known as domain adaptation (Daumé III and Marcu, 2006).

In this paper, we propose a Deep Learning approach for the problem of domain adaptation of sentiment classifiers. The promising new area of Deep Learning has emerged recently; see (Bengio, 2009) for a review. Deep Learning is based on algorithms for **discovering intermediate representations** built in a

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011. Copyright 2011 by the author(s)/owner(s).

hierarchical manner. Deep Learning relies on the discovery that unsupervised learning could be used to set each level of a hierarchy of features, one level at a time, based on the features discovered at the previous level. These features have successfully been used to initialize deep neural networks (Hinton and Salakhutdinov, 2006; Hinton *et al.*, 2006; Bengio *et al.*, 2006). Imagine a probabilistic graphical model in which we introduce latent variables which correspond to the true explanatory factors of the observed data. It is likely that answering questions and learning dependencies in the space of these latent variables would be easier than answering questions about the raw input. A simple linear classifier or non-parametric predictor trained from as few as one or a few examples might be able to do the job. The key to achieving this is learning better representations, mostly from unlabeled data: how this is done is what differentiates Deep Learning algorithms.

The Deep Learning system we introduce in Section 3 is designed to use unlabeled data to extract high-level features from reviews. We show in Section 4 that sentiment classifiers trained with these learnt features can: (i) surpass state-of-the-art performance on a benchmark of 4 kinds of products and (ii) successfully perform domain adaptation on a large-scale data set of 22 domains, beating all of the baselines we tried.

2. Domain Adaptation

Domain adaptation considers the setting in which the training and testing data are sampled from different distributions. Assume we have two sets of data: a *source* domain S providing labeled training instances and a *target* domain T providing instances on which the classifier is meant to be deployed. We do not make the assumption that these are drawn from the same distribution, but rather that S is drawn from a distribution p_S and T from a distribution p_T. The learning problem consists in finding a function realizing a good *transfer* from S to T i.e. it is trained on data drawn from p_S and generalizes well on data drawn from p_T.

Deep Learning algorithms learns intermediate concepts between raw input and target. Our intuition for using it in this setting is that these intermediate concepts could yield better transfer across domains. Suppose for example that these intermediate concepts indirectly capture things like product quality, product price, customer service, etc. Some of these concepts are general enough to make sense across a wide range of domains (corresponding to products or services, in the case of sentiment analysis). Because the same words or tuples of words may be used across domains to indicate the presence of these higher-level concepts, it should be possible to discover them. Furthermore, because Deep Learning exploits unsupervised learning to discover these concepts, one can exploit the large amounts of unlabeled data across all domains to learn these intermediate representations. Here, as in many other Deep Learning approaches, we do not engineer what these intermediate concepts should be, but instead use generic learning algorithms to discover them.

2.1. Related Work

Learning setups relating to domain adaptation have been proposed before and published under different names. Daumé III and Marcu (2006) formalized the problem and proposed an approach based on a mixture model. A general way to address domain adaptation is through instance weighting, in which instance-dependent weights are added to the loss function (Jiang and Zhai, 2007). Another solution to domain adaptation can be to transform the data representations of the source and target domains so that they present the same joint distribution of observations and labels. Ben-David *et al.* (2007) formally analyze the effect of representation change for domain adaptation while Blitzer *et al.* (2006) propose the Structural Correspondence Learning (SCL) algorithm that makes use of the unlabeled data from the target domain to find a low-rank joint representation of the data.

Finally, domain adaptation can be simply treated as a standard semi-supervised problem by ignoring the domain difference and considering the source instances as labeled data and the target ones as unlabeled data (Dai *et al.*, 2007). In that case, the framework is very close to that of self-taught learning (Raina *et al.*, 2007), in which one learns from labeled examples of some categories as well as unlabeled examples from a larger set of categories. The approach of Raina *et al.* (2007) relies crucially on the unsupervised learning of a representation, like the approach proposed here.

2.2. Applications to Sentiment Classification

Sentiment analysis and domain adaptation are closely related in the literature, and many works have studied domain adaptation exclusively for sentiment analysis. Among those, a large majority propose experiments performed on the benchmark made of reviews of Amazon products gathered by Blitzer *et al.* (2007).

Amazon data The data set proposes more than 340,000 reviews regarding 22 different product types[1] and for which reviews are labeled as either positive

[1]The data are available from `http://www.cs.jhu.edu/~mdredze/datasets/sentiment/`. It is actually composed of 25 domains but we removed 3 of them which were very small (less than 400 instances in total).

Table 1. **Amazon data statistics.** This table depicts the number of training, testing and unlabeled examples for each domain, as well as the portion of negative training examples for both versions of the data set.

Domain	Train size	Test size	Unlab. size	% Neg. ex
Complete (large-scale) data set				
Toys	6318	2527	3791	19.63%
Software	1032	413	620	37.77%
Apparel	4470	1788	2682	14.49%
Video	8694	3478	5217	13.63%
Automotive	362	145	218	20.69%
Books	10625	10857	32845	12.08%
Jewelry	982	393	589	15.01%
Grocery	1238	495	743	13.54%
Camera	2652	1061	1591	16.31%
Baby	2046	818	1227	21.39%
Magazines	1195	478	717	22.59%
Cell	464	186	279	37.10%
Electronics	10196	4079	6118	21.94%
DVDs	10625	9218	26245	14.16%
Outdoor	729	292	437	20.55%
Health	3254	1301	1952	21.21%
Music	10625	24872	88865	8.33%
Videogame	720	288	432	17.01%
Kitchen	9233	3693	5540	20.96%
Beauty	1314	526	788	15.78%
Sports	2679	1072	1607	18.75%
Food	691	277	415	13.36%
(Smaller-scale) benchmark				
Books	1600	400	4465	50%
Kitchen	1600	400	5945	50%
Electronics	1600	400	5681	50%
DVDs	1600	400	3586	50%

or negative. As detailed in Table 1 (top), there is a vast disparity between domains in the total number of instances and in the proportion of negative examples.

Since this data set is heterogeneous, heavily unbalanced and large-scale, a smaller and more controlled version has been released. The reduced data set contains 4 different domains: *Books*, *DVDs*, *Electronics* and *Kitchen* appliances. There are 1000 positive and 1000 negative instances for each domain, as well as a few thousand unlabeled examples. The positive and negative examples are also exactly balanced (see the bottom section of Table 1 for details). This latter version is used as a benchmark in the literature. To the best of our knowledge, this paper will contain the first published results on the large Amazon dataset.

Compared Methods In the original paper regarding the smaller 4-domain benchmark dataset, Blitzer *et al.* (2007) adapt Structural Correspondence Learning (SCL) for sentiment analysis. Li and Zong (2008) propose the Multi-label Consensus Training (MCT) approach which combines several base classifiers trained with SCL. Pan *et al.* (2010) first use a Spectral Feature Alignment (SFA) algorithm to align words from different source and target domains to help bridge the gap between them. These 3 methods serve as comparisons in our empirical evaluation.

3. Deep Learning Approach

3.1. Background

If Deep Learning algorithms are able to capture, to some extent, the underlying generative factors that explain the variations in the input data, what is really needed to exploit that ability is for the learned representations to help in **disentangling the underlying factors of variation**. The simplest and most useful way this could happen is if some of the features learned (the individual elements of the learned representation) are mostly related to only some of these factors, perhaps only one. Conversely, it would mean that such features would have *invariant properties*, i.e., they would be highly specific in their response to a subset (maybe only one) of these factors of variation and insensitive to the others. This hypothesis was tested by Goodfellow *et al.* (2009), for images and geometric invariances associated with movements of the camera.

It is interesting to evaluate Deep Learning algorithms on sentiment analysis for several reasons. First, if they can extract features that somewhat disentangle the underlying factors of variation, this would likely help to perform transfer across domains, since we expect that there exist generic concepts that characterize product reviews across many domains. Second, for our Amazon datasets, we know some of these factors (such as whether or not a review is about a particular product, or is a positive appraisal for that product), so we can use this knowledge to quantitatively check to what extent they are disentangled in the learned representation: domain adaptation for sentiment analysis becomes a medium for better understanding deep architectures. Finally, even though Deep Learning algorithms have not yet been evaluated for domain adaptation of sentiment classifiers, several very interesting results have been reported on other tasks involving textual data, beating the previous state-of-the-art in several cases (Salakhutdinov and Hinton, 2007; Collobert and Weston, 2008; Ranzato and Szummer, 2008).

3.2. Stacked Denoising Auto-encoders

The basic framework for our models is the Stacked Denoising Auto-encoder (Vincent *et al.*, 2008). An auto-encoder is comprised of an encoder function $h(\cdot)$ and a decoder function $g(\cdot)$, typically with the dimension of $h(\cdot)$ smaller than that of its argument. The reconstruction of input x is given by $r(x) = g(h(x))$, and auto-encoders are typically trained to minimize a form of reconstruction error $loss(x, r(x))$. Examples of reconstruction error include the squared error, or like here, when the elements of x or $r(x)$ can be considered as probabilities of a discrete event, the Kullback-

Liebler divergence between elements of x and elements of $r(x)$. When the encoder and decoder are linear and the reconstruction error is quadratic, one recovers in $h(x)$ the space of the principal components (PCA) of x. Once an auto-encoder has been trained, one can stack another auto-encoder on top of it, by training a second one which sees the encoded output of the first one as its training data. Stacked auto-encoders were one of the first methods for building deep architectures (Bengio *et al.*, 2006), along with Restricted Boltzmann Machines (RBMs) (Hinton *et al.*, 2006). Once a stack of auto-encoders or RBMs has been trained, their parameters describe multiple levels of representation for x and can be used to initialize a supervised deep neural network (Bengio, 2009) or directly feed a classifier, as we do in this paper.

An interesting alternative to the ordinary auto-encoder is the Denoising Auto-encoder (Vincent *et al.*, 2008) or DAE, in which the input vector x is stochastically corrupted into a vector $\tilde{x}$, and the model is trained to *denoise*, i.e., to minimize a denoising reconstruction error $loss(x, r(\tilde{x}))$. Hence the DAE cannot simply copy its input $\tilde{x}$ in its code layer $h(\tilde{x})$, even if the dimension of $h(\tilde{x})$ is greater than that of $\tilde{x}$. The denoising error can be linked in several ways to the likelihood of a generative model of the distribution of the uncorrupted examples x (Vincent, 2011).

3.3. Proposed Protocol

In our setting we have access to unlabeled data from various domains, and to the labels for one source domain only. We tackle the problem of domain adaptation for sentiment classifiers with a two-step procedure.

First, a higher-level feature extraction is learnt in an unsupervised fashion from the text reviews of all the available domains using a Stacked Denoising Auto-encoder (SDA) with rectifier units (i.e. $max(0, x)$) for the code layer. RBMs with (soft) rectifier units have been introduced in (Nair and Hinton, 2010). We have used such units because they have been shown to outperform other non-linearities on a sentiment analysis task (Glorot *et al.*, 2011). The SDA is learnt in a greedy layer-wise fashion using stochastic gradient descent. For the first layer, the non-linearity of the decoder is the logistic sigmoid, the corruption process is a masking noise (i.e. each active input has a probability P to be set to 0)[2] and the training criterion is the Kullback-Liebler divergence. The rectifier non-linearity is too hard to be used on “output” units: reconstruction error gradients would not flow if the reconstruction was 0 (argument of the rectifier is negative) when the target is positive. For training the DAEs of upper layers, we use the softplus activation function (i.e. $\log(1 + \exp(x))$, a smooth version of the rectifier) as non-linearity for the decoder output units. We also use the squared error as reconstruction error criterion and a Gaussian corruption noise, which is added *before* the rectifier non-linearity of the input layer in order to keep the sparsity of the representation. The code layer activations (after the rectifier), at different depths, define the new representations

In a second step, a linear classifier is trained on the transformed labeled data of the source domain. Support Vector Machines (SVM) being known to perform well on sentiment classification (Pang *et al.*, 2002), we use a linear SVM with squared hinge loss. This classifier is eventually tested on the target domain(s).

3.4. Discussion

The previous protocol exhibits appealing properties for domain adaptation of sentiment classifiers.

Existing domain adaptation methods for sentiment analysis focus on the information from the source and target distributions, whereas the SDA unsupervised learning can use data from other domains, *sharing the representation across all those domains*. This also reduces the computation required to transfer to several domains because a single round of unsupervised training is required, and allows us to *scale well with large amount of data* and consider real-world applications.

The code learned by the SDA is a *non-linear mapping of the input* and can therefore encode complex data variations. To the best of our knowledge, existing domain adaptation methods for sentiment analysis map inputs into a new or an augmented space using only linear projections. Furthermore, rectifier non-linearities have the the nice ability to naturally provide *sparse representations* (with exact zeros) for the code layer, which are well suited to linear classifiers and are efficient with respect to computational cost and memory use.

4. Empirical Evaluation

4.1. Experimental Setup

For both data sets, the preprocessing corresponds to the setting of (Blitzer *et al.*, 2007): each review text is treated as a bag-of-words and transformed into binary vectors encoding the presence/absence of unigrams and bigrams. For computational reasons, only the 5000 most frequent terms of the vocabulary of unigrams and bigrams are kept in the feature set. We

[2] We also tried to set inactive inputs to 1 with a different probability but we did not observe any improvement.

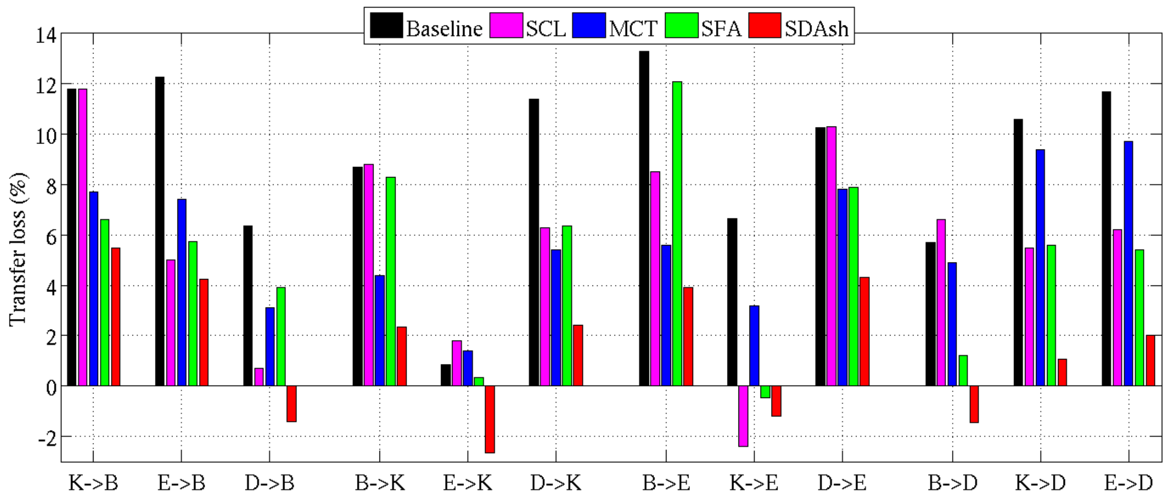
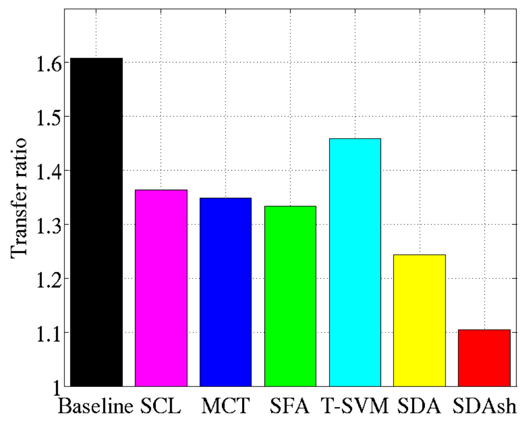


Figure 1. **Transfer losses on the Amazon benchmark** of 4 domains: *Kitchen*(K), *Electronics*(E), *DVDs*(D) and *Books*(B). All methods are trained on the labeled set of one domain and evaluated on the test sets of the others. SDA$_{sh}$ outperforms all others on 11 out of 12 cases.



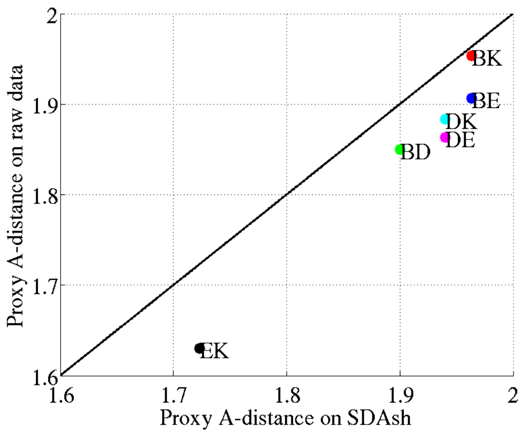


Figure 2. *Left:* **Transfer ratios** on the Amazon benchmark. Both SDA-based systems outperforms the rest even if SDA$_{sh}$ is better. *Right:* **Proxy A-distances** between domains of the Amazon benchmark for the 6 different pairs. Transforming data with SDA$_{sh}$ increases the proxy A-distance.

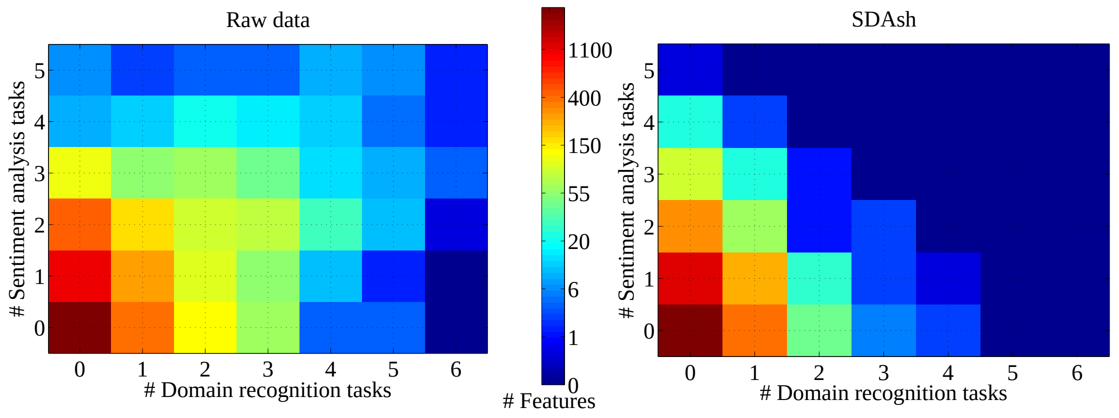


Figure 3. **L1 feature selection on the Amazon benchmark.** Both graphs depict the number of tasks of domain recognition (x-axis) and sentiment analysis (y-axis) in which a feature is re-used by L1-classifiers trained on raw features (*left*) or features transformed by SDA$_{sh}$. (*right*). See Section 4.3 for details.

use the train/test splits given in Table 1. For all experiments, the *baseline* is a linear SVM trained on the raw data whereas our method, denoted SDA$_{sh}$, corresponds to the same kind of SVM but trained and tested on data for which features have been transformed by the system described in Section 3. The hyper-parameters of all SVMs are chosen by cross-validation on the training set.

For SDA$_{sh}$, we explored an extensive set of hyper-parameters: a masking noise probability in $\{0.0, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9\}$, (its optimal value was usually high: 0.8); a Gaussian noise standard deviation for upper layers in $\{0.01, 0.1, 0.25, 0.5, 1\}$; a size of hidden layers in $\{1000, 2500, 5000\}$, (5000 always gave the best performance); an L_1 regularization penalty on the activation values in $\{0.0, 10^{-8}, 10^{-5}, 10^{-3}, 10^{-2}\}$; a learning rate in $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. All values were selected w.r.t. the averaged in-domain validation error. All algorithms were implemented using the Theano library (Bergstra *et al.*, 2010).

4.2. Metrics

We denote by $e(S, T)$, the *transfer error*, defined as the test error obtained by a method trained on the source domain S and tested on the target domain T ($e(T, T)$ is termed the *in-domain error*). The main point of comparison in domain adaptation is the *baseline in-domain error*, denoted $e_b(T, T)$, which corresponds to the test error obtained by the baseline method, i.e. a linear SVM on raw features trained and tested on the raw features of the target domain.

With these definitions, we can define the standard domain adaptation metric: the **transfer loss** t. It is the difference between the transfer error and the in-domain baseline error i.e. $t(S, T) = e(S, T) - e_b(T, T)$ for a source domain S and a target domain T.

Unfortunately, when one deals with a large number of heterogeneous domains with different difficulties (as with the large Amazon data), the transfer loss is not satisfactory. In addition, taking its mean over all possible couples of source-target domains is uninformative. Hence, we also introduce the following metrics:

- **Transfer ratio** $\mathcal{Q}$: it also characterizes the transfer but is defined by replacing the difference by a quotient in t because this is less sensitive to important variations of in-domain errors, and thus more adapted to averaging. We report its mean over all source-target couples of the data set: $\mathcal{Q} = \frac{1}{n} \sum_{(S,T)_{S \neq T}} \frac{e(S,T)}{e_b(T,T)}$ (with n the number of couples (S, T) with $S \neq T$).
- **In-domain ratio** $\mathcal{I}$: some domain adaptation methods, like ours, transform the feature representation of all the domains, including the source. Thus in-domain errors of such methods are different from those of the baseline. The in-domain ratio measures this and is defined by: $\mathcal{I} = \frac{1}{m} \sum_{S} \frac{e(T,T)}{e_b(T,T)}$ (with m the total number of domains).

4.3. Benchmark Experiments

On the benchmark of 4 domains, we compare our domain adaptation protocol with the 3 methods from the literature introduced in Section 2.2: SCL, SFA and MCT. We report the results from the original papers, which have been obtained using the whole feature vocabulary and on different splits, but of identical sizes as ours. From our experience, results are consistent whatever the train/test splits as long as set sizes are preserved. Hence, one can check that all baselines achieve similar performances. We also report results obtained by a Transductive SVM (Sindhwani and Keerthi, 2006) trained in a standard semi-supervised setup: the training set of the source domain is used as labeled set, and the training set of the other domains as the unlabeled set.[3] On this data set with a relatively small number of training instances, our unsupervised feature extractor is made of a single layer of 5000 units.

Main results Figure 1 depicts the transfer loss for all methods and for all source-target domain pairs. The best transfer is achieved by the SVM trained on our transformed features in 11 out of 12 cases (SCL is only slightly better in *Kitchen* → *Electronics*) and significantly better for 8 cases. Interestingly, for each target domain, there is one case of negative transfer loss for SDA$_{sh}$: an SVM trained on a different domain can outperform an SVM trained on the target domain because of the quality of our features.

Figure 2 (left) depicts the transfer ratio for the same methods plus the transductive SVM (T-SVM) and a second version of our system denoted SDA. Contrary to SDA$_{sh}$, the unsupervised training of SDA has not been performed on all the available domains but on couples, as does SCL: for instance, to transfer from *Books* to *DVD*, the feature extractor of SDA is trained on reviews from these 2 domains only. The transfer ratio for SDA being higher than for SDA$_{sh}$, we can conclude that sharing the unsupervised pre-training across all domains (even on those which are not directly concerned) is beneficial, as expected. Figure 2 also shows that the combination of an unsupervised and a super-

[3] Non-linear models (MLPs) were also investigated, with a similar protocol as presented in Section 4.4, but they did not reach the performance in transfer of the baseline. We believe this is due to the small training set size.

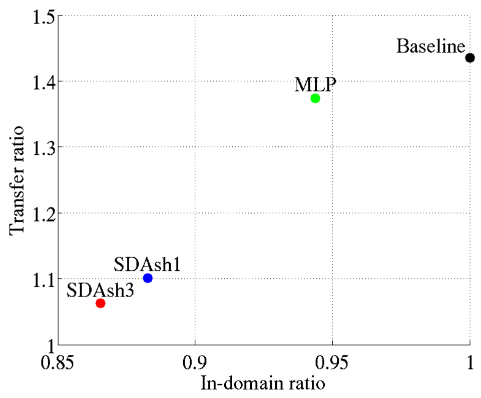


Figure 4. **Transfer ratios according to in-domain ratios on the large-scale Amazon data.** Systems based on SDA$_{sh}$ are better for both metrics and depth helps.

vised phase performed by SDA$_{sh}$ (and SDA) outperforms the pure semi-supervised T-SVM. In term of absolute classification performance, we obtained the following averaged transfer generalization errors: Baseline - 24.3%, SFA - 21.3%, SDA$_{sh}$ - 16.7%.

A-distance The A-distance is a measure of similarity between two probability distributions. Ben-David *et al.* (2007) showed that the A-distance between the source and target distributions is a crucial part of an upper generalization bound for domain adaptation. They hypothesized that it should be difficult to discriminate between the source and target domains in order to have a good transfer between them, because this would imply similar feature distributions. In practice, computing the exact A-distance is impossible and one has to compute a proxy. Hence, we measured the generalization error ϵ of a linear SVM classifier trained to discriminate between two domains. Our proxy for the A-distance is then defined as $\hat{d}_A = 2(1 - 2\epsilon)$.

Figure 2 (right) reports the results for each pair of domains. Surprisingly, $\hat{d}_A$ is *increased* in the new feature space: domain recognition is improved by the unsupervised feature extraction of SDA$_{sh}$. Consequently, following Ben-David *et al.* (2007), the representation of SDA$_{sh}$ should hurt transfer, but we also observe an improvement (see Figure 1). An explanation could be that the unsupervised feature extraction disentangles domain specific and sentiment polarity information.

To test this hypothesis, we trained an L_1-regularized SVM to select the most relevant features on 6 domain recognition tasks (one per domain pair), and 5 sentiment analysis tasks (one per domain plus all domains together). Figure 3 shows a histogram of the number of tasks associated with individual features, separated into the number of domain vs sentiment tasks. The color level at coordinate (n, m) indicates the number of features that have been re-used for n sentiment analysis tasks and for m domain recognition tasks. Comparing graphs obtained using raw data and data transformed by the SDA$_{sh}$ confirms our hypothesis: relevant features for domain recognition and sentiment analysis are far less overlapping in the latter case. Indeed, a complete feature disentangling would lead to a graph for which only the first column and the bottom line would be colored, indicating that each feature is either used for domain recognition *or* for sentiment classification, but not both. Transforming raw data with SDA$_{sh}$ brings features closer to that pattern.

4.4. Large-Scale Experiments

We now present results obtained on the larger version of the data. These conditions are more realistic and a better representation of the real-world than those of the previous experiments: more domains with different and larger sizes, different ratios between positive and negative examples, etc. We compare 3 methods in addition to the baseline: our feature extractor with either one (SDA$_{sh}$1) or 3 layers (SDA$_{sh}$3) of 5000 units, and a multi-layer perceptron (MLP) with the following architecture: a softmax logistic regression on top of one hidden layer with 5000 hyperbolic tangent units.

Figure 4 presents the transfer ratio of each model according to their in-domain ratio. Those results correspond to the following averaged transfer generalization errors: (Baseline) - 14.5%, (MLP) - 13.9%, (SDA$_{sh}$1) - 11.5% and (SDA$_{sh}$3) - 10.9%. Despite the large number of domains and their heterogeneity, there is a significant improvement for both SDA systems. The performance of the MLP shows that the non-linearity helps but is not sufficient to gather all necessary information from data: one needs an unsupervised phase which can encompass data from all domains. One can also verify that, on this large-scale problem, a single layer is not enough to reach optimal performance. Stacking 3 layers yields the best representation from the data. It is worth noting that the improvement of SDA$_{sh}$3 compared to the baseline is higher on the y-axis than on the x-axis: the representation learnt by SDA$_{sh}$3 is more beneficial for transfer than in-domain and is thus truly tailored to domain adaptation.

5. Conclusion

This paper has demonstrated that a Deep Learning system based on Stacked Denoising Auto-Encoders with sparse rectifier units can perform an unsupervised feature extraction which is highly beneficial for the domain adaptation of sentiment classifiers. Indeed, our experiments have shown that linear classifiers trained with this higher-level learnt feature representation of reviews outperform the current state-of-the-art. Furthermore, we have been able to successfully perform domain adaptation on an industrial-scale dataset of 22 domains, where we significantly improve generalization over the baseline and over a similarly structured but purely supervised alternative.

Acknowledgments

This work was supported by DARPA DL Program, CRSNG, MITACS, RQCHP and SHARCNET.

References

Ben-David, S., Blitzer, J., Crammer, K., and Sokolova, P. M. (2007). Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems 20 (NIPS'07)*.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, **2**(1), 1–127. Also published as a book. Now Publishers, 2009.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2006). Greedy layer-wise training of deep networks. In *Adv. in Neural Inf. Proc. Syst. 19*, pages 153–160.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., and Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*. Oral.

Blitzer, J., McDonald, R., and Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'06)*.

Blitzer, J., Dredze, M., and Pereira, F. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of Association for Computational Linguistics (ACL'07)*.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of Internationnal Conference on Machine Learning 2008*, pages 160–167.

Dai, W., Xue, G.-R., Yang, Q., and Yu, Y. (2007). Transferring naive Bayes classifiers for text classification. In *Proc. of Assoc. for the Adv. of Art. Int. (AAAI'07)*.

Daumé III, H. and Marcu, D. (2006). Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, **26**, 101–126.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceeding of the Conference on Artificial Intelligence and Statistics*.

Goodfellow, I., Le, Q., Saxe, A., and Ng, A. (2009). Measuring invariances in deep networks. In *Advances in Neural Information Processing Systems 22*, pages 646–654.

Hinton, G. E. and Salakhutdinov, R. (2006). Reducing the dimensionality of data with neural networks. *Science*, **313**(5786), 504–507.

Hinton, G. E., Osindero, S., and Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, **18**, 1527–1554.

Jiang, J. and Zhai, C. (2007). Instance weighting for domain adaptation in nlp. In *Proceedings of Association for Computational Linguistics (ACL'07)*.

Li, S. and Zong, C. (2008). Multi-domain adaptation for sentiment classification: Using multiple classifier combining methods. In *Proc. of the Conference on Natural Language Processing and Knowledge Engineering*.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning*.

Pan, S. J., Ni, X., Sun, J.-T., Yang, Q., and Chen, Z. (2010). Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the International World Wide Web Conference (WWW'10)*.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, **2**(1-2), 1–135.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. (2007). Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the International Conference on Machine Learning*, pages 759–766.

Ranzato, M. and Szummer, M. (2008). Semi-supervised learning of compact document representations with deep networks. In *Proceedings of the International Conference on Machine Learning (ICML'08)*, pages 792–799.

Salakhutdinov, R. and Hinton, G. E. (2007). Semantic hashing. In *Proceedings of the 2007 Workshop on Information Retrieval and applications of Graphical Models (SIGIR 2007)*, Amsterdam. Elsevier.

Sindhwani, V. and Keerthi, S. S. (2006). Large scale semi-supervised linear svms. In *Proc. of the 2006 Workshop on Inf. Retrieval and Applications of Graphical Models*.

Snyder, B. and Barzilay, R. (2007). Multiple aspect ranking using the Good Grief algorithm. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Thomas, M., Pang, B., and Lee, L. (2006). Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Empirical Methods in Natural Language Processing (EMNLP'06)*.

Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural Computation*, **to appear**.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning*, pages 1096–1103.