Journal of Machine Learning Research 13 (2012) 643-669 Submitted 6/11; Revised 11/11; Published 3/12

Learning Algorithms for the Classification Restricted Boltzmann Machine

Hugo Larochelle HUGO.LAROCHELLE@USHERBROOKE.CA
Université de Sherbrooke
2500, boul. de l'Université
Sherbrooke, Québec, Canada, J1K 2R1

Michael Mandel MANDELM@IRO.UMONTREAL.CA
Razvan Pascanu PASCANUR@IRO.UMONTREAL.CA
Yoshua Bengio BENGIOY@IRO.UMONTREAL.CA
Département d'informatique et de recherche opérationnelle
Université de Montréal
2920, chemin de la Tour
Montréal, Québec, Canada, H3T 1J8

Editor: Daniel Lee

Abstract

Recent developments have demonstrated the capacity of restricted Boltzmann machines (RBM) to be powerful generative models, able to extract useful features from input data or construct deep artificial neural networks. In such settings, the RBM only yields a preprocessing or an initialization for some other model, instead of acting as a complete supervised model in its own right. In this paper, we argue that RBMs can provide a self-contained framework for developing competitive classifiers. We study the Classification RBM (ClassRBM), a variant on the RBM adapted to the classification setting. We study different strategies for training the ClassRBM and show that competitive classification performances can be reached when appropriately combining discriminative and generative training objectives. Since training according to the generative objective requires the computation of a generally intractable gradient, we also compare different approaches to estimating this gradient and address the issue of obtaining such a gradient for problems with very high dimensional inputs. Finally, we describe how to adapt the ClassRBM to two special cases of classification problems, namely semi-supervised and multitask learning.

Keywords: restricted Boltzmann machine, classification, discriminative learning, generative learning

1. Introduction

The restricted Boltzmann machine (RBM) is a probabilistic model that uses a layer of hidden binary variables or units to model the distribution of a visible layer of variables. It has been successfully applied to problems involving high dimensional data such as images (Hinton et al., 2006; Larochelle et al., 2007) and text (Welling et al., 2005; Salakhutdinov and Hinton, 2007; Mnih and Hinton, 2007). In this context, two approaches are usually followed. First, an RBM is trained in an unsupervised manner to model the distribution of the inputs (possibly more than one RBM could be trained, stacking them on top of each other (Hinton et al., 2006)). Then, the RBM is used in one of two ways: either its hidden layer is used to preprocess the input data by replacing it with the representation given by the hidden layer, or the parameters of the RBM are used to initialize a feedforward neural network. In both cases, the RBM is paired with some other learning algorithm (the classifier using the preprocessed inputs or the neural network) to solve the supervised learning problem at hand. This approach unfortunately requires one to tune both sets of hyper-parameters (those of the RBM and of the other learning algorithm) at the same time. Moreover, since the RBM is trained in an unsupervised manner, it is blind to the nature of the supervised task that needs to be solved and provides no guarantees that the information extracted by its hidden layer will be useful.

In this paper, we argue that RBMs can provide a self-contained and competitive framework for solving supervised learning problems. Based on the Classification Restricted Boltzmann Machine (ClassRBM), the proposed approach and learning algorithms address both aforementioned issues. Indeed, by relying only on the RBM, the number of hyper-parameters that one needs to tune will be relatively smaller, and by modelling the *joint* distribution of the input and target, the ClassRBM will be encouraged to allocate some of its capacity at modelling their relationship as well as the relationships between the input variables. Using experiments on character recognition and text classification problems, we show that the classification performance that the ClassRBM can obtain is competitive with respect to other "black box" classifiers such as standard neural networks and Support Vector Machines (SVM). We compare different training strategies for the ClassRBM, which rely on discriminative and/or generative learning objectives. As we will see, the best approach tends to be an appropriately tuned combination of both learning objectives. Moreover, since the generative learning objective doesn't allow for the exact computation of the gradient with respect to the ClassRBM's parameters, we compare the use of different approximations of that gradient. We also address the issue of generative learning on very high dimensional inputs and propose an approach to reduce the computational cost per example of training. Finally, we describe how the ClassRBM can be used to tackle semi-supervised and multitask learning problems.

2. Classification Restricted Boltzmann Machines

The Classification Restricted Boltzmann Machine (ClassRBM) (Hinton et al., 2006) models the joint distribution of an input $\mathbf{x} = (x_1, \ldots, x_D)$ and target class $y \in \{1, \ldots, C\}$ using a hidden layer of binary stochastic units $\mathbf{h} = (h_1, \ldots, h_H)$. This is done by first defining an energy function

$$E(y, \mathbf{x}, \mathbf{h}) = -\mathbf{h}^T \mathbf{W} \mathbf{x} - \mathbf{b}^T \mathbf{x} - \mathbf{c}^T \mathbf{h} - \mathbf{d}^T \mathbf{e}_y - \mathbf{h}^T \mathbf{U} \mathbf{e}_y$$

with parameters $\Theta = (\mathbf{W}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{U})$ and where $\mathbf{e}_y = (1_{i=y})_{i=1}^C$ is the "one out of C" representation of y. From the energy function, we assign probabilities to values of y, $\mathbf{x}$ and $\mathbf{h}$ as follows:

$$p(y, \mathbf{x}, \mathbf{h}) = \frac{\exp(-E(y, \mathbf{x}, \mathbf{h}))}{Z} \tag{1}$$

where Z is a normalization constant (also called partition function) which ensures that Equation 1 is a valid probability distribution. We will assume that the elements of $\mathbf{x}$ are binary, but extensions to real-valued units on bounded or unbounded intervals are straightforward (Welling et al., 2005). An illustration of the ClassRBM is given in Figure 1.

Unfortunately, computing $p(y, \mathbf{x}, \mathbf{h})$ or $p(y, \mathbf{x})$ is typically intractable. However, it is possible to sample from the ClassRBM, using Gibbs sampling, that is, alternating between sampling a value for the hidden layer given the current value of the visible layer (made of variables $\mathbf{x}$ and the $\mathbf{e}_y$

representation of y), and vice versa. All the required conditional distributions are very simple. When conditioning on the visible layer, we have

$$p(\mathbf{h}|y,\mathbf{x}) = \prod_j p(h_j|y,\mathbf{x}), \text{ with } p(h_j = 1|y,\mathbf{x}) = \text{sigm}(c_j + U_{jy} + \sum_i W_{ji}x_i)$$

where $\text{sigm}(a) = 1/(1+\exp(-a))$ is the logistic sigmoid function. When conditioning on the hidden layer, we have

$$\begin{aligned} p(\mathbf{x}|\mathbf{h}) &= \prod_i p(x_i|\mathbf{h}), \text{ with } p(x_i = 1|\mathbf{h}) = \text{sigm}(b_i + \sum_j W_{ji}h_j)\ , \\ p(y|\mathbf{h}) &= \frac{\exp(d_y + \sum_j U_{jy}h_j)}{\sum_{y^*} \exp(d_{y^*} + \sum_j U_{jy^*}h_j)}\ . \end{aligned}$$

It is also possible to compute $p(y|\mathbf{x})$ exactly and hence perform classification. Indeed, noticing that

$$\begin{aligned} &\sum_{h_1\in\{0,1\}} \cdots \sum_{h_H\in\{0,1\}} \exp(\mathbf{h}^T\mathbf{W}\mathbf{x} + \mathbf{b}^T\mathbf{x} + \mathbf{c}^T\mathbf{h} + \mathbf{d}^T\mathbf{e}_y + \mathbf{h}^T\mathbf{U}\mathbf{e}_y) \\ &= \exp(d_y) \sum_{h_1\in\{0,1\}} \exp(h_1(c_1 + U_{1y} + \sum_i W_{1i}x_i)) \cdots \sum_{h_H\in\{0,1\}} \exp(h_H(c_H + U_{Hy} + \sum_i W_{Hi}x_i)) \\ &= \exp(d_y) \left(1 + \exp(c_1 + U_{1y} + \sum_i W_{1i}x_i)\right) \ldots \left(1 + \exp(c_n + U_{Hy} + \sum_i W_{ni}x_i)\right) \\ &= \exp(d_y + \sum_j \log(1 + \exp(c_j + U_{jy} + \sum_i W_{ji}x_i))) \\ &= \exp(d_y + \sum_j \text{softplus}(c_j + U_{jy} + \sum_i W_{ji}x_i)) \end{aligned}$$

where $\text{softplus}(a) = \log(1+\exp(a))$, then we can write

$$\begin{aligned} p(y|\mathbf{x}) &= \frac{\sum_{h_1\in\{0,1\}} \cdots \sum_{h_H\in\{0,1\}} \exp(\mathbf{h}^T\mathbf{W}\mathbf{x} + \mathbf{b}^T\mathbf{x} + \mathbf{c}^T\mathbf{h} + \mathbf{d}^T\mathbf{e}_y + \mathbf{h}^T\mathbf{U}\mathbf{e}_y)}{\sum_{y^*\in\{1,\ldots,C\}} \sum_{h_1\in\{0,1\}} \cdots \sum_{h_H\in\{0,1\}} \exp(\mathbf{h}^T\mathbf{W}\mathbf{x} + \mathbf{b}^T\mathbf{x} + \mathbf{c}^T\mathbf{h} + \mathbf{d}^T\mathbf{e}_{y^*} + \mathbf{h}^T\mathbf{U}\mathbf{e}_{y^*})} \\ &= \frac{\exp(d_y + \sum_j \text{softplus}(c_j + U_{jy} + \sum_i W_{ji}x_i))}{\sum_{y^*\in\{1,\ldots,C\}} \exp(d_y^* + \sum_j \text{softplus}(c_j + U_{jy^*} + \sum_i W_{ji}x_i))} \\ &= \frac{\exp(-F(y,\mathbf{x}))}{\sum_{y^*\in\{1,\ldots,C\}} \exp(-F(y,\mathbf{x}))}\ . \end{aligned} \tag{2}$$

where $F(y,\mathbf{x})$ is referred to as the free energy. Precomputing the terms $c_j + \sum_i W_{ji}x_i$ and reusing them when computing $\text{softplus}(c_j + U_{jy^*} + \sum_i W_{ji}x_i)$ for all classes y^* yields a procedure for computing this conditional distribution in time $O(HD+HC)$.

One way of interpreting Equation 2 is that, when assigning probabilities to a particular class y for some input $\mathbf{x}$, the ClassRBM looks at how well the input $\mathbf{x}$ fits or aligns with the different filters associated with the rows $\mathbf{W}_{j\cdot}$ of $\mathbf{W}$. These filters are shared across the different classes, but different classes will make comparisons with different filters by controlling the class-dependent biases U_{jy} in the $\text{softplus}(c_j + U_{jy} + \sum_i W_{ji}x_i)$ terms. Notice also that two similar classes could share some of the filters in $\mathbf{W}$, that is, both could simultaneously have large positive values of U_{jy} for some of the rows $\mathbf{W}_{j\cdot}$.

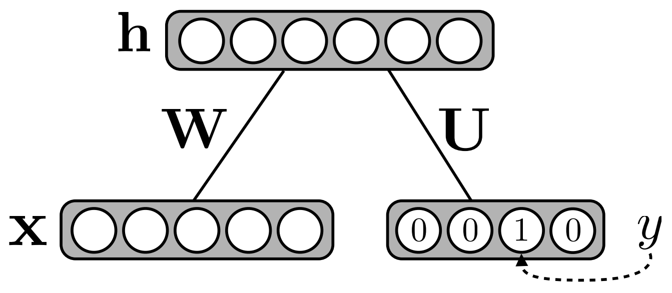


Figure 1: Illustration of a Classification Restricted Boltzmann Machine

3. Training Objectives

In order to train a ClassRBM to solve a particular classification problem, we can simply define an objective to minimize for all examples in the training set $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_t, y_t)\}$. The next sections describe the different training objectives which will be considered here, starting with the one most commonly used, that is, the generative training objective.

3.1 Generative Training Objective

Given that we have a model which defines a value for the joint probability $p(y, \mathbf{x})$, a natural choice for a training objective is the generative training objective

$$\mathcal{L}_{\text{gen}}(\mathcal{D}_{\text{train}}) = -\sum_{t=1}^{|\mathcal{D}_{\text{train}}|} \log p(y_t, \mathbf{x}_t) \ . \tag{3}$$

This is the most popular training objective for RBMs, for which a lot of effort has been put to obtain better estimates for its gradient (Hinton, 2002; Tieleman, 2008; Tieleman and Hinton, 2009). Indeed, as mentioned previously, computing $p(y_t, \mathbf{x}_t)$ for some example $(\mathbf{x}_t, y_t)$ is generally intractable, as is computing $\log p(y_t, \mathbf{x}_t)$ and its gradient with respect to any ClassRBM parameter θ

$$\frac{\partial \log p(y_t, \mathbf{x}_t)}{\partial \theta} = -\mathbb{E}_{\mathbf{h}|y_t,\mathbf{x}_t}\left[\frac{\partial}{\partial \theta} E(y_t, \mathbf{x}_t, \mathbf{h})\right] + \mathbb{E}_{y,\mathbf{x},\mathbf{h}}\left[\frac{\partial}{\partial \theta} E(y, \mathbf{x}, \mathbf{h})\right] .$$

Specifically, though the first expectation is tractable, the second is not. Different approaches have been proposed to estimate this second expectation. One which is known to work well in practice is the contrastive divergence estimator (Hinton, 2002). This approximation replaces the expectation by a point estimate at a sample generated after a limited number of Gibbs sampling steps, with the sampler's initial state for the visible variables initialized at the training example $(\mathbf{x}_t, y_t)$. A single Gibbs sampling iteration is often used and is usually found to be sufficient to learn a meaningful representation of the data. Then, this gradient estimate can be used in a stochastic gradient descent procedure for training. A pseudocode of the procedure is given by Algorithm 1, where the learning rate is controlled by λ. We will consider this procedure for now and postpone the consideration of other estimates of the generative gradient to Section 7.3.

Algorithm 1 Generative training update of the ClassRBM using Contrastive Divergence

```
**Input:** training pair $(y_t, \mathbf{x}_t)$ and learning rate $\lambda$
# Notation: $a \leftarrow b$ means $a$ is set to value $b$
#           $a \sim p$ means $a$ is sampled from $p$

# Positive phase
$y^0 \leftarrow y_t$, $\mathbf{x}^0 \leftarrow \mathbf{x}_t$, $\widehat{\mathbf{h}}^0 \leftarrow \mathrm{sigm}(\mathbf{c} + W\mathbf{x}^0 + \mathbf{U}\mathbf{e}_{y^0})$

# Negative phase
$\mathbf{h}^0 \sim p(\mathbf{h}|y^0, \mathbf{x}^0)$, $y^1 \sim p(y|\mathbf{h}^0)$, $\mathbf{x}^1 \sim p(\mathbf{x}|\mathbf{h}^0)$
$\widehat{\mathbf{h}}^1 \leftarrow \mathrm{sigm}(\mathbf{c} + W\mathbf{x}^1 + \mathbf{U}\mathbf{e}_{y^1})$

# Update
for $\theta \in \Theta$ do
    $\theta \leftarrow \theta - \lambda \left( \frac{\partial}{\partial\theta} E(y^0, \mathbf{x}^0, \widehat{\mathbf{h}}^0) - \frac{\partial}{\partial\theta} E(y^1, \mathbf{x}^1, \widehat{\mathbf{h}}^1) \right)$
end for
```

4. Discriminative Training Objective

The generative training objective can be decomposed as follows:

$$\mathcal{L}_{\mathrm{gen}}(\mathcal{D}_{\mathrm{train}}) = -\sum_{t=1}^{|\mathcal{D}_{\mathrm{train}}|} (\log p(y_t|\mathbf{x}_t) + \log p(\mathbf{x}_t)) = -\sum_{t=1}^{|\mathcal{D}_{\mathrm{train}}|} \log p(y_t|\mathbf{x}_t) - \sum_{t=1}^{|\mathcal{D}_{\mathrm{train}}|} \log p(\mathbf{x}_t) \tag{4}$$

hinting that the ClassRBM will dedicate some of its capacity at modelling the marginal distribution of the input only. Since we are in a supervised learning setting and are only interested in obtaining a good prediction of the target given the input, it might be more appropriate to ignore this unsupervised part of the generative objective and focus on the supervised part.

Doing so is referred to as discriminative training, where the following training objective is used:

$$\mathcal{L}_{\mathrm{disc}}(\mathcal{D}_{\mathrm{train}}) = -\sum_{t=1}^{|\mathcal{D}_{\mathrm{train}}|} \log p(y_t|\mathbf{x}_t) \ . \tag{5}$$

This objective is also similar to the one used by feedforward neural networks whose outputs can be interpreted as an estimate of $p(y|\mathbf{x})$. Moreover, just like neural networks, ClassRBMs trained this way are universal approximators for distributions $p(y|\mathbf{x})$ with binary inputs, since RBMs are universal approximators of distributions over binary inputs (Le Roux and Bengio, 2010).

An important advantage of the discriminative training objective is that it is possible to compute its gradient with respect to the ClassRBM's parameters exactly. The general form of the gradient for a single example $(\mathbf{x}_t, y_t)$ is

$$\frac{\partial \log p(y_t|\mathbf{x}_t)}{\partial\theta} = -\mathbb{E}_{\mathbf{h}|y_t,\mathbf{x}_t}\left[\frac{\partial}{\partial\theta}E(y_t, \mathbf{x}_t, \mathbf{h})\right] + \mathbb{E}_{y,\mathbf{h}|\mathbf{x}}\left[\frac{\partial}{\partial\theta}E(y, \mathbf{x}, \mathbf{h})\right]$$

and, more specifically, we obtain

$$\frac{\partial \log p(y_t|\mathbf{x}_t)}{\partial d_y} = 1_{y=y_t} - p(y|\mathbf{x}_t), \quad \forall y \in \{1, \ldots, C\}$$

for the target biases **d** and

$$\frac{\partial \log p(y_t|\mathbf{x}_t)}{\partial\theta} = \sum_j \mathrm{sigm}(o_{y_t j}(\mathbf{x}_t))\frac{\partial o_{y_t j}(\mathbf{x}_t)}{\partial\theta} - \sum_{j,y^*} \mathrm{sigm}(o_{y^* j}(\mathbf{x}_t))p(y^*|\mathbf{x}_t)\frac{\partial o_{y^* j}(\mathbf{x}_t)}{\partial\theta}$$

for the other parameters $\theta \in \{\mathbf{c}, \mathbf{U}, \mathbf{W}\}$, where $o_{yj}(\mathbf{x}) = c_j + \sum_k W_{jk}x_k + U_{jy}$. Notice that the gradient with respect to **b** is 0, since the input biases are not involved in the computation of $p(y|\mathbf{x})$. This discriminative approach has been used previously for fine-tuning the top RBM of a Deep Belief Network (Hinton, 2007).

5. Hybrid Training Objective

In order to get an idea of when and why generative training can be better than discriminative training or vice versa, we can look at some of the known theoretical properties of both approaches.

In Ng and Jordan (2002), an analysis of naive Bayes and logistic regression classifiers (which can be seen as the same parametrization but trained according to a generative or discriminative objective respectively) indicates that the generative training objective yields models that can reach more rapidly (with training set size) their best (asymptotic) generalization performance, than models trained discriminatively. However, when the model is misspecified, the discriminative training objective allows the model to reach a better performance for sufficiently large training sets (i.e., has better asymptotic performance). In Liang and Jordan (2008), it is also shown that for models from the general exponential family, parameter estimates based on the generative training objective have smaller variance than discriminative estimates. However, if the model is misspecified, generative estimates will asymptotically yield models with worse performances. These results suggest interpreting the model trained with the generative training objective as being more regularized than the model trained with the discriminative objective.

When the ultimate task is classification, adding the generative training objective to the discriminative training objective can be seen as a way to regularize the discriminative training objective. It was already found that unsupervised pre-training of RBMs can be seen as a form of regularization (Erhan et al., 2010). Since we might want to adapt the amount of regularization to the problem at hand, we could consider interpolating between the generative and discriminative objectives as in Bouchard and Triggs (2004) or, similarly, use the following hybrid objective:

$$\mathcal{L}_{\mathrm{hybrid}}(\mathcal{D}_{\mathrm{train}}) = \mathcal{L}_{\mathrm{disc}}(\mathcal{D}_{\mathrm{train}}) + \alpha\mathcal{L}_{\mathrm{gen}}(\mathcal{D}_{\mathrm{train}}) \qquad (6)$$

where the weight α of the generative criterion can be adjusted based on the performance of the model on a validation set. As in Equation 4, we can separate the $\log p(y_t, \mathbf{x}_t)$ terms in two and rewrite Equation 6 as

$$\mathcal{L}_{\mathrm{hybrid}}(\mathcal{D}_{\mathrm{train}}) = -(1+\alpha)\sum_{t=1}^{|\mathcal{D}_{\mathrm{train}}|} \log p(y_t|\mathbf{x}_t) - \alpha \sum_{t=1}^{|\mathcal{D}_{\mathrm{train}}|} \log p(\mathbf{x}_t)\ .$$

This different expression for $\mathcal{L}_{\mathrm{hybrid}}(\mathcal{D}_{\mathrm{train}})$ highlights the nature of the regularization that is imposed: among all configurations of the parameters of the ClassRBMs that can solve the supervised problem well, we will favor those that also assign high probability to the inputs and hence have extracted some of the structure present in the input's distribution.

6. Related Work

As mentioned previously, RBMs—sometimes also referred to as harmoniums (Welling et al., 2005), usually when the hidden layer's units are not binary—have been popular feature extractors in classification applications. Most of the time however, the features are learned while ignoring the target label information (Gehler et al., 2006; Xing et al., 2005). This implies that some label-related information may be thrown away and McCallum et al. (2006) have shown that incorporating labels in a feature learning procedure can be beneficial, in their work on Multi-Conditional Learning (MCL).[1] However, this latter work still required that the relationship between the hidden features and the target be learned a posteriori, by a separate classifier. One of the main points we wish to make in this paper is that RBMs provide a self-contained framework for classification, which does not need to rely on the availability of a separate classifier. This approach has two advantages: model selection is facilitated since no additional hyper-parameters from the separate classifier must be tuned, and no additional classifier training phase is required, making it possible to employ the ClassRBM in an online learning setting or to track the discriminative performance of the latent representation on a validation set. Another frequent use of RBMs is as an initializing or pretraining algorithm for deep neural networks (Hinton, 2007), but this approach shares the same disadvantages of having two training phases.

Schmah et al. (2009) proposed a different approach to discriminative training of RBMs, where each class is associated with its own individual RBM, as in a Bayes classifier. However, this approach does not rely on a global hidden representation (with shared parameters) for all classes and hence cannot model directly the latent similarity between classes, which should be advantageous for classification problems with large number of classes. From this perspective, the ClassRBM can be seen as a form of *multi-task* training, since the input to hidden weights are shared across all classes.

Yang et al. (2007) developed variants of harmoniums for video classification that can model several modalities and class information jointly. One variant uses a separate harmonium for each class as in Schmah et al. (2009), while a second is based on a shared hidden representation across classes like in the ClassRBM. However, they proposed training these models generatively, which is often not the optimal training strategy, as discussed in Section 7.

There are also several similarities between classification RBMs and ordinary multi-layer neural networks. In particular, the computation of $p(y|\mathbf{x})$ could be implemented by a single layer neural network with softplus and softmax activation functions in its hidden and output layers respectively, as well as with a special structure in the output and hidden weights where the value of the output weights is fixed and many of the hidden layer weights are shared. Glorot et al. (2011) highlight that softplus hidden activation functions tend to yield better performances than logistic-shaped functions (including the hyperbolic tangent). This might be one explanation behind the slightly superior performance of discriminatively trained ClassRBMs, compared to neural networks with hyperbolic tangent hidden units (see Sections 7.1 and 7.2). However, the main advantage of working in the framework of RBMs is that it provides a natural way to introduce generative learning, which can provide data set-dependent regularization and, as we will see, can be used to extend learning in the semi-supervised setting. As mentioned earlier, a form of generative learning can be introduced in standard neural networks with unsupervised pre-training, simply by using RBMs to initialize the hidden layer weights. However, the extent to which the final solution for the parameters of

1. We experimented with a version of MCL for the ClassRBM, however the results did not improve on those of hybrid training.

the neural network is influenced by generative learning is not well controlled, while the hybrid objective provides an explicit handle on the role played by generative learning. One could argue that the advantage of unsupervised pre-training is that it allows to build deeper models. However, consider that it is always possible to use the ClassRBM as the top layer of a stack of RBMs, in the spirit already suggested in Hinton et al. (2006).

7. Evaluation of the Training Objectives

To evaluate the performance of the different training objectives described thus far, we present experiments on two classification problems: character recognition and text classification. Such problems are particularly interesting as they are known to benefit from the extraction of non-linear features and hence are well suited for the ClassRBM. In all experiments, for the ClassRBM variants and for all baselines, we performed model selection based on the validation set performance. For the different RBM models, model selection was done with a grid-like search over the learning rate λ (between 0.0005 and 0.1, on a log scale), H (50 to 6000), the generative learning weight α for hybrid training (0 to 0.5, on a log scale) and the weight β for semi-supervised learning (0, 0.01 or 0.1). In general, bigger values of H were found to be more appropriate with more generative learning. If no local minima was apparent, the grid was extended. The biases **b**, **c** and **d** were initialized to 0 and the initial values for the elements of the weight matrices **U** and **W** were each taken from uniform samples in $\left[-m^{-0.5}, m^{-0.5}\right]$, where m is the maximum between the number of rows and columns of the matrix. The number of iterations over the training set was determined using early stopping according to the validation set classification error, with a look ahead of 15 iterations.

7.1 Character Recognition

We first evaluate the different training objectives for the ClassRBM on the problem of classifying images of digits. The images were taken from the MNIST data set, where we separated the original training set into training and validation sets of 50000 and 10000 examples and used the standard test set of 10000 examples. The results are given in Table 1. The RBM+NNet approach is simply an unsupervised RBM used to initialize a one-hidden layer supervised neural net (as in Bengio et al. 2007). We give as a comparison the results of a Gaussian kernel SVM, a random forest classifier[2] and of a regular neural network (with random initialization, one hidden layer and hyperbolic tangent hidden activation functions).

First, we observe that discriminative training of the ClassRBM outperforms generative training. However, hybrid training appears able to make the best out of both worlds and outperforms the other approaches.

We also experimented with a sparse version of the hybrid ClassRBM, since sparsity is known to be a good characteristic for features of images. Sparse RBMs were developed by Lee et al. (2008) in the context of deep neural networks. They suggest to introduce sparsity in the hidden layer of an RBM by setting the biases **c** in the hidden layer to a value that maintains the average of the conditional expected value of these neurons to an arbitrarily small value, and so after each iteration through the whole training set. This procedure tends to make the biases negative and large. We followed a different approach by simply subtracting a small constant δ value, considered as an hyper-parameter, from the biases after each update, which is more appropriate in an online setting

2. We used the implementation provided by the TreeLearn library: `https://github.com/capitalk/treelearn`.

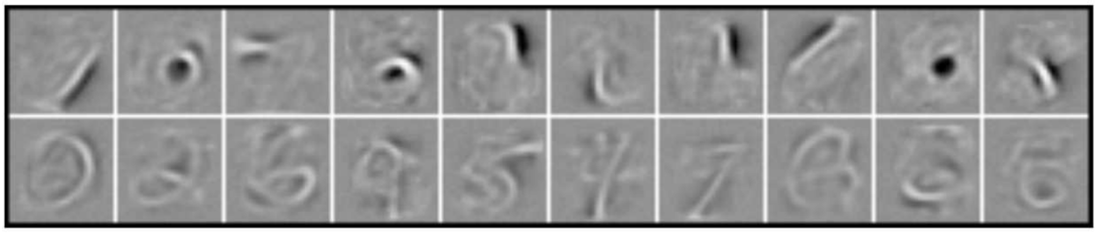


Figure 2: Filters learned by the ClassRBM on the MNIST data set. The top row shows filters that act as spatially localized stroke detectors, and the bottom shows filters more specific to a particular shape of digit.

or for large data sets. To chose δ, given the selected values for λ and α for the “non sparse” hybrid ClassRBM, we performed a second grid-search over δ (between 10^{-5} and 0.1, on a log scale) and the hidden layer size, testing bigger hidden layer sizes than previously selected.

This sparse version of the hybrid ClassRBM outperforms all the other RBM approaches, and yields significantly lower classification error than the SVM, the random forest and the standard neural network classifiers. The performance achieved by the sparse ClassRBM is particularly impressive when compared to reported performances for Deep Belief Networks (1.25% in Hinton et al. 2006) or of a deep neural network initialized using RBMs (around 1.2% in Bengio et al. 2007 and Hinton 2007) for the MNIST data set with 50000 training examples.

The discriminative power of the hybrid ClassRBM can be better understood by looking a the rows of the weight matrix **W**, which act as filter features. Figure 2 displays some of these learned filters. Some of them are spatially localized stroke detectors which can possibly be active for a wide variety of digit images, and others are much more specific to a particular shape of digit.

In practice, we find that the most influential hyper-parameters are the learning rate and the generative learning weight. Conveniently, we also find that that the best learning rate value is the same for each values of the generative learning weight we tested. In other words, finding a good learning rate does not require having found the best value for the generative learning weight. Once these two hyper-parameters are set to good values, we also find that a wide range of hidden layer sizes (between 750 to 3000) yield a competitive performance, that is, under 1.4% classification error.

7.2 Document Classification

We also evaluated the RBM models on the problem of classifying documents into their corresponding newsgroup topic. We used a version of the 20 Newsgroups data set[3] for which the training and test sets contain documents collected at different times, a setting that is more reflective of a practical application. The original training set was divided into a smaller training set and a validation set, with 9578 and 1691 examples respectively. The test set contains 7505 examples. We used the 5000 most frequent words for the binary input features. The results are given in Table 2. Again, we

3. This data set is available in Matlab format here: `http://people.csail.mit.edu/jrennie/20Newsgroups/20news-bydate-matlab.tgz`.

Model	Objective	Error
ClassRBM	Generative ($\lambda = 0.005$, $H = 6000$)	3.39%
	Discriminative ($\lambda = 0.05$, $H = 500$)	1.81%
	Hybrid ($\alpha = 0.01$, $\lambda = 0.05$, $H = 1500$)	1.28%
	Sparse Hybrid (idem + $H = 3000$, $\delta = 10^{-4}$)	**1.16%**
SVM	-	1.40%
Random Forest		2.94%
NNet		1.93%
RBM+NNet		1.41%

Table 1: Comparison of the classification performances on the MNIST data set. SVM results for MNIST were taken from `http://yann.lecun.com/exdb/mnist/`. On this data set, differences of 0.2% in classification error are statistically significant.

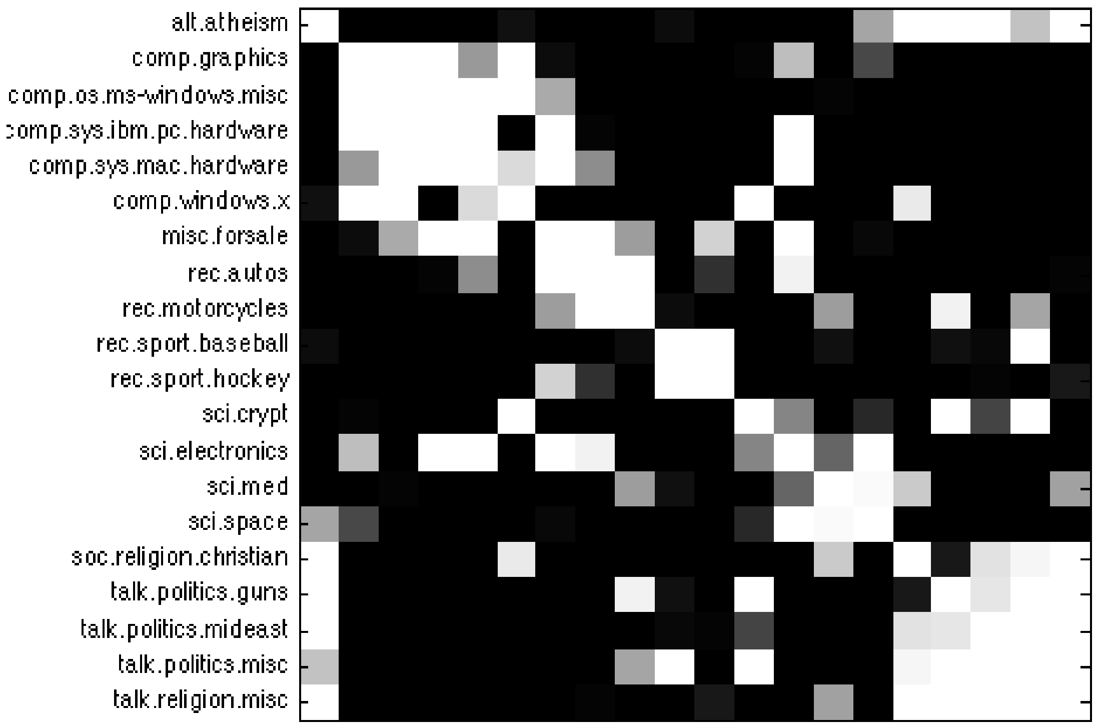


Figure 3: Similarity matrix of the newsgroup weights vectors $U_{\cdot y}$.

also provide the results of a Gaussian kernel SVM,[4] a random forest classifier and a regular neural network for comparison.

Once again, hybrid training of the ClassRBM outperforms the other approaches, including the SVM and neural network classifiers. Notice that here generative training performs better than discriminative training.

4. We used `libSVM` v2.85 to train the SVM model.

Model	Objective	Error
ClassRBM	Generative ($\lambda = 0.0005$, $H = 1000$)	24.9%
	Discriminative ($\lambda = 0.0005$, $H = 50$)	27.6%
	Hybrid ($\alpha = 0.005$, $\lambda = 0.1$, $H = 1000$)	**23.8%**
SVM	-	32.8%
Random Forest		29.0%
NNet		28.2%
RBM+NNet		26.8%

Table 2: Classification performances on 20 Newsgroups data set for the different models. The error differences between the hybrid ClassRBM and other approaches is statistically significant.

Much like for the character recognition experiment of the previous section, we find that the learning rate and generative learning weight are the most crucial hyper-parameters to tune, and that the performance is quite stable across hidden layer size as long as it is large enough (500 or greater for this problem).

In order to get a better understanding of how the hybrid ClassRBM solves this classification problem, we looked at the weights connecting each of the classes to the hidden neurons. This corresponds to the columns $\mathbf{U}_{\cdot y}$ of the weight matrix $\mathbf{U}$. Figure 3 shows a similarity matrix $\mathbf{M}(\mathbf{U})$ for the weights of the different newsgroups, where $\mathbf{M}(\mathbf{U})_{y_1 y_2} = \text{sigm}(\mathbf{U}_{\cdot y_1}^T \mathbf{U}_{\cdot y_2})$. We see that the ClassRBM does not use strictly non-overlapping sets of neurons for different newsgroups, but shares some of those neurons for newsgroups that are semantically related. We see that the ClassRBM tends to share neurons for topics such as computer (`comp.*`), science (`sci.*`) and politics (`talk.politics.*`), or secondary topics such as sports (`rec.sports.*`) and other recreational activities (`rec.autos` and `rec.motorcycles`).

Table 3 also gives the set of words used by the ClassRBM to recognize some of the newsgroups. To obtain this table we proceeded as follows: for each newsgroup y, we looked at the 20 neurons with the largest weight among $\mathbf{U}_{\cdot y}$, aggregated (by summing) the associated input-to-hidden weight vectors, sorted the words in decreasing order of their associated aggregated weights and picked the first few words according to that order. This procedure attempts to approximate the positive contribution of the words to the conditional probability of each newsgroup.

7.3 Variations on the Generative Learning Gradient Estimator

In the previous sections, we considered contrastive divergence for estimating the generative learning gradient. However, other alternatives have also been proposed. An interesting question is how much impact does the choice between these different gradient estimators has on the classification performance of the ClassRBM?

A first alternative is based on the concept of pseudolikelihood (PL) (Besag, 1975), which aims at replacing the regular likelihood objective with one more tractable, that is, for which gradients can be computed exactly. The negative log-likelihood objective on a $(\mathbf{x}, y)$ pair is then replaced by

$$-\log p(y|\mathbf{x}) - \sum_{k=1}^{D} \log p(x_k|\mathbf{x}_{\backslash k}, y) \tag{7}$$

Class	Words
alt.atheism	bible, atheists, benedikt, atheism, religion, scholars, biblical
comp.graphics	tiff, ftp, window, gif, images, pixel, rgb, viewer, image, color
comp.os.ms-windows.misc	windows, cica, bmp, window, win, installed, toronto, dos, nt
comp.sys.ibm.pc.hardware	dos, ide, adaptec, pc, config, irq, vlb, bios, scsi, esdi, dma
comp.sys.mac.hardware	apple, mac, quadra, powerbook, lc, pds, centris, fpu, power, lciii
comp.windows.x	xlib, man, motif, widget, openwindows, xterm, colormap, xdm
misc.forsale	sell, condition, floppy, week, am, obo, shipping, company, wpi
rec.autos	cars, ford, autos, sho, toyota, roads, vw, callison, sc, drive
rec.motorcycles	bikes, motorcycle, ride, bike, dod, rider, bmw, honda
rec.sport.baseball	pitching, braves, hitter, ryan, pitchers, so, rbi, yankees, teams
rec.sport.hockey	playoffs, penguins, didn, playoff, game, out, play, cup, stanley
sci.crypt	sternlight, bontchev, nsa, escrow, hamburg, encryption, rm
sci.electronics	amp, cco, together, voltage, circuits, detector, connectors
sci.med	drug, syndrome, dyer, diet, foods, physician, medicine, disease
sci.space	orbit, spacecraft, speed, safety, known, lunar, then, rockets
soc.religion.christian	rutgers, athos, jesus, christ, geneva, clh, christians, sin, paul
talk.politics.guns	firearms, handgun, firearm, gun, rkba, concealed, second, nra
talk.politics.mideast	armenia, serdar, turkish, turks, cs, argic, stated, armenians, uci
talk.politics.misc	having, laws, clinton, time, koresh, president, federal, choose
talk.religion.misc	christians, christian, bible, weiss, religion, she, latter, dwyer

Table 3: Most influential words in the hybrid ClassRBM for predicting some of the document classes

where $\mathbf{x}_{\backslash k}$ is a vector made of all elements of $\mathbf{x}$ except x_k. Hence, the model is trained to maximize the likelihood of each observed variable *given* all other observed variables. Notice that the first term corresponds to discriminative training. Hence, to obtain hybrid training we can simply weight the second summation term by α.

Equation 7 as well as its gradient with respect to the ClassRBM's parameters can be computed exactly by backpropagation. In the ClassRBM, with a development similar to the one for $p(y|\mathbf{x})$, we can show that:

$$\begin{aligned} p(x_k|\mathbf{x}_{\backslash k}, y) &= \frac{p(\mathbf{x}|y)}{p(\mathbf{x}|y) + p(\bar{\mathbf{x}}_k|y)} \\ &= \frac{\exp(x_k b_k + \sum_j \text{softplus}(c_j + U_{jy} + W_{jk}x_k + \sum_{i \neq k} W_{ji}x_i))}{\sum_{x'_k \in \{0,1\}} \exp(x'_k b_k + \sum_j \text{softplus}(c_j + U_{jy^*} + W_{jk}x'_k + \sum_{i \neq k} W_{ji}x_i))} \end{aligned}$$

where $\bar{\mathbf{x}}_k$ corresponds to $\mathbf{x}$ but where the input's k^{th} bit has been flipped. So the terms $\log p(x_k|\mathbf{x}_{\backslash k}, y)$ can be computed in $O(HD)$. A naive computation of Equation 7, which would compute the H terms $\log p(x_k|\mathbf{x}_{\backslash k}, y)$ separately, would then scale in $O(HD^2 + HC)$. However, by computing $\sum_i W_{ji}x_i$ for all j only once and reusing those terms to obtain the terms $\sum_{i \neq k} W_{ji}x_i$ for any k, we can obtain a procedure that is still linear in D, as is the CD gradient estimator. In practice, pseudolikelihood training still has some computational overhead compared to CD. Indeed, pseudolikelihood training requires $O(HD)$ computations of the exponential function, whereas CD only requires $O(H + D)$ such computations.

Generative gradient estimator	Error	
	MNIST	20News
Contrastive Divergence - 1 Gibbs sampling step	1.16%	23.8%
Contrastive Divergence - 10 Gibbs sampling step	1.15%	24.8%
Persistent Contrastive Divergence	1.41%	24.9%
Pseudolikelihood	1.21%	24.7%

Table 4: Comparison of the classification performances using different generative gradient estimators.

To perform stochastic gradient descent, a gradient step update is made according to the objective of Equation 7 every time a training example is visited. Because of the higher computational cost of PL, we use a sampling trick to estimate the gradient on the second summation term of Equation 7. Indeed, before every update, we randomly select a subset of the input variables x_k and sum only over those in the second term. This trick was necessary to scale down training to a reasonable time. We used a subset of size 100 and 500 for the MNIST and 20 Newsgroups data sets respectively.

Another generative gradient estimator for RBMs that has been recently proposed is the Persistent CD (PCD) estimator (Tieleman, 2008). PCD improves on CD by running a set of Gibbs sampling chains which persist through training, instead of always being reinitialized at each training example. Tieleman (2008) has shown that this new estimator can sometimes improve the rate of training as well as the quality of the solution that is found. As proposed by Tieleman (2008), we used 100 parallel chains for Gibbs sampling. Since we use stochastic gradient descent (instead of mini-batch gradient descent), only one chain was updated per update. The chains were updated sequentially by cycling through the set of chains.

Finally, an even simpler way of improving the gradient estimate that CD computes is to increase the number of Gibbs sampling steps that is used in the negative phase. In their experiments, Tieleman (2008) have found that CD with 10 Gibbs sampling steps often compares quite well to PCD.

Is the choice of the generative gradient estimator in the hybrid objective crucial for obtaining good classification performances? To answer this question, we have trained ClassRBMs using the hybrid objective on the MNIST and 20 Newsgroup data sets, with the different generative gradient estimators. Hyper-parameters were tuned separately for each variant, as in the previous sections. For the MNIST data set, we used the sparse training variant. The results of this experiment are given in Table 4. We see that in general, none of the alternative estimators provided significant performance improvements. On 20 Newsgroups, the performance even worsened. We notice that the performance obtained with PCD tends to be the particularly bad. This is explained by the fact that PCD requires smaller learning rates to work well, so that the model doesn't change faster than the rate in which the parallel Gibbs chains mix. However, using a small learning rate does not correspond to the regime at which the ClassRBM performs best in terms of classification error for these problems. This is particularly true for MNIST where the optimal learning rate is between 0.05 and 0.1.

7.4 Scaling Up to Large Input Spaces

Computing the generative gradient is typically much more computationally expensive than the discriminative gradient. This is particularly true on problems were the input data is very high-dimensional and sparse, such as the text classification problem. For instance, though we have restricted its dimensionality to 5000, the 20 Newsgroup data set could be made much more high dimensional by including more words as input features. While the computations for estimating the discriminative gradient can take advantage of the sparsity of the input (mainly when multiplying the input with the filters), estimating the generative gradient for all of the estimators in Section 7.3 requires an explicit loop over all inputs.

It would hence be beneficial to derive a more general generative gradient estimator that would allow us to control more directly its computational cost, and perhaps let us trade a little bit of accuracy for more computational efficiency. This would particularly be useful in an online learning setting, where a stream of training examples is available, with examples being presented at some given rate. In such a setting, we might want to reduce the computational time required by the generative learning objective so that updating the parameters of the ClassRBM for a training example can be done before the next sample is given.

As mentioned, the computational expense of training is closely related to the number of variables who's distribution is being modelled. At one extreme, discriminative learning is very efficient since we are only modelling the (conditional) distribution of the target variable while, at the other extreme, generative learning is much more expensive because the distribution of the target and all input variables is being modelled. Hence, a good handle over the computational complexity of an estimator would be the total number of variables involved in the conditional distribution on which the training objective is based.

Following this idea, let $I=\{1,\dots,D\}$ be the set of input variable indices and let $\mathcal{P}_{=L}(I)$ be all the subsets of I of cardinality L, we could define the following as our new computation-aware generative training objective

$$-\sum_{t=1}^{|\mathcal{D}_{\text{train}}|} \mathbb{E}_{\mathcal{S}\in\mathcal{P}_{=L}(I)}\left[\log p(y_t,\mathbf{x}_{\mathcal{S}}|\mathbf{x}_{\backslash\mathcal{S}})\right] = -\sum_{t=1}^{|\mathcal{D}_{\text{train}}|}\sum_{\mathcal{S}\in\mathcal{P}_{=L}(I)}\frac{1}{|\mathcal{P}_{=L}(I)|}\log p(y_t,\mathbf{x}_{\mathcal{S}}|\mathbf{x}_{\backslash\mathcal{S}}) \tag{8}$$

where $\mathbf{x}_{\mathcal{S}}$ is the vector of input variables with index in $\mathcal{S}$ and $\mathbf{x}_{\backslash\mathcal{S}}$ is the vector of all other variables. Put briefly, this objective aims at maximizing the conditional likelihood of the target and all subsets of input variables of size L given the other variables, and with a uniform distribution or weight on all such possible partitions of the inputs. Since the expectation over $\mathcal{S}$ is intractable even for relatively small values of L, in practice we approximate it by sampling a single value from the associated uniform distribution over $\mathcal{S}$, and so for every parameter update.

This training objective actually corresponds to a particular type of composite likelihood estimator (Lindsay, 1988; Liang and Jordan, 2008). Here, we in addition propose to approximate the gradients of the $\log p(y_t,\mathbf{x}_{\mathcal{S}}|\mathbf{x}_{\backslash\mathcal{S}})$ terms

$$\frac{\partial \log p(y_t,\mathbf{x}_{\mathcal{S}}|\mathbf{x}_{\backslash\mathcal{S}})}{\partial\theta} = -\mathbb{E}_{\mathbf{h}|y_t,\mathbf{x}_t}\left[\frac{\partial}{\partial\theta}E(y_t,\mathbf{x}_t,\mathbf{h})\right] + \mathbb{E}_{y,\mathbf{x}_{\mathcal{S}},\mathbf{h}|\mathbf{x}_{\backslash\mathcal{S}}}\left[\frac{\partial}{\partial\theta}E(y,\mathbf{x},\mathbf{h})\right]$$

by using contrastive divergence with one step of Gibbs sampling. This approximation requires that only the L variables in $\mathbf{x}_{\mathcal{S}}$ be sampled, making this procedure efficient for small L.

Model	Hybrid training (for varying L)				Discriminative
	250	500	5000	10000	training
Error	22.9%	22.2%	21.9%	21.9%	26.9%

Table 5: Evaluation of the composite likelihood variant of contrastive divergence on the 20 Newsgroups data set, with an input dimensionality of 25247.

We experimentally investigated how varying L impacts the performance of ClassRBMs trained using a hybrid objective based on the generative objective of Equation 8. We took the 20 Newsgroups data set and, instead of only using the 5000 most frequent words as features, we considered all words appearing at least 5 times, adding up to 25247 words. The results are given in Table 5. We observe a big improvement on the classification error obtained by restricting the input to only 5000 words, as in Table 2. The performance of purely discriminative training in the large vocabulary setting, which is essentially equivalent to setting $L = 0$, is also improved on. We see that the composite likelihood variant still allows for better generalization performance to be achieved, even for relatively small values of L. Interestingly, we also observe a fairly rapid diminishing return in the improvement of generalization error as L increases.

The idea of combining composite likelihood objectives and contrastive divergence has also been combined previously by Asuncion et al. (2010), but in a different way. Asuncion et al. (2010) focused on models for which standard contrastive divergence with Gibbs sampling corresponds to sampling only a single randomly selected variable at each step. In this case, contrastive divergence with one sampling step actually corresponds to a stochastic version of pseudolikelihood (Hyvärinen, 2006). They propose instead to use block-Gibbs sampling on randomly selected blocks of variables of limited size L at each step. L must be small however since, in general, computing the associated conditionals is exponential in L. Using a single sampling step then corresponds to a stochastic version of composite likelihood. They show that increasing L and using a single Gibbs step can be more advantageous than using $L = 1$ and increasing the number of iterations. Their work can be understood as an investigation of how to improve contrastive divergence using ideas from composite likelihood objectives.

However, for RBMs, block-Gibbs sampling is actually the standard, where we first sample all hidden units and then all input variables in one iteration. Hence, the approach of Asuncion et al. (2010) is not directly applicable here. What we propose instead, is to apply contrastive divergence to a composite likelihood objective, such that we approximate the gradients on the $\log p(y_t, \mathbf{x}_{\mathcal{S}}|\mathbf{x}_{\backslash\mathcal{S}})$ terms. Crucially, this approach is linear in L, as opposed to exponential.

8. Semi-supervised Learning

In certain situations, in addition to a (possibly small) set of labeled training examples $\mathcal{D}_{\text{train}}$, even more data can be obtained in the form of an unlabeled training set $\mathcal{D}_{\text{unlab}} = \{(\mathbf{x}_t)\}$. This is particularly true for data such as images and text documents, for which the Internet is an almost infinite source. Semi-supervised learning algorithms (Chapelle et al., 2006) address this situation by leveraging the unlabeled data to bias learning towards solutions that are also "consistent" with the unlabeled data. Different algorithms can then be seen as defining different notions of consistency.

Because a ClassRBM is a proper generative model, a very natural notion of consistency in this context is that unlabeled training data have high likelihood under it. To achieve this, one can optimize the following negative log-likelihood

$$\mathcal{L}_{\text{unsup}}(\mathcal{D}_{\text{unlab}}) = -\sum_{t=1}^{|\mathcal{D}_{\text{unlab}}|} \log p(\mathbf{x}_t) \tag{9}$$

which requires computing the gradients

$$\frac{\partial \log p(\mathbf{x}_t)}{\partial \theta} = -\mathbb{E}_{y,\mathbf{h}|\mathbf{x}_t}\left[\frac{\partial}{\partial \theta} E(y_t, \mathbf{x}_t, \mathbf{h})\right] + \mathbb{E}_{y,\mathbf{x},\mathbf{h}}\left[\frac{\partial}{\partial \theta} E(y, \mathbf{x}, \mathbf{h})\right] .$$

The contrastive divergence approximation proceeds slightly differently here. The first term can be computed in time $O(HD+HC)$, by noticing that

$$\mathbb{E}_{y,\mathbf{h}|\mathbf{x}_t}\left[\frac{\partial}{\partial \theta} E(y_t, \mathbf{x}_t, \mathbf{h})\right] = \mathbb{E}_{y|\mathbf{x}_t}\left[\mathbb{E}_{\mathbf{h}|y,\mathbf{x}_t}\left[\frac{\partial}{\partial \theta} E(y_t, \mathbf{x}_t, \mathbf{h})\right]\right]$$

and then either average the usual RBM gradient $\frac{\partial}{\partial \theta} E(y_t, \mathbf{x}_t, \mathbf{h})$ for each class y (weighted by $p(y|\mathbf{x}_t)$), or sample from $p(y|\mathbf{x}_t)$ and only collect the gradient for the sampled value of y. In the latter sampling version, the online training update for this objective can be described as replacing the statement $y^0 \leftarrow y_t$ with $y^0 \sim p(y|\mathbf{x}_t)$ in Algorithm 1. We used this version in our experiments.

In order to perform semi-supervised learning, we can weight and combine the objective of Equation 9 with those of Equations 3, 5 or 6 as follows:

$$\mathcal{L}_{\text{semi-sup}}(\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{unlab}}) = \mathcal{L}_{\text{TYPE}}(\mathcal{D}_{\text{train}}) + \beta \mathcal{L}_{\text{unsup}}(\mathcal{D}_{\text{unlab}}) \tag{10}$$

where TYPE $\in \{gen, disc, hybrid\}$. Online training by stochastic gradient descent then corresponds to applying two gradients updates: one for the objective $\mathcal{L}_{\text{TYPE}}$ and one for the unlabeled data objective $\mathcal{L}_{\text{unsup}}$.

We evaluated our semi-supervised learning algorithm for the hybrid ClassRBM on both previous digit recognition and document classification problems. We also experimented with a version (noted MNIST-BI) of the MNIST data set proposed by Larochelle et al. (2007) where background images have been added to MNIST digit images. This version corresponds to a much harder problem and it will help to illustrate the advantage brought by semi-supervised learning in ClassRBMs. The ClassRBM trained on this data used truncated exponential input units (see Bengio et al., 2007).

In this semi-supervised setting, we reduced the size of the labeled training set to 800 examples, and used some of the remaining data to form an unlabeled data set $\mathcal{D}_{\text{unlab}}$. The validation set was also reduced to 200 labeled examples. Model selection covered all the parameters of the hybrid ClassRBM as well as the unsupervised objective weight β of Equation 10, with $\beta = 0.1$ for MNIST and 20 Newsgroups, and $\beta = 0.01$ for MNIST-BI performing best. For comparison purposes, we also provide the performance of a standard non-parametric semi-supervised learning algorithm based on function induction (Bengio et al., 2006a), which is very similar to other non-parametric semi-supervised learning algorithms such as Zhu et al. (2003). We provide results for the use of a Gaussian kernel (NP-Gauss) and a data-dependent truncated Gaussian kernel (NP-Trunc-Gauss) used in Bengio et al. (2006a), which essentially outputs zero for pairs of inputs that are not near neighbors. The experiments on the MNIST and MNIST-BI (with background images) data

Model	Objective	MNIST	MNIST-BI	20News
ClassRBM	Hybrid	9.73%	42.4%	40.5%
	Semi-supervised + Hybrid	8.04%	**37.5%**	**31.8%**
NP-Gauss	-	10.60%	66.5%	85.0%
NP-Trunc-Gauss		**7.49%**	61.3%	82.6%

Table 6: Comparison of the classification errors in semi-supervised learning setting. The errors in bold are statistically significantly better.

sets used 5000 unlabeled examples and the experiment on 20 Newsgroups used 8778. The results are given in Table 6, where we observe that semi-supervised learning consistently improves the performance of the ClassRBM trained based on the hybrid objective.

The usefulness of non-parametric semi-supervised learning algorithms has been demonstrated many times in the past, but usually so on problems where the dimensionality of the inputs is low or the data lies on a much lower dimensional manifold. This is reflected in the result on MNIST for the non-parametric methods. However, for high dimensional data with many factors of variation, these methods can quickly suffer from the curse of dimensionality, as argued by Bengio et al. (2006b). This is also reflected in the results for the MNIST-BI data set which contains many factors of variation, and for the 20 Newsgroups data set where the input is very high dimensional. Finally, it is important to notice that semi-supervised learning in ClassRBMs proceeds in an online fashion and hence could scale to very large data sets, unlike most non-parametric methods.

We mention that, in the context of log-linear models, Druck et al. (2007) introduced semi-supervised learning in hybrid generative/discriminative models using a similar approach to the one presented in here. While log-linear models depend much more on the discriminative quality of the features that are fed as input, the ClassRBM can learn useful features through its hidden layer and model non-linear decision boundaries.

9. Multitask Learning

The classification problems considered so far had in common that a given input could only belong to a single class, that is, classes were mutually exclusive. For certain problems, this assumption is too restrictive and inputs can be simultaneously associated with multiple classes or labels. One example is online collections of images, documents or music augmented with social tags (see Lamere 2008 for an example), which are short descriptions applied by users to items and can be used by users to search and browse through a collection. One approach to this problem would be to train a separate classifier for each tag. However, a better approach is to perform multitask learning (Caruana, 1997), where a single model is trained to perform all tasks simultaneously. This allows for the model to leverage the similarity between certain tasks and improve generalization.

We describe here how multitask learning can also be performed within a ClassRBM. In this context, the target's representation in the energy function of the ClassRBM does not follow the "one out of C" constraint and is an unconstrained binary vector $\mathbf{y}$. The conditional distribution of $\mathbf{y}$ given

h then becomes:

$$p(\mathbf{y}|\mathbf{h}) = \prod_c p(y_c|\mathbf{h}), \text{ with } p(y_c = 1|\mathbf{h}) = \text{sigm}(d_c + \sum_i U_{jc}h_j).$$

Another important implication of this change is that the predictive posterior $p(\mathbf{y}|\mathbf{x})$ is no longer tractable, since $\mathbf{y}$ now has 2^C possible values. At test time, we are particularly interested in estimating $p(y_c = 1|\mathbf{x})$ for each label, in order to make a prediction of the binary value of each individual label. Fortunately, there exist several message-passing approximate inference procedures for general graphical models that can be employed here. The two most popular are mean field and loopy belief propagation.

The mean field (MF) approach tries to approximate the joint posterior $p(\mathbf{y},\mathbf{h}|\mathbf{x})$ by a factorial distribution $q(\mathbf{y},\mathbf{h}) = \prod_{c=1}^{C} \mu_c^{y_c}(1-\mu_c)^{1-y_c} \prod_{j=1}^{n} \tau_j^{h_j}(1-\tau_j)^{1-h_j}$ that minimizes the Kullback-Leibler (KL) divergence with the true posterior. Running the following message passing procedure to convergence

$$\begin{aligned}
\mu_c &\leftarrow \text{sigm}\left(d_c + \sum_j U_{jc}\tau_j\right) \ \forall\, c \in \{1,\ldots,C\}, \\
\tau_j &\leftarrow \text{sigm}\left(c_j + \sum_c U_{jc}\mu_c + \sum_i W_{ji}x_i\right) \ \forall\, j \in \{1,\ldots,n\}
\end{aligned}$$

we can reach a saddle point of the KL divergence, at which point μ_c serves as the estimate for $p(y_c = 1|\mathbf{x})$ and τ_j can be used to estimate $p(h_j = 1|\mathbf{x})$. In our experiments, we initialized the messages to 0. Moreover, we treat the number of message passing iterations as an hyper-parameter, so as to control the computational cost of inference.

Loopy belief propagation (Pearl, 1988) (LBP) also relies on a message passing procedure between variables. LBP is more complex than MF in that the number of distinct messages to be maintained scales in $O(HC)$, that is, the number of connections between $\mathbf{y}$ and $\mathbf{h}$, instead of in $O(H+C)$ as in MF. It also provides a direct estimate of the pair-wise probabilities $p(y_c = 1, h_j = 1|\mathbf{x})$. LBP tends to give estimates of the true marginals that are more accurate than the iterative MF procedure (Weiss, 2001). While not guaranteed to converge it frequently does in practice. One method that has been shown to be useful in aiding convergence is message damped belief propagation (Pretti, 2005). In this case the normal updates computed by belief propagation are mixed with the previous updates in order to smooth them, the damping factor being a parameter of the algorithm. Algorithm 2 details the procedure.

As for learning, the discriminative gradient expression, which is now

$$\frac{\partial \log p(\mathbf{y}_t|\mathbf{x}_t)}{\partial\theta} = -\mathbb{E}_{\mathbf{h}|\mathbf{y}_t,\mathbf{x}_t}\left[\frac{\partial}{\partial\theta}E(\mathbf{y}_t,\mathbf{x}_t,\mathbf{h})\right] + \mathbb{E}_{\mathbf{y},\mathbf{h}|\mathbf{x}}\left[\frac{\partial}{\partial\theta}E(\mathbf{y},\mathbf{x},\mathbf{h})\right]$$

must also be approximated, specifically the second expectation over $\mathbf{y}$ and $\mathbf{h}$. Contrastive divergence is a natural approach to estimating this expectation, using K iterations of Gibbs sampling alternating between sampling $\mathbf{h}$ and $\mathbf{y}$.

However, MF or LBP can also be used to approximate the expectation. Because the energy function decomposes into sums of either unary or pairwise terms, only the marginals $p(y_c = 1|\mathbf{x})$, $p(h_j = 1|\mathbf{x})$ and $p(y_c = 1, h_j = 1|\mathbf{x})$ are required. The assumption of a factorial distribution behind

Algorithm 2 Loopy Belief Propagation algorithm for inference in the multilabel ClassRBM

```
Input: training pair $(\mathbf{y},\mathbf{x})$, number of iterations $K$ and damping factor $\beta$
$m^{\uparrow}_{jc} \leftarrow 0,\ \ m^{\downarrow}_{jc} \leftarrow 0 \quad \forall\, c, j$
$\mathbf{c}^{\text{data}} \leftarrow \mathbf{c} + \mathbf{W}\mathbf{x}$

# Update downwards (towards $\mathbf{y}$) and upwards (towards $\mathbf{h}$) messages
for $K$ iterations do
    $m^{\downarrow}_{jc} \leftarrow \beta m^{\downarrow}_{jc} + (1-\beta)\log\left(1 + (\exp(U_{jc}) - 1)\ \text{sigm}(c^{\text{data}}_j + \sum_{c^* \neq c} m^{\uparrow}_{jc^*})\right), \quad \forall\, c, j$
    $m^{\uparrow}_{jc} \leftarrow \beta m^{\uparrow}_{jc} + (1-\beta)\log\left(1 + (\exp(U_{jc}) - 1)\ \text{sigm}(d_c + \sum_{j^* \neq j} m^{\downarrow}_{j^*c})\right), \quad \forall\, c, j$
end for

# Compute estimated marginals
$p^{\text{LBP}}(y_c = 1|\mathbf{x}) \leftarrow \text{sigm}(d_c + \sum_j m^{\downarrow}_{jc}), \quad \forall\, c$
$p^{\text{LBP}}(h_j = 1|\mathbf{x}) \leftarrow \text{sigm}(c^{\text{data}}_j + \sum_c m^{\uparrow}_{jc}), \quad \forall\, j$

$\text{num}^{01}_{jc} \leftarrow d_c + \sum_{j^* \neq j} m^{\downarrow}_{j^*c},\ \ \text{num}^{10}_{jc} \leftarrow c^{\text{data}}_j + \sum_{c^* \neq c} m^{\uparrow}_{jc^*}, \quad \forall\, c, j$
$\text{num}^{11}_{jc} \leftarrow U_{jc} + \text{num}^{10}_{jc} + \text{num}^{01}_{jc}, \quad \forall\, c, j$
$p^{\text{LBP}}(y_c = 1, h_j = 1|\mathbf{x}) = \exp(\text{num}^{11}_{jc})/(\exp(\text{num}^{11}_{jc}) + \exp(\text{num}^{01}_{jc}) + \exp(\text{num}^{10}_{jc})), \quad \forall\, c, j$
```

MF means that $p(y_c = 1, h_j = 1|\mathbf{x})$ is simply estimated as the product of its estimates for $p(y_c = 1|\mathbf{x})$ and $p(h_j = 1|\mathbf{x})$, while LBP provides a more sophisticated estimate. The MF gradient estimates can also be improved by initializing the μ_c message to the value of the associated training target y_k. This approach was first described by Welling and Hinton (2002) and is known as mean field contrastive divergence. It was also extended to general variational approximations in Welling and Sutton (2005). When making predictions at test time however, we still must initialize μ_c to 0.

Finally, as in Section 7.3, the intractability of discriminative maximum likelihood training can be avoided by using a pseudolikelihood objective $-\sum_{c=1}^{C} \log p(y_c|\mathbf{y}_{\backslash c}\mathbf{x})$ for which exact gradients can be computed.

Given all of these possible ways of approximating the marginal posteriors $p(y_c = 1|\mathbf{x})$ at test time and of performing discriminative training, we performed an extensive comparison of all possible combinations of such choices. We used three different music social tags data sets based on databases of 10-second song clips. The first data set, was collected from Amazon.com's Mechanical Turk service and is described in Mandel et al. (2010). The second data set was collected from the MajorMiner music labeling game and is described in Mandel and Ellis (2008). The final data set was collected from Last.fm's website and is described in Schifanella et al. (2010). We will refer to these data sets as MTurk, MajMin and Last.fm respectively.

All of these data sets were in the form of (user, item, tag) triples, where the items were either 10-second clips of tracks or whole tracks. These data were condensed into (item, tag, count) triples by summing across users. Converting (item, tag, count) triples to binary matrices for training and evaluation purposes required some care. In the MajorMiner and Last.fm data, the counts were high enough that we could require the verification of an (item, tag) pair by at least two people, meaning that the count had to be at least 2 to be considered as a positive example. The Mechanical Turk data set did not have high enough counts to allow this, so we had to count every (item, tag) pair.

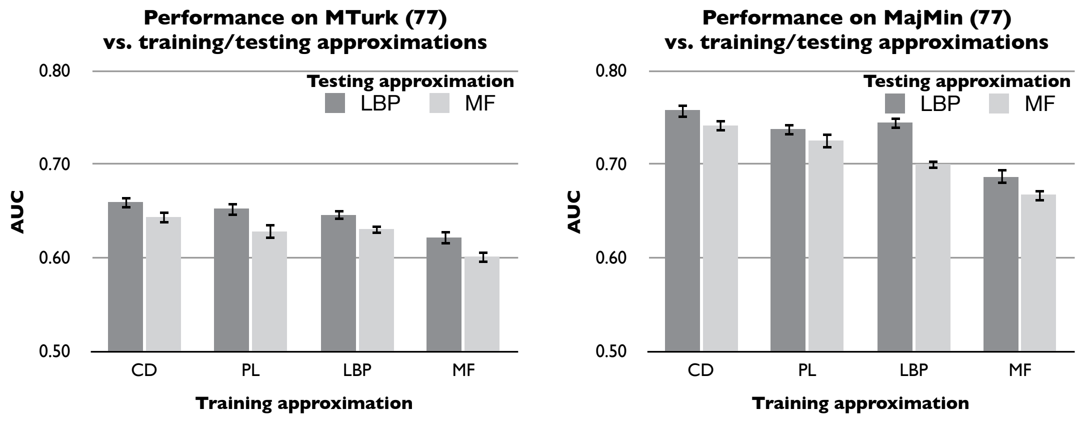
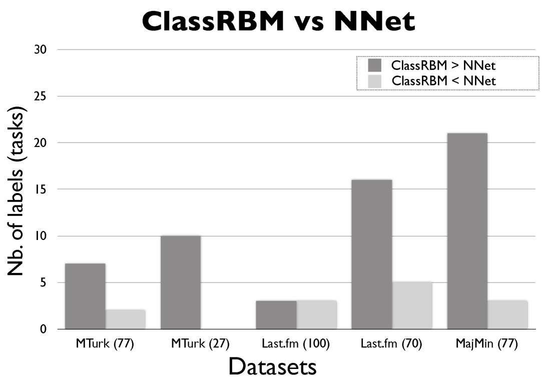


Figure 4: Results of the multilabel ClassRBM (discriminative training) on the Mechanical Turk and MajorMiner data sets, comparing the performance of different approximation combinations for training and testing. The approximations used during training are represented on the x-axis, while the approximations used during testing are represented through the color of the bar. The error bars correspond to the standard error across folds.

In the MajorMiner and Last.fm data sets, (item, tag) pairs with only a single count were not used as negative examples because we assumed that they had higher potential relevance than (item, tag) pairs that never occurred, which served as stronger negative examples.

The timbral and rhythmic features of Mandel and Ellis (2008) were used to characterize the audio of 10-second song clips. Each dimension of both sets of features was normalized across the database to have zero-mean and unit-variance, and then each feature vector was normalized to be unit norm to reduce the effect of outliers. The timbral features were 189-dimensional and the rhythmic features were 200-dimensional, making the combined feature vector 389-dimensional.

In order to asses the impact of different approximations (of the gradients or $p(\mathbf{y}|\mathbf{x})$) on the solution found by the model we only considered discriminative learning. We also augmented the number of data sets by changing the number of tags, to see how this factor influences the results. Next to a data set name, the number in parenthesis thus indicates the number of tags considered. The tags were selected by sorting them by popularity and picking the leading tags. For all data sets we select the hyper-parameters of the model using a 5-fold cross-validation. In order to increase the accuracy of our procedure, for each fold we computed the score as an average across 4 sub-folds. Each run used a different fold (from the remaining 4 folds) as the validation set and the other 3 as the training set. From this validation procedure, 50, 100 and 200 hidden units were selected respectively for the MTurk, MajMin and Last.fm data sets and a learning rate of 0.01 for all data sets. We also fixed a priori the number of iterations for approximating the gradients (for CD, MF or LBP) to 10, and the number of MF or LBP iterations for approximating $p(\mathbf{y}|\mathbf{x})$ to 20, to limit



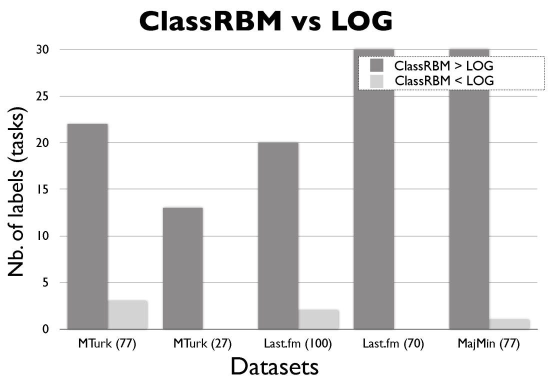


Figure 5: Comparison of the multilabel ClassRBM and with a multitask neural network (NNet) and with single task logistic regression classifiers (LOG). Bars show the number of labels (tags) on which the ClassRBM is significantly better (>) or worse (<) than the baseline in a two-sided paired t-test.

the hyper-parameter search space.[5] Finally, we set to 0.9 the damping factor for LBP inference, but other values were found to yield similar performances.

Figure 4 provides the performance of all possible combinations of approximations at training and test time, on two data sets. The performance is evaluated in terms of retrieval performance using the area under the ROC curve (AROC) (Cortes and Mohri, 2004).[6] We measure the AROC for each tag separately and use the average across tags and folds as an overall measure of performance. As we see, contrastive divergence tends to outperform other approaches for training the ClassRBM, either when mean field or loopy belief propagation is used at test time. Using the same deterministic inference at training and test time hence appears not to be optimal, with mean field being the worst option.

We also compared the performance of the ClassRBM with two baselines. The first is a multitask neural network (Caruana, 1997), which is among the best baselines for multitask learning. Moreover, a neural network makes for an interesting comparison because its prediction for the marginals $p(y_c = 1|\mathbf{x})$ is also non-linear, but feedforward and non-recursive, unlike in the ClassRBM. The second baseline is a set of single task logistic regression classifiers (one for each task). Though previous work on these multitask data sets has instead considered single task SVMs as a baseline (Mandel et al., 2011a), we have found logistic regression classifiers to outperform SVMs, hence we use those here as the single task baseline.

The same model selection procedure was used to select the baselines' hyper-parameters, namely the learning rate (both baselines) and hidden layer size (neural network baseline only). Contrastive divergence and loopy belief propagation was used in this comparison, for discriminative training.

5. We validated this choice for these hyper-parameters afterwards, based on the best learning rate and hidden layer size found, and observed that while the performance increases with the number of iterations, the increase is not considerable, especially when we account for the increase in training time.
6. This metric scores the ability of an algorithm to rank relevant examples in a collection above irrelevant examples. A random ranking will achieve an AROC of approximately 0.5, while a perfect ranking will achieve an AROC of 1.0.

Model	MTurk (77)	MTurk (27)	Last.fm (100)	Last.fm (70)	MajMin (77)
ClassRBM	65.9	68.8	72.4	72.2	76.1
NNet	65.8	65.4	72.4	72.0	75.3
LOG	63.4	65.7	70.2	70.3	70.7

Table 7: Average AROC across labels as a percentage for each model on all multitask data sets.

We compared the ClassRBM in a head to head fashion with each baseline, and computed a two-sided paired t-test across folds, per tag, to count the number of tags for which either model performs significantly better than the other. As illustrated in Figure 5, the ClassRBM is a better classifier for strictly more tags on all data sets when compared to the logistic regression approach and on 4 out of 5 data sets when compared to the neural network (with a tie on the remaining data set). Finally, Table 7 gives the absolute performance of the ClassRBM and the baselines.

10. Conclusion

We argued that RBMs can and should be used as stand-alone non-linear classifiers alongside other standard and more popular classifiers, instead of merely being considered as simple feature extractors. We considered different training strategies for the Classification RBM and evaluated them. In particular, we highlighted the importance of combining generative and discriminative training and we explored the impact of using different generative gradient estimators on the classification performance of the ClassRBM. We also extended the range of situations where the ClassRBM can be employed, by presenting learning algorithms tailored to settings where unlabeled data are available, where the input is sparse and very high-dimensional, as well as when multiple classification problems must be solved.

By describing and establishing the ClassRBM as a “black box” classifier in its own right, we hope to make its use more accessible and stimulate research in how to adapt it to even more application settings. As an illustration of this potential, we end by mentioning extensions of the ClassRBM that have already been developed, since the first conference publication of this work (Larochelle and Bengio, 2008). Gelfand et al. (2010) explored a different way of using the ClassRBM energy function to perform classification, using a conditional herding learning algorithm. Memisevic et al. (2010) investigated a variant of the ClassRBM with third-order (as opposed to pair-wise) interactions between the input, target and hidden units. van der Maaten et al. (2011) developed an extension for sequential classification problems with linear-chain interactions between the sequence of targets, while Mnih et al. (2011) considered other structured output prediction problems such as denoising. Finally Louradour and Larochelle (2011) adapted the ClassRBM to problems where the input $\mathbf{x}$ is a set containing an arbitrary number of input vectors.

Acknowledgments

We thank Dumitru Erhan for discussions about sparse RBMs. Earlier versions of some of the material presented here appeared in the ICML 2008 conference proceedings (Larochelle and Bengio, 2008) and on arXiv (Mandel et al., 2011b).

References

Arthur Asuncion, Qiang Liu, Alexander Ihler, and Padhraic Smyth. Learning with blocks: Composite likelihood and contrastive divergence. In *Proceedings of The Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS'10)*, pages 33–40, 2010.

Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. Label propagation and quadratic criterion. In Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors, *Semi-Supervised Learning*, pages 193–216. MIT Press, 2006a. URL `http://www.iro.umontreal.ca/~lisa/pointeurs/bengio_ssl.pdf`.

Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. The curse of highly variable functions for local kernel machines. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18 (NIPS'05)*, pages 107–114. MIT Press, Cambridge, MA, 2006b.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In Bernhard Schölkopf, John Platt, and Thomas Hoffman, editors, *Advances in Neural Information Processing Systems 19 (NIPS'06)*, pages 153–160. MIT Press, 2007.

Julian Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3):179–195, 1975.

Guillaume Bouchard and Bill Triggs. The tradeoff between generative and discriminative classifiers. In *IASC International Symposium on Computational Statistics (COMPSTAT)*, pages 721–728, Prague, August 2004.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

Olivier Chapelle, Bernhard. Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

Corinna Cortes and Mohri Mohri. Auc optimization vs. error rate minimization. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16 (NIPS'03)*, volume 16, Cambridge, MA, 2004. MIT Press.

Gregory Druck, Chris Pal, Andrew Mccallum, and Xiaojin Zhu. Semi-supervised classification with hybrid generative/discriminative methods. In *Proceedings of the Thirteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'07)*, pages 280–289, New York, NY, USA, 2007. ACM.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, February 2010.

Peter V. Gehler, Alex D. Holub, and Max Welling. The rate adapting poisson model for information retrieval and object recognition. In William W. Cohen and Andrew Moore, editors, *Proceedings of the Twenty-three International Conference on Machine Learning (ICML'06)*, pages 337–344, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2. doi: http://doi.acm.org/10.1145/1143844.1143887.

Andrew Gelfand, Yutian Chen, Laurens van der Maaten, and Max Welling. On herding and the perceptron cycling theorem. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23 (NIPS'10)*, pages 694–702. Curran Associates, 2010.

Xavier Glorot, Antoire Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS'11)*, April 2011.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

Geoffrey E. Hinton. To recognize shapes, first learn to generate images. In Paul Cisek, Trevor Drew, and John Kalaska, editors, *Computational Neuroscience: Theoretical Insights into Brain Function*. Elsevier, 2007.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Aapo Hyvärinen. Consistency of Pseudolikelihood Estimation of Fully Visible Boltzmann Machines. *Neural Computation*, 18:2283–2292, 2006.

Paul Lamere. Social tagging and music information retrieval. *Journal of New Music Research*, 37 (2):101–114, 2008.

Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted Boltzmann machines. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 536–543. ACM, 2008.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In Zoubin Ghahramani, editor, *Proceedings of the Twenty-fourth International Conference on Machine Learning (ICML'07)*, pages 473–480. ACM, 2007.

Nicolas Le Roux and Yoshua Bengio. Deep belief networks are compact universal approximators. *Neural Computation*, 22(8):2192–2207, August 2010. ISSN 0899-7667.

Honglak Lee, Chaitanya Ekanadham, and Andrew Ng. Sparse deep belief net model for visual area V2. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS'07)*, pages 873–880. MIT Press, Cambridge, MA, 2008.

Percy Liang and Michael I. Jordan. An asymptotic analysis of generative, discriminative, and pseudolikelihood estimators. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 584–591, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: http://doi.acm.org/10.1145/1390156.1390230.

Bruce G. Lindsay. Composite likelihood methods. *Contemporary Mathematics*, 80:221–239, 1988.

Jérôme Louradour and Hugo Larochelle. Classification of sets using restricted Boltzmann machines. In *Proceedings of the Twenty-seventh Conference on Uncertainty in Artificial Intelligence (UAI'11) (to appear)*. AUAI Press, 2011.

Michael I. Mandel and Daniel P. W. Ellis. A web-based game for collecting music metadata. *Journal of New Music Research*, 37(2):151–165, 2008.

Michael I. Mandel, Douglas Eck, and Yoshua Bengio. Learning tags that vary within a song. In *Proceedings of the Eleventh International Conference on Music Information Retrieval (ISMIR)*, pages 399–404, August 2010.

Michael I. Mandel, Razvan Pascanu, Douglas Eck, Yoshua Bengio, Luca M. Aiello, Rossano Schifanella, and Filippo Menczer. Contextual tag inference. *ACM Transactions on Multimedia Computing, Communications and Applications*, 7S(1):32:1–32:18, October 2011a.

Michael I. Mandel, Razvan Pascanu, Hugo Larochelle, and Yoshua Bengio. Autotagging music with conditional restricted Boltzmann machines. *ArXiv e-prints*, March 2011b.

Andrew McCallum, Chris Pal, Gregory Druck, and Xuerui Wang. Multi-conditional learning: Generative/discriminative training for clustering and classification. In *Twenty-first National Conference on Artificial Intelligence (AAAI'06)*. AAAI Press, 2006.

Roland Memisevic, Christopher Zach, Geoffrey Hinton, and Marc Pollefeys. Gated softmax classification. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23 (NIPS'10)*, pages 1603–1611. Curran Associates, 2010.

Andriy Mnih and Geoffrey E. Hinton. Three new graphical models for statistical language modelling. In Zoubin Ghahramani, editor, *Proceedings of the Twenty-fourth International Conference on Machine Learning (ICML'07)*, pages 641–648. ACM, 2007.

Volodymyr Mnih, Hugo Larochelle, and Geoffrey E. Hinton. Conditional restricted boltzmann machines for structured output prediction. In *Proceedings of the Twenty-seventh Conference on Uncertainty in Artificial Intelligence (UAI'11) (to appear)*. AUAI Press, 2011.

Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14 (NIPS'01)*, pages 841–848, 2002.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems:Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Marco Pretti. A message-passing algorithm with damping. *Journal of Statistical Mechanics: Theory and Experiment*, page P11008, 2005.

Ruslan Salakhutdinov and Geoffrey E. Hinton. Semantic hashing. In *Proceedings of the 2007 Workshop on Information Retrieval and applications of Graphical Models (SIGIR'07)*, Amsterdam, 2007. Elsevier.

Rossano Schifanella, Alain Barrat, Ciro Cattuto, Benjamin Markines, and Filippo Menczer. Folks in folksonomies: Social link prediction from shared metadata. In *Proceedings of the Third International Conference on Web Search and Data Mining (WSDM'10)*, pages 271–280. ACM, Mar 2010.

Tanya Schmah, Geoffrey E. Hinton, Richard Zemel, Steven L. Small, and Stephen Strother. Generative versus discriminative training of RBMs for classification of fMRI images. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Leon Bottou, editors, *Advances in Neural Information Processing Systems 21 (NIPS'08)*, pages 1409–1416. Curran Associates, 2009.

Tijmen Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 1064–1071. ACM, 2008.

Tijmen Tieleman and Geoffrey Hinton. Using fast weights to improve persistent contrastive divergence. In Léon Bottou and Michael Littman, editors, *Proceedings of the Twenty-sixth International Conference on Machine Learning (ICML'09)*, pages 1033–1040. ACM, 2009. ISBN 978-1-60558-516-1. doi: http://doi.acm.org/10.1145/1553374.1553506.

Laurens van der Maaten, Max Welling, and Lawrence K. Saul. Hidden-unit conditional random fields. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS'11)*, volume 15 of JMLR: W&CP, 2011.

Yair Weiss. Comparing the mean field method and belief propagation for approximate inference in MRFs. In *Advanced Mean Field Methods - Theory and Practice*. MIT Press, 2001.

Max Welling and Geoffrey E. Hinton. A new learning algorithm for mean field Boltzmann machines. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN'02)*, pages 351–357, London, UK, 2002. Springer-Verlag. ISBN 3-540-44074-7.

Max Welling and Charles Sutton. Learning in markov random fields with contrastive free energies. In *In Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics (AISTATS'05)*, pages 397–404, 2005.

Max Welling, Michal Rosen-Zvi, and Geoffrey E. Hinton. Exponential family harmoniums with an application to information retrieval. In L.K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17 (NIPS'04)*, volume 17, Cambridge, MA, 2005. MIT Press.

Eric P. Xing, Rong Yan, and Alexander G. Hauptmann. Mining associated text and images with dual-wing harmoniums. In *Proceedings of the Twenty-first Conference in Uncertainty in Artificial Intelligence (UAI'05)*, pages 633–641. AUAI Press, 2005. ISBN 0-9749039-1-4.

Jun Yang, Yan Liu, Eric P. Xing, and Alexander G. Hauptmann. Harmonium models for semantic video representation and classification. In *Proceedings of the Seventh SIAM International Conference on Data Mining (SDM'07)*. SIAM, 2007.

Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In Tom Fawcett and Nina Mishra, editors, *Proceedings of the Twenty International Conference on Machine Learning (ICML'03)*, pages 912–919. AAAI Press, 2003.