Domain Adaptation for Image Analysis: An Unsupervised Approach Using Boltzmann Machines Trained by Perturbation

Szymon Zaręba, Marcin Kocot, and Jakub M. Tomczak(✉)

Department of Computer Science, Faculty of Computer Science and Management, WrocłAw University of Science and Technology, Wrocław, Poland
{szymon.zareba,marcin.kocot,jakub.tomczak}@pwr.edu.pl

Abstract. In this paper, we apply Restricted Boltzmann Machine and Subspace Restricted Boltzmann Machine to domain adaptation. Moreover, we train these models using the Perturb-and-MAP approach to draw approximate sample from the Gibbs distribution. We evaluate our approach on domain adaptation task between two image corpora: MNIST and Handwritten Character Recognition dataset.

Keywords: Representation transfer · Deep learning · Boltzmann machine · Perturbation method · Low-dimensional perturbations

1 Introduction

Typically, in machine learning it is assumed that training and test samples are generated from the same underlying distribution. However, very often these two samples are drawn from the training (source) distribution that is similar (*e.g.*, in the sense of the Kullback-Leibler divergence) but not the same as the test (target) distribution). Furthermore, in many real-life applications, a source sample becomes out-dated and there is a need to adapt a model to the target domain. For example, in handwriting recognition there could be a vast of data from different users but the target distribution would correspond only to a specific user that possesses her own writing style. A recognition system, however, should quickly adapt to that user using a small amount of new data. It is worth stressing out that in the considered setting the source data is no longer available and we assume that the only information preserved from the source domain is the model itself. This assumption holds true in many real-life applications where we cannot afford to store and transform a large amount of training data, *e.g.*, on mobile devices.

The problem of domain adaptation becomes an important direction in current machine learning research [12]. The increasing interest in this issue is brought by modern applications, such as, natural language processing [3], sentiment analysis [7], text categorization [16] or image categorization [24]. Moreover, the ability to learn in an unsupervised manner and adapt to new problems using small samples is a distinctive trait of the human brain. Therefore, the domain adaptation seems to be one the most important means for formulating artificial intelligence [2].

There are different approaches to the domain adaptation. One solution is to train a model using a sample from the target domain with a kind of regularizer utilizing the sample from the source domain, *e.g.*, by using Maximum Mean Discrepacy [16]. A different but similar approach aims at regularizing parameters of the model using the parameters estimated on the source domain. In [3,24] the model was regularized using ℓ_2-norm and in [14] it was further generalized and shown that from the Bayesian perspective the regularizer takes the form of the Kullback-Leibler divergence. In [1] conditions for learning a classifier for domain adaptation were provided that led to formulating an upper-bound combining source and target data. However, in many cases it remains unclear what kind of a regularization term should be proposed to adapt to new domains.

Another popular research direction is *instance weighting*. A simple modification of the risk functional for given loss function reveals that learning a model with domain adaptation relies on weighting instances from the target domain by the probability of these instances using the source distribution [12]. The manner these distributions are estimated leads to different methods. For example, one of them applies an optimization perspective and solves a minimax optimization problem [15]. Although, the instance weighting-based methods are theoretically well-motivated, they require maintaining the source sample and thus could be very time consuming.

Recently, a very promising approach is *representation adaptation* (or *representation transfer*) that utilizes hierarchical (deep) representation that contains information about different domains. First methods utilized probabilistic graphical models to combine source and target domains [5], however, these required fixed structure for the domains. More flexible approach was about augmenting feature representation for new domains [4]. Lately, *deep learning* gives the most promising results. The general idea is to apply deep neural networks that can share features among domains [6,7,26].

In this paper, we follow this line of thinking. However, we go beyond directed neural networks like auto-encoders [7] or convolutional neural networks [6,26] and utilize generative models, namely, Boltzmann Machines [11]. We aim at verifying whether adapting to new domain is possible using Restricted Boltzmann Machine [20] and its recently proposed modification, Subspace Restricted Boltzmann Machine [23], without any additional techniques for domain adaptation. In other words, we want to quantify to which extent these models provide representation transferable to new domains. Additionally, we apply new learning procedure for the two mentioned Restricted Boltzmann Machines using the Perturb-and-MAP (PM) approach [18]. The basic idea of the PM approach is to perturb the parameters of the model, and then, starting from the training data, to find the local optima of the energy function using and optimization method. This procedure produces approximate samples from the Gibbs distribution that can be further used to approximate the gradient.

The contribution of the paper is threefold:

- We apply Restricted Boltzmann Machine and Subspace Restricted Boltzmann machine to unsupervised domain adaptation task.

- We utilize the Perturb-and-MAP approach to learning Subspace Restricted Boltzmann Machine.
- We propose an evaluation metric for the domain adaptation in the fully unsupervised setting (*i.e.*, there are no labels available from source and target domains).

The paper is organized as follows. First, in Sect. 2 we formulate the problem of domain adaptation. Next, we outline Boltzmann machines used in the paper in Sect. 3. Further, we describe a new learning schema for the Subspace Restricted Boltzmann Machine basing on the Perturb-and-MAP approach in Sect. 4. The presented approaches are evaluated in Sect. 5 using MNIST dataset [13] and Handwritten Characted Recognition dataset [25]. At the end, the conclusions are drawn in Sect. 6.

2 Unsupervised Domain Adaptation Problem

Let us define a data space $\mathcal{X}$ and a distribution $P(\mathbf{x})$ where $\mathbf{x} \in \mathcal{X}$. The objective of (unsupervised) machine learning is to learn a hypothesis (a model) $h(\mathbf{x})$ basing on N training data $\{x_n\}_{n=1}^{N}$. We define a *domain* as a pair $\mathbb{D} = \big(\mathcal{X}, P(\mathbf{x})\big)$. Further, we distinguish a *source domain* $\mathbb{D}_S$ and a *target domain* $\mathbb{D}_T$. We formulate the problem of *domain adaptation* as follows:

Given a source domain $\mathbb{D}_S$ *and a target domain* $\mathbb{D}_T$ *such that* $P_S(\mathbf{x}) \neq P_T(\mathbf{x})$, *and a model* $h_S(\mathbf{x})$ *trained using examples from* $\mathbb{D}_S$, *the* domain adaptation *aims to learn* $h(\mathbf{x})$ *using a sample from* $\mathbb{D}_T$ *that transfers knowledge from* $h_S(\mathbf{x})$ *and makes as small errors on both* $\mathbb{D}_S$ *and* $\mathbb{D}_T$ *as possible.*

There are different definitions of the domain adaptation (see, *e.g.*, [12,16]), however, in this paper we focus on a fully unsupervised problem statement. Moreover, we would like to emphasize that in our definition we have access to the source domain through the model $h_S(\mathbf{x})$ only. This forbids an application of a technique that takes advantage of examples from the source domain (see, *e.g.*, [16]). Therefore, a beneficial approach would be to transfer representation from source to target domain, *e.g.*, by applying deep learning [26].

3 Boltzmann Machines

A general Boltzmann machine (BM) is defined through a *Gibbs distribution* and an *energy function* that describe relationships among random variables.

Restricted Boltzmann Machine. The binary Restricted Boltzmann Machine (RBM) is a bipartite BM that defines the joint distribution over binary visible and hidden units [21], where $\mathbf{x} \in \{0,1\}^D$ are the visibles and $\mathbf{h} \in \{0,1\}^M$ are the hiddens. The relationships among variables are specified through the energy function:

$$E(\mathbf{x}, \mathbf{h}|\Theta) = -\mathbf{x}^\top \mathbf{W}\mathbf{h} - \mathbf{b}^\top \mathbf{x} - \mathbf{c}^\top \mathbf{h}, \tag{1}$$

where $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$ is a set of parameters, $\mathbf{W} \in \mathbb{R}^{D\times M}$, $\mathbf{b} \in \mathbb{R}^D$, and $\mathbf{c} \in \mathbb{R}^M$.

We can train RBM using the maximum likelihood approach that seeks the maximum of the averaged log-likelihood. However, since the partition function (*i.e.*, normalizing constant) in the Gibbs distribution requires summing over exponential number of configurations, an application of a gradient-based optimization methods is troublesome. Therefore, RBM are trained using some kind of gradient approximations like in the *contrastive divergence* (CD) algorithm that applies T steps of the block Gibbs sampling [10].

Subspace Restricted Boltzmann Machine. The sRBM introduces third-order multiplicative interactions of one visible x_i and two types of hidden binary units, a gate unit g_j and a subspace unit s_{jk}. Each gate unit is associated with a group of subspace hidden units. The energy function of a joint configuration is defined as follows [23]:

$$E(\mathbf{x}, \mathbf{h}, \mathbf{S}|\boldsymbol{\theta}) = -\sum_{i=1}^{D}\sum_{j=1}^{M}\sum_{k=1}^{K} V_{ijk} x_i g_j S_{jk} - \sum_{i=1}^{D} b_i x_i - \sum_{j=1}^{M} c_j g_j - \sum_{j=1}^{M} g_j \sum_{k=1}^{K} D_{jk} S_{jk}. \quad (2)$$

where $\mathbf{x} \in \{0,1\}^D$ denotes a vector of visible variables, $\mathbf{g} \in \{0,1\}^M$ is a vector of gate units, $\mathbf{S} \in \{0,1\}^{M\times K}$ is a matrix of subspace units, the parameters are $\boldsymbol{\theta} = \{\mathbf{V}, \mathbf{b}, \mathbf{c}, \mathbf{D}\}$, $\mathbf{V} \in \mathbb{R}^{D\times M\times K}$, $\mathbf{b} \in \mathbb{R}^D$, $\mathbf{c} \in \mathbb{R}^M$, and $\mathbf{D} \in \mathbb{R}^{M\times K}$.

The sRBM can be seen as a mixture of many simple RBMs, where gate units allow to activate or deactivate subsets of hidden units (subspace units). We believe that these third-order relationships allow to better reflect domain adaptation by maintaining information about domains in different small RBMs.

4 Perturb-and-MAP Learning Algorithm

The idea of the *Perturb-and-MAP* (PM) approach is about first adding i.i.d. Gumbel perturbations to the energy function and next finding the MAP configuration that is a sample from the original Gibbs distribution [18]. Nevertheless, since the domain of visibles and hiddens in RBM and sRBM grows exponentially with the number of variables, it is troublesome to find the MAP assignment of the perturbed energy efficiently. Therefore, first order (low-dimensional) Gumbel perturbations are often employed to obtain an approximate sample [8,17,22]. Then the joint perturbation is fully decomposable into a sum of perturbations and it corresponds to perturbing unary potentials (*i.e.*, biases) only [8,9,18]. Further, we denote Gumbel perturbation for value z by $\gamma(z)$.

The manner of how the MAP configurations are found is crucial for learning process. Since finding MAP solutions is run for each example in a mini-batch, the method cannot be computationally complex. In [19] it was proposed to obtain samples from the RBM by first perturbing the unary potentials, and further, starting from a data point, applying block coordinate descent to optimize the energy function (Perturb-and-Descent, PD). The procedure for sampling from RBM is presented in Algorithm 1.[1]

[1] $\mathbb{I}[\cdot]$ denotes the indicator function, and $\odot$ is the element-wise multiplication.

Algorithm 1. Perturb-and-Descent for RBM

```
Input  : $\mathbf{x}^{(0)}$: training datum, $T$: number of optimization steps
Output: $\{\hat{\mathbf{x}}^{(T)}, \mathbf{h}^{(T)}\}$: approximate MAP solutions of the perturbed energy
for $i = 1, \ldots, D,\ j = 1, \ldots, M$ do
    $\tilde{b}_i = b_i + \gamma(x_i = 1) - \gamma(x_i = 0)$;
    $\tilde{c}_j = c_j + \gamma(h_j = 1) - \gamma(h_j = 0)$;
end
for $t = 1, \ldots, T$ do
    $\mathbf{x}^{(t)} = \mathbb{I}[\mathbf{W}\mathbf{h}^{(t-1)} + \tilde{\mathbf{b}} > \mathbf{0}]$;
    $\mathbf{h}^{(t)} = \mathbb{I}[\mathbf{W}^\top \mathbf{x}^{(t)} + \tilde{\mathbf{c}} > \mathbf{0}]$;
end
return $\{\hat{\mathbf{x}}^{(T)}, \mathbf{h}^{(T)}\}$;
```

In the context of the sRBM the PD algorithm takes the similar form and is presented in Algorithm 2.

Algorithm 2. Perturb-and-Descent for sRBM

```
Input  : $\mathbf{x}^{(0)}$: training datum, $T$: number of optimization steps
Output: $\{\hat{\mathbf{x}}^{(T)}, \mathbf{g}^{(T)}, \mathbf{S}^{(T)}\}$: approximate MAP solutions of the perturbed
        energy
for $i = 1, \ldots, D,\ j = 1, \ldots, M$ and $k = 1, \ldots, K$ do
    $\tilde{b}_i = b_i + \gamma(x_i = 1) - \gamma(x_i = 0)$;
    $\tilde{c}_j = c_j + \gamma(g_j = 1) - \gamma(g_j = 0)$;
    $\tilde{D}_{jk} = D_{jk} + \gamma(S_{jk} = 1) - \gamma(S_{jk} = 0)$;
end
for $t = 1, \ldots, T$ do
    $\mathbf{x}^{(t)} = \mathbb{I}[\sum_{j=1}^{M} \sum_{k=1}^{K} V_{\cdot jk} g_j^{(t-1)} S_{jk}^{(t-1)} + \tilde{\mathbf{b}} > \mathbf{0}]$;
    $\mathbf{g}^{(t)} = \mathbb{I}[\sum_{i=1}^{D} \sum_{k=1}^{K} V_{i \cdot k} x_i^{(t)} S_{jk}^{(t-1)} + \tilde{\mathbf{c}} + \sum_{k=1}^{K} \tilde{D}_{\cdot k} \odot S_{\cdot k}^{(t-1)} > \mathbf{0}]$;
    $\mathbf{S}^{(t)} = \mathbb{I}[\sum_{i=1}^{D} \sum_{j=1}^{M} V_{ij\cdot} x_i^{(t)} g_j^{(t)} + \sum_{j=1}^{M} g_j^{(t)} \tilde{D}_{j\cdot} \odot S_{j\cdot}^{(t-1)} > \mathbf{0}]$;
end
return $\{\hat{\mathbf{x}}^{(T)}, \mathbf{g}^{(T)}, \mathbf{S}^{(T)}\}$;
```

5 Experiments

Data and Training Details. We use two datasets in the experiment: Handwritten Character Recognition (HCR) [25], and MNIST [13]. MNIST contains handwritten digits of size 28×28, and HCR consists of images of digits and letters[2]. We split HCR into 24,000 training images, 8,134 test images, and remaining 8000 images formulate validation set. MNIST is divided into 50,000 training images, 10,000 validation images, and 10,000 test images (Fig. 1).

[2] The original HCR is scaled to fit MNIST images.

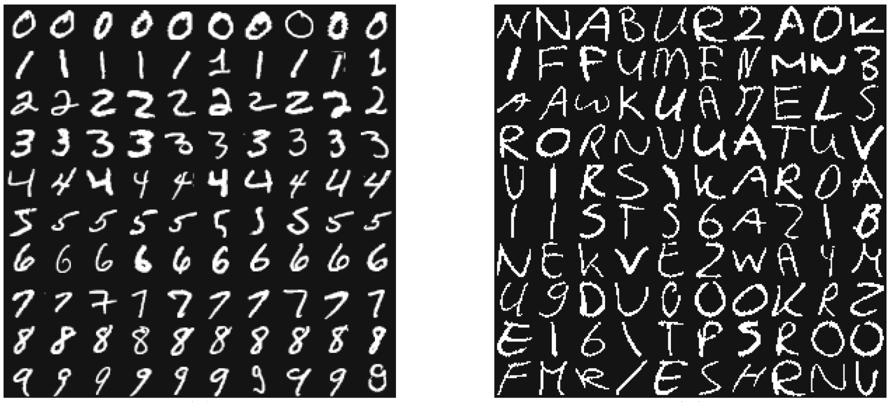

(a) (b)

Fig. 1. Exemplary images from: (a) MNIST, and (b) HCR.

In the experiment we consider RBM with 500 hidden units and sRBM with 100 subspace units and 5 gate units to have comparable number of parameters. We consider learning rate in $\{0.1, 0.01\}$, weight decay in $\{0, 10^{-6}, 10^{-5}, 10^{-4}\}$, momentum equal 0.9 and optimization steps (or Gibbs sampler steps) in $\{1, 5, 10\}$. The hyperparameters are chosen using the validation set. Additionally, we apply early stopping with 30 look ahead steps. We train a model on first domain and once the training process converges, we switch the domain and proceed learning.

Evaluation Methodology. Our basic evaluation criterion is the negative *reconstruction error* (*cross-entropy*):

$$\varepsilon(\mathcal{D}) = \sum_{n=1}^{N}\sum_{d=1}^{D}\big(x_{n,d}\log\tilde{x}_{n,d} + (1-x_{n,d})\log(1-\tilde{x}_{n,d})\big), \tag{3}$$

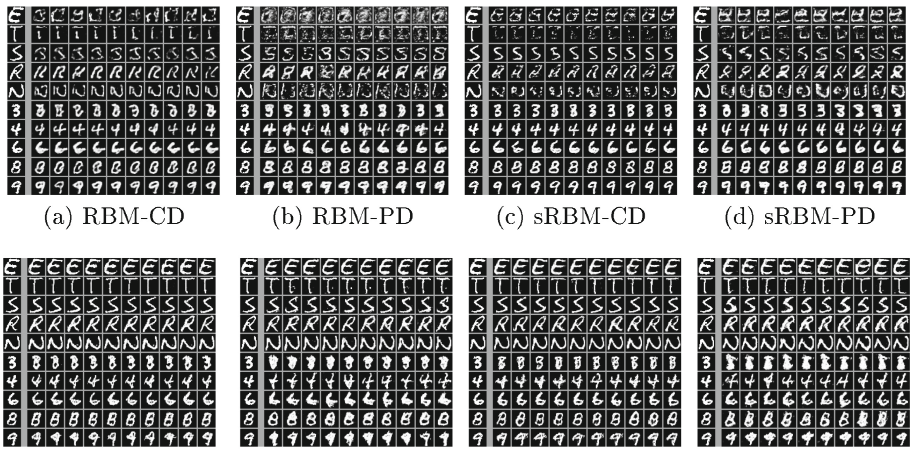
where $\mathcal{D}$ is a dataset, $\mathbf{x}_n$ is an original image and $\tilde{\mathbf{x}}_n$ is its reconstruction. We measure the reconstruction error at different timestamps during learning: (i) after learning the first domain (*SWAP*), and (ii) at the end of training on the second domain (*END*). In order to measure the quality of a model on the domain adaptation task we define a domain adaptation accuracy as follows. The reconstruction error of a model that was first trained on $\mathcal{D}_i$ and further learned on $\mathcal{D}_j$ and eventually evaluated on $\mathcal{D}_i$ is denoted by $\varepsilon_{i\to j}(\mathcal{D}_i)$. In order to be invariant to specific domain error, we introduce a *baseline* model, multiple Bernoulli distribution with pixel probabilities calculated on the training data. The reconstruction error on dataset $\mathcal{D}_i$ for baseline is denoted by $\varepsilon_b(\mathcal{D}_i)$. The domain adaptation accuracy is then calculated as follows:

$$\alpha(\mathcal{D}_i \to \mathcal{D}_j) = \frac{1}{2}\Big(\frac{\varepsilon_b(\mathcal{D}_i) - \varepsilon_{i\to j}(\mathcal{D}_i)}{\varepsilon_b(\mathcal{D}_i)} + \frac{\varepsilon_b(\mathcal{D}_j) - \varepsilon_{i\to j}(\mathcal{D}_j)}{\varepsilon_b(\mathcal{D}_j)}\Big). \tag{4}$$

Table 1. Reconstruction errors and domain adaptation accuracy.

MNIST → HCR	SWAP		END		Domain adaptation accuracy
	HCR	MNIST	HCR	MNIST	
RBM-CD	159.1 ± 0.9	61.4 ± 0.2	32.9 ± 0.4	73.4 ± 0.4	$76.6\,\% \pm 0.1$
RBM-PD	163.4 ± 2.0	41.8 ± 0.1	40.0 ± 0.1	45.6 ± 0.6	$82.0\,\% \pm 0.1$
sRBM-CD	197.7 ± 8.9	42.2 ± 0.4	50.5 ± 0.6	55.3 ± 1.1	$77.9\,\% \pm 0.3$
sRBM-PD	202.5 ± 4.0	47.3 ± 0.1	49.2 ± 0.8	55.3 ± 0.7	$78.1\,\% \pm 0.1$
HCR → MNIST	SWAP		END		Domain adaptation accuracy
	HCR	MNIST	HCR	MNIST	
RBM-CD	29.6 ± 0.4	71.2 ± 0.2	109.6 ± 6.3	62.4 ± 0.1	$65.9\,\% \pm 0.1$
RBM-PD	36.6 ± 0.4	42.6 ± 1.0	110.6 ± 2.4	44.3 ± 0.3	$70.1\,\% \pm 0.1$
sRBM-CD	43.6 ± 0.2	52.0 ± 0.7	152 ± 10.2	43.1 ± 0.7	$63.1\,\% \pm 1.7$
sRBM-PD	48.2 ± 0.3	55.3 ± 0.7	130.2 ± 9.9	47.3 ± 0.7	$66.0\,\% \pm 0.1$

Results. The reconstruction errors and the domain adaptation accuracies are presented in Table 1.[3] The baseline achieves the following reconstruction errors: (i) HCR: 288.2, (ii) MNIST: 206.8. We present sampling capabilities of the considered methods in Fig. 2.



(a) RBM-CD (b) RBM-PD (c) sRBM-CD (d) sRBM-PD

(e) RBM-CD (f) RBM-PD (g) sRBM-CD (h) sRBM-PD

Fig. 2. Sampling using RBM and sRBM: (*top row*) HCR→MNIST, (*bottom row*) MNIST→HCR. In each figure the most left column contains real images.

[3] RBM and sRBM trained with contrastive divergence (CD) and PD are denoted by RBM-CD and sRBM-CD, and RBM-PD and sRBM-PD, respectively.

6 Discussion and Conclusion

In general, we notice that sRBM performs similarly to RBM, however, RBM has tendency to overfit to the first domain while the sRBM has problems with learning HCR (see Table 1). However, application of the PD training seems to help RBM to adapt to new domain by obtaining the best domain accuracy. This effect is less evident for sRBM. We hypothesize that sampling using PD approach allows to obtain more diverse examples during training and that is why the model does not drastically overfit to the first domain. We believe that similar result could be obtained for sRBM, however, in order to compare RBM with sRBM we chose the number of hidden units to be the same in both models but 5 gate and 100 subspace units could be too small amount to properly reconstruct images. Next, a closer inspection of the sampling capabilities of the considered approaches reveals that sampling with PD indeed results in more diverse sample. It seems that application of CD leads to copying training images while PD gives a model that generates more various images, *e.g.*, see letter S and digit 6 in Fig. 2e and f. Interestingly, sRBM trained with CD and PD allows to obtain more diverse examples than the ones from RBM-CD. Concluding, we notice that the learning procedure clearly matters in the domain adaptation using RBM, nevertheless, the obtained reconstructions are still imperfect. An open question is whether application of deeper BM is enough to handle the domain adaptation problem or other optimization techniques are needed. We leave investigating this issue for further research.

Acknowledgments. The work presented in the paper is partially co-financed by the Ministry of Science and Higher Education in Poland.

References

1. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Mach. Learn. **79**(1–2), 151–175 (2010)
2. Bengio, Y.: Learning deep architectures for AI. Found. Trends Mach. Learn. **2**(1), 1–127 (2009)
3. Chelba, C., Acero, A.: Adaptation of maximum entropy capitalizer: little data can help a lot. Comput. Speech Lang. **20**(4), 382–399 (2006)
4. Daumé III., H., Frustratingly easy domain adaptation. arXiv preprint arXiv: 0907.1815 (2009)
5. Daume III, H., Marcu, D.: Domain adaptation for statistical classifiers. J. Artif. Intell. Res. **26**, 101–126 (2006)
6. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: Proceedings of The 32nd International Conference on Machine Learning, pp. 1180–1189 (2015)
7. Glorot, X., Bordes, A., Bengio, Y.: Domain adaptation for large-scale sentiment classification: a deep learning approach. In: Proceedings of the 28th International Conference on Machine Learning (ICML 2011), pp. 513–520 (2011
8. Hazan, T., Jaakkola, T.: On the partition function and random maximum a-posteriori perturbations. In: ICML (2012)

9. Hazan, T., Maji, S., Jaakkola, T.: On sampling from the gibbs distribution with random maximum a-posteriori perturbations. In: NIPS, pp. 1268–1276 (2013)
10. Hinton, G.: Training products of experts by minimizing contrastive divergence. Neural Comput. **14**(8), 1771–1800 (2002)
11. Hinton, G.E., Sejnowski, T.J.: Learning and relearning in Boltzmann machines. Parallel Distrib. Process. Explor. Microstruct. Cogn. **1**, 282–317 (1986)
12. Jiang, J.: A literature survey on domain adaptation of statistical classifiers. Technical Report http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/da_survey.pdf
13. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
14. Li, X., Bilmes, J.: A bayesian divergence prior for classiffier adaptation. In: International Conference on Artificial Intelligence and Statistics, pp. 275–282 (2007)
15. Liu, A., Ziebart, B.: Robust classification under sample selection bias. In: Advances in Neural Information Processing Systems, pp. 37–45 (2014)
16. Long, M., Wang, J., Ding, G., Pan, S.J., Yu, P.S.: Adaptation regularization: a general framework for transfer learning. IEEE Trans. Knowl. Data Eng. **26**(5), 1076–1089 (2014)
17. Orabona, F., Hazan, T., Sarwate, A., Jaakkola, T.: On measure concentration of random maximum a-posteriori perturbations. In: ICML, pp. 432–440 (2014)
18. Papandreou, G., Yuille, A., Perturb-and-map random fields: using discrete optimization to learn and sample from energy models. In: ICCV, pp. 193–200 (2011)
19. Ravanbakhsh, S., Greiner, R., Frey, B.: Machine, Training Restricted Boltzmann by Perturbation. arXiv preprint arXiv: 1405.1436 (2014)
20. Sejnowski, T.: Higher-order Boltzmann machines. In: AIP Conference Proceedings, vol. 151, pp. 398–403 (1986)
21. Smolensky, P.: Information processing in dynamical systems: foundations of harmony theory. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1, pp. 194–281. MIT Press (1986)
22. Tomczak, J.M.: On some properties of the low-dimensional Gumbel perturbations in the Perturb-and-MAP model. Stat. Probab. Lett. **115**, 8–15 (2016). http://dx.doi.org/10.1016/j.spl.2016.03.019
23. Tomczak, J.M., Gonczarek, A.: Learning Invariant Features Using Subspace Restricted Boltzmann Machine. Neural Process. Lett. 1–10 (2016). doi:10.1007/s11063-016-9519-9
24. Tommasi, T., Orabona, F., Caputo, B.: Learning categories from few examples with multi model knowledge transfer. IEEE Trans. Pattern Anal. Mach. Intell. **36**(5), 928–941 (2014)
25. Van der Maaten, L.: A new benchmark dataset for handwritten character recognition, pp. 2–5. Tilburg University (2009)
26. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems, pp. 3320–3328 (2014)