Statistics and Its Interface Volume 12 (2019) 377–385

Accelerate training of restricted Boltzmann machines via iterative conditional maximum likelihood estimation

Mingqi Wu, Ye Luo, and Faming Liang*

Restricted Boltzmann machines (RBMs) have become a popular tool of feature coding or extraction for unsupervised learning in recent years. However, there still lacks an efficient algorithm for training the RBM due to that its likelihood function contains an intractable normalizing constant. The existing algorithms, such as contrastive divergence and its variants, approximate the gradient of the likelihood function using Markov chain Monte Carlo. However, the approximation is time consuming and, moreover, the approximation error often impedes the convergence of the training algorithm. This paper proposes a fast algorithm for training RBMs by treating the hidden states as missing data and then estimating the parameters of the RBM via an iterative conditional maximum likelihood estimation approach, which avoids the issue of intractable normalizing constants. The numerical results indicate that the proposed algorithm can provide a drastic improvement over the contrastive divergence algorithm in RBM training. This paper also presents an extension of the proposed algorithm for how to cope with missing data in RBM training and illustrates its application using an example about drug-target interaction prediction.

AMS 2000 subject classifications: Primary 62G99; secondary 62P99.

Keywords and phrases: Collaborative filtering, Imputation-regularized optimization algorithm, Missing data, Stochastic EM algorithm.

1. INTRODUCTION

During the past decade, the restricted Boltzmann machine (RBM) has received much attention as a feature coding or extraction tool for unsupervised learning, and a basic building block for deep belief networks as well. [1, 2]. The variants and extensions of the RBM have been applied in a wide range of pattern recognition problems, such as handwriting recognition [1], document processing [3, 5], and collaborative filtering [6]. Despite great successes, there still lacks an efficient algorithm for training RBMs. The existing algorithms aim to maximize the log-likelihood function of the RBM using a gradient-based method, while the true gradient of the log-likelihood function is not available as the likelihood function contains an intractable normalizing constant. In [7], the Contrastive Divergence (CD) algorithm was proposed to train RBMs, where the log-likelihood gradient is approximated using Markov chain Monte Carlo (MCMC) at each iteration. Due to the approximation errors, the CD algorithm does not necessarily converge to the maximum likelihood estimate (MLE) of the parameters as noted in [8] and [9]. Other authors, such as [10], observed that the approximation errors can even lead to a distortion of the learning process; that is, after some iterations the likelihood can start to diverge in the sense that the model systematically get worse if the run of MCMC is not long enough. To address the issue of convergence, some variants of the CD algorithm have been proposed with a general strategy to obtain better approximation of the log-likelihood gradient by sampling from a Markov chain with a greater mixing rate. These variants include persistent CD [11], fast persistent CD [12], tempered transitions [13], and parallel tempering [14, 15]. However, as pointed out by [16], most of these variants come with a variety of hyperparameters in addition to the more common heuristics of weight-decay, momentum and learning rate schedules, and it is unclear how to set the hyperparameters and which heuristic to choose because exact evaluation of the log-likelihood function is infeasible for even a middle-sized RBM.

In this paper, we propose a fast algorithm for training RBMs by treating the hidden states as missing data and then estimating the parameters of the RBM via an iterative conditional maximum likelihood estimation approach, which avoids the issue of intractable normalizing constants. The proposed algorithm works under the framework of the imputation-regularized optimization (IRO) algorithm [29]. The IRO algorithm, as an extension of the stochastic EM algorithm [17, 18], was originally proposed for dealing with high-dimensional missing data problems. It works by iterating between an imputation step and a regularized optimization step. At the imputation step, the missing data are imputed conditional on the observed data and the current estimate of parameters; and at the regularized optimiza-

*To whom the correspondence should be addressed.

tion step, a pseudo-consistent estimate of parameters is obtained by maximizing a penalized log-likelihood function of the pseudo-complete data. Under quite general conditions, it is shown that the average of the pseudo-consistent estimates is consistent to the true parameter when the number of iterations is sufficiently large and the data sample size is sufficiently large. However, the IRO algorithm cannot be directly applied to train RBMs, as for which the likelihood function of the pseudo-complete data contains an intractable normalizing constant.

To get around this issue, we propose to estimate the parameters of the RBM using the conditional maximum likelihood estimation approach [19] at each iteration of the IRO algorithm, observing that the RBM belongs to an exponential family and the conditional maximum likelihood estimator converges to the respective true parameters almost surely. For the RBM, finding the conditional maximum likelihood estimate for the connection weights can be reduced to solving a sequence of logistic regressions. To further accelerate computation, we propose to solve the logistic regressions using the coordinate descent algorithm [20, 21]. By employing an appropriate penalty term for the logistic regressions, such as those encouraging model sparsity, the proposed algorithm provides a simple way to "drop out" redundant connections for the RBM. The numerical results indicate that the proposed algorithm can make a drastic improvement over the CD algorithm in RBM training. We also present an extension of the proposed algorithm for how to accommodate missing visible data in RBM training, and apply the extended algorithm to drug-target interaction predictions. The numerical results indicate a great success of the extended algorithm over the traditional single value decomposition (SVD) method for this problem.

2. ITERATIVE CONDITIONAL MAXIMUM LIKELIHOOD ESTIMATION FOR RBM TRAINING

In this section, we first give a brief review for RBMs and the IRO algorithm, and then describe the proposed algorithm.

2.1 Restricted Boltzmann machine

A RBM is a bipartite undirected graphical model, as shown in Figure 1, which can be used to learn a probability distribution over its set of inputs. Suppose that it has M visible units $\boldsymbol{v} = (v_1, v_2, \ldots, v_M)$ and N hidden units $\boldsymbol{h} = (h_1, h_2, \ldots, h_N)$, and consists of a $N \times M$-matrix of weights $\boldsymbol{W} = (w_{ij})$ associated with the connections between the hidden and visible units, as well as the bias weights $\boldsymbol{b} = (b_1, b_2, \ldots, b_M)$ for the visible units and $\boldsymbol{c} = (c_1, c_2, \ldots, c_N)$ for the hidden units. For the time being, we assume that the RBM is binary-binary, i.e., both the visible and hidden units take binary values. Extension of the proposed algorithm to other types of RBMs will be discussed later. For the binary-binary RBM, the joint distribution of $(\boldsymbol{v}, \boldsymbol{h})$ is given by the Gibbs distribution

$$P_{\boldsymbol{\theta}}(\boldsymbol{v}, \boldsymbol{h}) = \frac{1}{Z(\boldsymbol{\theta})} e^{-E_{\boldsymbol{\theta}}(\boldsymbol{v}, \boldsymbol{h})}, \tag{1}$$

where $\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c}\}$ denotes the set of parameters, $Z(\boldsymbol{\theta})$ is the normalizing constant function defined as the sum of $e^{-E_{\boldsymbol{\theta}}(\boldsymbol{v}, \boldsymbol{h})}$ over all possible configurations of $(\boldsymbol{v}, \boldsymbol{h})$, and $E_{\boldsymbol{\theta}}(\boldsymbol{v}, \boldsymbol{h})$ is the energy function given by

$$E_{\boldsymbol{\theta}}(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{i=1}^{N}\sum_{j=1}^{M} w_{ij} h_i v_j - \sum_{j=1}^{M} b_j v_j - \sum_{i=1}^{N} c_i h_i.$$

Since there are no intra-layer connections in the RBM, the v_j's are mutually independent conditional on $\boldsymbol{h}$ and conversely, the h_i's are mutually independent conditional on $\boldsymbol{v}$. That is,

$$P_{\boldsymbol{\theta}}(\boldsymbol{v}|\boldsymbol{h}) = \prod_{j=1}^{M} f_{\boldsymbol{\theta}_j}(v_j|\boldsymbol{h}), \quad P_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{v}) = \prod_{i=1}^{N} f_{\tilde{\boldsymbol{\theta}}_i}(h_i|\boldsymbol{v}), \tag{2}$$

where $\boldsymbol{\theta}_j = \{b_j, w_{ij} : i = 1, \ldots, N\}$ and $\tilde{\boldsymbol{\theta}}_i = \{c_i, w_{ij} : j = 1, \ldots, M\}$ denote the subsets of parameters of respective conditional distributions, and

$$\begin{aligned} f_{\boldsymbol{\theta}_j}(v_j = 1|\boldsymbol{h}) &= \sigma(b_j + \sum_{i=1}^{N} w_{ij} h_i), \\ f_{\tilde{\boldsymbol{\theta}}_i}(h_i = 1|\boldsymbol{v}) &= \sigma(c_i + \sum_{j=1}^{M} w_{ij} v_j), \end{aligned} \tag{3}$$

where $\sigma(\cdot)$ denotes the logistic sigmoid, i.e., $\sigma(z) = 1/(1 + e^{-z})$.

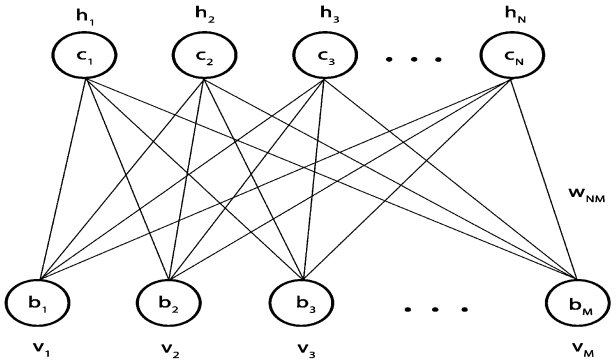

Figure 1. An illustrative graph of RBM with M visible units (bottom row) and N hidden units (upper row).

2.2 The imputation-regularized optimization algorithm

Missing data are ubiquitous throughout almost all fields of science and technology. An inappropriate treatment of missing data can lead to a significant loss of data information and/or a biased statistical inference. For low-dimensional problems, the MLE of the parameters can be searched using the EM algorithm [22] or its variants, such as Monte Carlo EM [23], ECM [24], and stochastic EM [17, 18]. However, for high-dimensional problems, where the data dimension is greater than the sample size, the EM algorithm and its variants often fail to work. Although some problem-specific algorithms have been developed, see e.g. [25, 26, 27], there still lacks a general algorithm. The IRO algorithm [29] fills this gap: In principle, it can be applied to any missing data problems, regardless of dimension and distribution of the data. The IRO algorithm can be described as follows.

Let $X_1, \ldots, X_n$ denote a set of independent and identically distributed samples drawn from the distribution $f_{\boldsymbol{\theta}}(x)$, where n is the sample size, and $\boldsymbol{\theta}$ is a vector of parameters. Let $X_i = (X_i^{\text{obs}}, X_i^{\text{mis}})$, where X_i^{obs} is observed and X_i^{mis} is missed. Let $\boldsymbol{X} = (X_1, \ldots, X_n)$, $\boldsymbol{X}^{\text{obs}} = (X_1^{\text{obs}}, \ldots, X_n^{\text{obs}})$, and $\boldsymbol{X}^{\text{mis}} = (X_1^{\text{mis}}, \ldots, X_n^{\text{mis}})$. To indicate the dependence of the dimension of $\boldsymbol{\theta}$ on the sample size n, we write $\boldsymbol{\theta}$ as $\boldsymbol{\theta}_n$ and denote by $\boldsymbol{\theta}_n^{(t)}$ the estimate of $\boldsymbol{\theta}$ obtained at the t^{th} iteration of the IC algorithm. The IRO algorithm starts with an initial guess $\boldsymbol{\theta}_n^{(0)}$ and then iterates between the imputation and the regularized optimization steps:

- **I-step**: Draw $\tilde{\boldsymbol{X}}^{\text{mis}}$ from the predictive distribution $h(\boldsymbol{x}^{\text{mis}}|\boldsymbol{X}^{\text{obs}}, \boldsymbol{\theta}_n^{(t)})$ conditioned on $\boldsymbol{X}^{\text{obs}}$ and $\boldsymbol{\theta}_n^{(t)}$.
- **RO-step**: Based on the pseudo-complete data $\tilde{\boldsymbol{X}} = (\boldsymbol{X}^{\text{obs}}, \tilde{\boldsymbol{X}}^{\text{mis}})$, find an updated estimate $\boldsymbol{\theta}_n^{(t+1)}$ which forms a consistent estimate of

$$\boldsymbol{\theta}_*^{(t)} = \arg\max_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}_n^{(t)}} \log f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}), \tag{4}$$

  where $E_{\boldsymbol{\theta}_n^{(t)}} \log f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}) = \int \log f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}) f(\boldsymbol{x}^{\text{obs}}|\boldsymbol{\theta}^*) h(\tilde{\boldsymbol{x}}^{\text{mis}}|\boldsymbol{x}^{\text{obs}}, \boldsymbol{\theta}_n^{(t)}) d\boldsymbol{x}^{\text{obs}} d\tilde{\boldsymbol{x}}^{\text{mis}}$, $\boldsymbol{\theta}^*$ denotes the true value of the parameters, and $f(\boldsymbol{x}^{\text{obs}}|\boldsymbol{\theta}^*)$ denotes the marginal density function of $\boldsymbol{x}^{\text{obs}}$.

To compute $\boldsymbol{\theta}_n^{(t+1)}$, [29] suggested a regularization approach, i.e., setting

$$\boldsymbol{\theta}_n^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \left[\sum_{i=1}^{n} f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}) - \lambda P(\boldsymbol{\theta}) \right], \tag{5}$$

where $P(\boldsymbol{\theta})$ denotes a penalty function, and λ is the regularization parameter. As discussed in [29], the regularization in (5) should be interpreted in a general sense. For low-dimensional problems, one can simply set $\lambda = 0$. For high-dimensional problems, one can choose an appropriate penalty function that enforces the sparsity of $\boldsymbol{\theta}$. It is interesting to point out that such a regularization estimator also includes the sure screening estimator [30, 31] as a special case for which the penalty function is of binary type, taking a value of zero in the desired subspace and infinity otherwise.

Based on the theory of empirical process [32], [29] showed that $\boldsymbol{\theta}_n^{(t+1)}$ obtained through the regularization approach is a consistent estimate of $\boldsymbol{\theta}_*^{(t)}$ under some regularity conditions, such as n is sufficiently large, the dimension of $\boldsymbol{\theta}$ grows at a rate of $O(n^\alpha)$ for some constant $0 < \alpha < \infty$, appropriate metric entropy conditions, and $\log f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})$ is well behaved with $|\log f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})|$ being uniformly bounded by an integrable function and the distribution of $[\log f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}) - \int \log f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}) h(\tilde{\boldsymbol{x}}^{\text{mis}}|\boldsymbol{x}^{\text{obs}}, \boldsymbol{\theta}_n^{(t)}) d\tilde{\boldsymbol{x}}^{\text{mis}}]$ having a sub-exponential tail. The later is related to the conditions for imputed data, while the metric entropy conditions are related to the sparsity conditions imposed on $\boldsymbol{\theta}$ for high-dimensional problems. Refer to Theorem 1 of [29] for the detail. Similar to the stochastic EM algorithm, $\{\boldsymbol{\theta}_n^{(t)}\}$ produced by the IRO algorithm forms a Markov chain which converges to a stationary distribution. Further, by assuming that the mapping $M(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}'} E_{\boldsymbol{\theta}} g(\boldsymbol{\theta}', \tilde{\boldsymbol{x}})$ satisfies a contraction condition, [29] proved that $\boldsymbol{\theta}_n^{(t+1)}$ will converge to the true parameter $\boldsymbol{\theta}^*$ in probability when both the sample size $n \to \infty$ and the iteration number $t \to \infty$, and that the average of $\{\boldsymbol{\theta}_n^{(t)}\}$ over t also converges to $\boldsymbol{\theta}^*$ in probability. Refer to Theorem 4 of [29] for the detail.

The IRO algorithm is attractive only when the consistent estimate of $\boldsymbol{\theta}_*^{(t)}$ can be easily obtained at each RO-step. For many problems, similar to the ECM algorithm [24], $\boldsymbol{\theta}_n^{(t)}$ can be easily obtained with a number of conditional consistency steps. That is, one can partition $\boldsymbol{\theta}$ into a number of blocks and then find a consistent estimate for each block conditional on the current estimates of other blocks. Suppose that $\boldsymbol{\theta} = (\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(k)})$ has been partitioned into k blocks. The RO-step of the IRO algorithm can be replaced by the following conditional regularized-optimization (CRO) step:

- **CRO-step**. Based on the pseudo-complete data $\tilde{\boldsymbol{X}} = (\boldsymbol{X}^{\text{obs}}, \tilde{\boldsymbol{X}}^{\text{mis}})$, do the following:
  (1) Fixed on $(\boldsymbol{\theta}_n^{(t,2)}, \ldots, \boldsymbol{\theta}_n^{(t,k)})$, find $\boldsymbol{\theta}_n^{(t+1,1)}$ which forms a consistent estimate of

$$\boldsymbol{\theta}_*^{(t,1)} = \arg\max_{\boldsymbol{\theta}^{(t,1)'}} E_{\boldsymbol{\theta}_n^{(t,1)}, \ldots, \boldsymbol{\theta}_n^{(t,k)}} \log f(\tilde{\boldsymbol{x}}|\boldsymbol{\theta}_n^{(t,1)'}, \boldsymbol{\theta}_n^{(t,2)}, \ldots, \boldsymbol{\theta}_n^{(t,k)}),$$

  where the expectation $E(\cdot)$ is taken with respect to the joint distribution of $\tilde{\boldsymbol{x}} = (\boldsymbol{x}^{\text{obs}}, \boldsymbol{x}^{\text{mis}})$ and its subscript indicates the current estimate of $\boldsymbol{\theta}$.
  (2) Fixed on $(\boldsymbol{\theta}_n^{(t+1,1)}, \boldsymbol{\theta}_n^{(t,3)}, \ldots, \boldsymbol{\theta}_n^{(t,k)})$, find $\boldsymbol{\theta}_n^{(t+1,2)}$ which forms a consistent estimate of

$$\boldsymbol{\theta}_*^{(t,2)} = \arg\max_{\boldsymbol{\theta}^{(t,2)'}} E_{\boldsymbol{\theta}_n^{(t+1,1)}, \boldsymbol{\theta}_n^{(t,2)}, \boldsymbol{\theta}_n^{(t,3)}, \ldots, \boldsymbol{\theta}_n^{(t,k)}} \log f(\tilde{\boldsymbol{x}}| \boldsymbol{\theta}_n^{(t+1,1)}, \boldsymbol{\theta}_n^{(t,2)'}, \boldsymbol{\theta}_n^{(t,3)}, \ldots, \boldsymbol{\theta}_n^{(t,k)}).$$

.....

(k) Fixed on $(\boldsymbol{\theta}_n^{(t+1,1)}, \ldots, \boldsymbol{\theta}_n^{(t+1,k-1)})$, find $\boldsymbol{\theta}_n^{(t+1,k)}$ which forms a consistent estimate of

$$\boldsymbol{\theta}_*^{(t,k)} = \arg\max_{\boldsymbol{\theta}^{(t,k)'}} E_{\boldsymbol{\theta}_n^{(t+1,1)},\ldots,\boldsymbol{\theta}_n^{(t+1,k-1)},\boldsymbol{\theta}_n^{(t,k)}} \log f(\tilde{\boldsymbol{x}}| \boldsymbol{\theta}_n^{(t+1,1)}, \ldots, \boldsymbol{\theta}_n^{(t+1,k-1)}, \boldsymbol{\theta}_n^{(t,k)'}).$$

It is easy to see that the estimate sequence $\boldsymbol{\theta}_n^{(t)} = \{(\boldsymbol{\theta}_n^{(t,1)}, \ldots, \boldsymbol{\theta}_n^{(t,k)})\}$ forms a Markov chain. Under similar conditions, [29] proved that the ICRO algorithm shares the same theoretical properties with the IRO algorithm; that is, both $\boldsymbol{\theta}_n^{(t)}$ and its path average form a consistent estimate of $\boldsymbol{\theta}^*$ when both t and n are sufficiently large.

2.3 Iterative conditional maximum likelihood estimation for RBM training

For the RBM, with a slight abuse of notation, we let $\boldsymbol{v} = (v_1, v_2, \ldots, v_M)$ denote a generic observation on the visible units, and let $\boldsymbol{h} = (h_1, h_2, \ldots, h_N)$ denote the hidden values corresponding to $\boldsymbol{v}$. Further, we let $\boldsymbol{v}_k = (v_{1k}, v_{2k}, \ldots, v_{Mk})$ denote the kth observation on the visible units, let n denote the sample size, and assume that the samples $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$ are independent and identically distributed. The task of RBM training is to find a set of estimates for the parameters $\boldsymbol{\theta}$ in (1) given the samples $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$. By treating $\boldsymbol{v}$ as observed data and $\boldsymbol{h}$ as missing data, it is natural to apply the IRO/ICRO algorithm to estimate $\boldsymbol{\theta}$. However, since the normalizing constant function $Z(\boldsymbol{\theta})$ is intractable, it is hard to find an estimate for $\boldsymbol{\theta}$ or a component of $\boldsymbol{\theta}$ by directly maximizing a regularized log-likelihood function of the pseudo-complete data. To address this issue, we propose an iterative conditional maximum likelihood estimation (ICMLE) algorithm, which can be described as follows.

Since the joint distribution of $\boldsymbol{v}$ and $\boldsymbol{h}$ belongs to the exponential family, $\{(h_i v_j), \boldsymbol{v}, \boldsymbol{h}\}$ forms a complete statistic for $\boldsymbol{\theta} = (\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c})$, where $(h_i v_j)$ denotes a matrix with $i = 1, \ldots, N$ and $j = 1, \ldots, M$. In addition, the joint distribution of $\boldsymbol{v}$ and $\boldsymbol{h}$ can be factored as

$$P_{\boldsymbol{\theta}}(\boldsymbol{v}, \boldsymbol{h}) = P_c(\boldsymbol{v}|\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{h}) P_r(\boldsymbol{h}|\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c}), \tag{6}$$

where P_c stands for the conditional likelihood function of $\boldsymbol{v}$ given $\boldsymbol{h}$ and it is free from $\boldsymbol{c}$, and P_r stands for the residual likelihood for $\boldsymbol{h}$. Traditionally, in the factorization of (6), $(\boldsymbol{W}, \boldsymbol{b})$ is called structural parameters and $\boldsymbol{c}$ is called nuisance or incidental parameters. According to the theory developed in [19], for which the conditions are satisfied by the exponential family, the structural parameters can be estimated by maximizing the conditional likelihood function and such an estimator converges almost surely to the true value of the structural parameters. By (3), the conditional likelihood function can be further factored as the product of a sequence of sigmoid functions, i.e.,

$$P_c(\boldsymbol{v}|\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{h}) = \prod_{j=1}^{M} \sigma(b_j + \sum_{i=1}^{N} w_{ij} h_i).$$

Therefore, conditioned on $\boldsymbol{h}$, $(\boldsymbol{W}, \boldsymbol{b})$ can be estimated by solving a sequence of logistic regressions in parallel. Given the estimate of $(\boldsymbol{W}, \boldsymbol{b})$, $\boldsymbol{c}$ can be further estimated via a conditional maximization step under the framework provided by the ICRO algorithm.

To have a more precise description for the ICMLE algorithm, we let $(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_M, \boldsymbol{c})$ denote a partition of $\boldsymbol{\theta} = (\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c})$, where $\boldsymbol{\theta}_i$'s and $\boldsymbol{c}$ are as defined in Section 2.1. Let $\boldsymbol{h}^{(t+1)}$ denote the imputed values of $\boldsymbol{h}$ given $\boldsymbol{\theta}^{(t)}$ and $\boldsymbol{v}$, where t indexes the iteration of the ICRO algorithm. Then, each $\boldsymbol{\theta}_j^{(t+1)}$ can be calculated by solving a logistic regression, for which v_j works as the response variable and $\boldsymbol{h}^{(t+1)}$ works as the predictors. To enforce the sparsity for each $\boldsymbol{\theta}_j$, we further suggest a regularization approach which is to set

$$\boldsymbol{\theta}_j^{(t+1)} = \arg\max_{\boldsymbol{\theta}_j} \left[\sum_{k=1}^{n} \log f_{\boldsymbol{\theta}_j}(v_{jk}|\boldsymbol{h}_k^{(t+1)}) - \lambda P(\boldsymbol{\theta}_j) \right], \tag{7}$$

where v_{jk} denotes the kth observation of the visible unit j, and $\boldsymbol{h}_k^{(t+1)}$ denotes the subset of the elements of $\boldsymbol{h}^{(t+1)}$ corresponding to the kth observation. This provides a simple way to drop out redundant connections and is expected to improve the generalization ability of the RBM. Refer to [4] for more discussions on "dropout" methods. For (7), a variety of penalty functions can be used, such as those used in Lasso [33], elastic net [34], SCAD [35] and MCP [36], while ensuring the consistency of the resulting estimator. In this paper, we used the L_1-penalty in all simulations and set the regularization parameter $\lambda \equiv 10^{-5}$. Practically, the value of λ can be determined using a cross-validation approach. For a given value of λ, (7) can be solved using the coordinate descent algorithm [20].

Conditioned on the updated estimates $\boldsymbol{W}^{(t+1)}$ and $\boldsymbol{b}^{(t+1)}$, i.e., the collection of $\boldsymbol{\theta}_j^{(t+1)}$'s, the parameters in $\boldsymbol{c}$ can be estimated by maximizing the following conditional likelihood function, i.e., setting

$$c_i^{(t+1)} = \arg\max_{c_i} \sum_{k=1}^{n} \log f_{\tilde{\boldsymbol{\theta}}_i}(h_{ik}|\boldsymbol{v}_k,), \quad i = 1, \ldots, N, \tag{8}$$

under the framework provided by the ICRO algorithm. Each optimization problem in (8) is one-dimensional and can be easily solved by a root-finding algorithm, say, Brent's algorithm [37, 38] based on its gradient. The rationale underlying (8) can be explained based on an alternative factorization for the joint distribution of $\boldsymbol{v}$ and $\boldsymbol{h}$:

$$P_{\boldsymbol{\theta}}(\boldsymbol{v}, \boldsymbol{h}) = P_c'(\boldsymbol{h}|\boldsymbol{W}, \boldsymbol{c}, \boldsymbol{v}) P_r'(\boldsymbol{v}|\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c}), \tag{9}$$

where P'_c stands for the conditional likelihood function of $\boldsymbol{h}$ given $\boldsymbol{v}$ and it is free from $\boldsymbol{b}$, and P'_r stands for the residual likelihood for $\boldsymbol{v}$. Following from the theory by [19], $\boldsymbol{c}$ can be estimated via (8) when the estimate of $\boldsymbol{W}$ are given.

In summary, we have the following algorithm for RBM training:

- **I-step**. Draw $\boldsymbol{h}^{(t+1)}$ from the distribution $P_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{v})$ as defined in (2) and (3), conditioned on $\boldsymbol{v}$ and the current estimate $\boldsymbol{\theta}^{(t)} = (\boldsymbol{\theta}_1^{(t)}, \ldots, \boldsymbol{\theta}_M^{(t)}, \boldsymbol{c}^{(t)})$.
- **CMLE-step**. Based on the pseudo-complete data $(\boldsymbol{h}^{(t+1)}, \boldsymbol{v})$, do the following:
  - (i) For $j = 1, 2, \ldots, M$ (in parallel), calculate $\boldsymbol{\theta}_j^{(t+1)}$ according to (7) by solving a penalized logistic regression using the coordinate descent algorithm.
  - (ii) Fixed on $\{\boldsymbol{\theta}_1^{(t+1)}, \ldots, \boldsymbol{\theta}_M^{(t+1)}\}$, for $i = 1, 2, \ldots, N$ (in parallel), calculate $c_i^{(t+1)}$ according to (8) using a one-dimensional root-finding algorithm.

For the coordinate descent algorithm, we suggest to pass on the current estimate $\boldsymbol{\theta}_j^{(t)}$ to the next iteration as the initial guess in calculating $\boldsymbol{\theta}_j^{(t+1)}$, and this can substantially accelerate the convergence of the algorithm. In addition, $\boldsymbol{\theta}_j^{(t+1)}$'s can be calculated in parallel (with respect to j), and $c_i^{(t+1)}$'s can also be calculated in parallel (with respect to i). The validity of the algorithm follows from the standard theory of the ICRO algorithm and the conditional maximum likelihood estimation: $\boldsymbol{\theta}_n^{(t)}$ will converge to $\boldsymbol{\theta}^*$ in probability as the sample size $n \to \infty$ and the iteration number $t \to \infty$. Through working on conditional likelihood functions, the proposed algorithm gets around the intractable normalizing constant problem in parameter estimation for RBMs.

For the RBM, however, as pointed out in [28], it suffers from the parameter identifiability issue; not only it is possible to approximate any distribution on the visibles arbitrarily well, but quite different parameter settings can induce the same essential RBM model. Therefore, in general, the ICMLE algorithm will converge to a solution close to the starting point given the iterative nature of the coordinate descent algorithm we employed and the parameter identifiability issue of the RBM.

3. NUMERICAL STUDIES

3.0.0.1. Example 1 This example, taken from the R package *deepnet* [39], is used to test the validity of the ICMLE algorithm for RBM training. For this example, the RBM is used as a feature coding tool for unsupervised learning. The RBM consists of $M = 4$ visible units and $N = 2$ hidden units. The input data consists of 200 observations, with 50 observations for each of the four patterns (1,0,1,0), (0,1,1,0), (1,0,0,1), and (0,1,0,1). We are interested in this example as for which it is known that the input patterns can be coded as (1,1), (0,1), (1,0) and (0,0) (up to permutations) on the two hidden units when the RBM is well trained. Therefore, it provides a simple test for the validity of the proposed training algorithm.

The ICMLE algorithm was applied to this example, with each component of $\boldsymbol{\theta}$ initialized by a random variable drawn from the Gaussian distribution $N(0, 0.1^2)$. To measure the convergence of the algorithm, we calculated the reconstruction error, which is defined as the sum of the squared difference between the visible values and their "reconstructed values", i.e.,

$$\sum_{k=1}^{n}\sum_{j=1}^{M}\left[v_{jk} - \sigma\left(b_j + \sum_{i=1}^{N} w_{ij}\sigma\Big(c_i + \sum_{l=1}^{M} w_{il}v_{lk}\Big)\right)\right]^2, \tag{10}$$

where $\sigma(z) = 1/(1+e^{-z})$ denotes the logistic sigmoid. As mentioned previously, the RBM model suffers from the parameter identifiability issue. Therefore, for a RBM model, we are usually not interested in the accuracy of parameter estimation but the reconstruction error which measures the quality of the RBM in feature coding or extraction. Issues on convergence diagnostic for the ICMLE algorithm will be further discussed at the end of the paper.

Figure 2 shows the convergence paths of the reconstruction error in 10 independent runs of ICMLE. Each run consisted of 20 iterations and cost about 3.4 seconds CPU time on a T7610 workstation of 3.6GHz. All computations reported in this paper were done on the same computer. If the code was executed in parallel on the workstation (under the OpenMP platform with 45 thresholds), the real time cost by each run was only about 0.2 seconds. We have checked the values of the hidden units obtained in each run, they all converged to one permutation of the four patterns (1,1), (0,1), (1,0) and (0,0). This indicates the validity of the ICMLE algorithm for RBM training. As shown by Figure 2, the algorithm can converge very fast. In some runs, it can converge within less than 10 iterations.

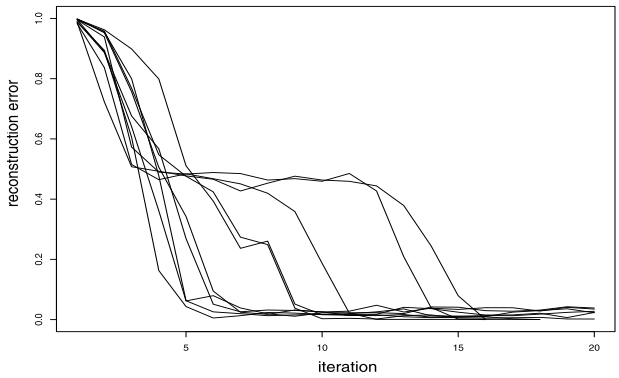

Figure 2. Example 1: Convergence paths (10 independent runs) of the ICMLE algorithm for learning a RBM with 2 hidden units.

3.0.0.2. Example 2 This example is to train a RBM for a wheat line dataset. The dataset consists of $n = 599$ wheat lines, which are treated as independent and identically distributed samples, and each line is genotyped with 1279 DArT markers (Diversity Array Technology). The DArT markers take binary values, denoted by their presence or absence. In the dataset, the overall mean frequency of the allele coded as '1' is 0.561, with a minimum of 0.008 and a maximum of 0.987. The dataset originally came from the International Maize and Wheat improvement center, and it can be downloaded from R package BLR [40].

This is a very difficult example for RBMs, as the data does not contain obvious patterns. We fitted the data by a RBM with 300 hidden units, which consists of 385,279 (=1279*300+1279+300) parameters. Training a RBM with such a large number of parameters is a challenging task for any gradient-based methods, including the contrastive divergence (CD) algorithm. However, the ICMLE algorithm works extremely well for this example. Figure 3 shows the convergence path of the algorithm. Only after 5 iterations, the reconstruction error has been reduced to nearly zero. This implies that for this example, the 1279 DArT markers can be coded on 300 hidden units and thus the RBM provides a good feature coding or dimension reduction tool (reduced from 1279 to 300). The entire run consisted of 20 iterations and cost about 12 minutes in real time on the T7610 workstation with 45 threads running in parallel. The total CPU time was about 467 minutes. Multiple runs have been tried and the convergence path of each run is very similar to that shown in Figure 3.

For comparison, we also applied the CD-10 algorithm to this example, where 10 is the number of Gibbs iterations performed at each iteration of the CD algorithm for evaluating the likelihood gradient. The CD algorithm has been implemented in the R package *deepnet* [39]. In our run, we have adjusted the number of iterations such that the algorithm also cost about 467 CPU minutes at the end of the run on the same T7610 workstation. The convergence path of the algorithm is shown in Figure 3. The comparison indicates that ICMLE has made a drastic improvement over the CD algorithm in RBM training. For this example, CD-10 might fail to converge to a global optimal solution or it will take extremely long time to reach the level of reconstruction error achieved by ICMLE in just a few iterations.

4. PREDICTION OF DRUG-TARGET INTERACTIONS

Drug development is known to be expensive and time consuming. Moreover, the success rate is extremely low. Motivated by the polypharmacology property that individual drugs can interact with multiple targets, drug developers often actively seek new uses for existing drugs, where each target refers to a different protein. This is the so-called drug repositioning strategy. Toward prediction of drug-target interactions, numerous work have been published in recent

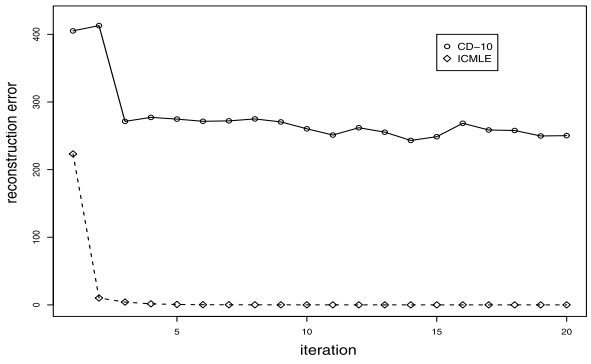

Figure 3. Convergence paths of the ICMLE and CD-10 algorithms for learning a RBM with 300 hidden units for the wheat line dataset.

years, see e.g., [41, 42, 43], and [44]. Among these work, [43] is of particular interest to us, where they formulated the problem as a collaborative filtering problem and applied the RBM to make the prediction. Collaborative filtering has become an important application of RBMs since the publication of the seminal work [6]. To deal with the high proportion of missing values that are often encountered in the rating data, e.g., user's ratings of movies, [6] proposed a population RBM model, which consists of a large number of RBMs. Each RBM has the same number of hidden units, but it can have different numbers of visible units depending on how many ratings the user has made. The weights and biases of the RBMs are tied together; if two users have rated the same movies, the two RBMs must use the same weights between the hidden units and the visible unit for that movie. [6] showed that the population RBM model can be used to handle very large datasets, and it slightly outperforms carefully-tuned SVD models.

[43] adopted the population RBM model to predict drug-target interactions, where each target corresponds to a user and each drug corresponds to a "movie". In this paper we provide a different view to the problem: we view the drug-target interactions to be predicted as missing data. Therefore, only a single RBM is used where each target works as an independent sample, and the missing drug-target interactions can be imputed iteratively in training the RBM. Compared to the population RBM model, this model is much simpler. Let $\boldsymbol{v}^{\text{mis}}$ and $\boldsymbol{v}^{\text{obs}}$ denote the missing and observed parts of the visible data, respectively. Let $\boldsymbol{v}^{\text{mis}}_{(t)}$ denote the imputed value of $\boldsymbol{v}^{\text{mis}}$ at iteration t. In summary, we have the following extended ICMLE algorithm to train RBMs with missing data:

- **I-step**. Impute missing data and hidden units:
  - (i) Draw $\boldsymbol{v}^{\text{mis}}_{(t+1)}$ conditioned on $\boldsymbol{h}^{(t)}$ and the current parameter estimate $\boldsymbol{\theta}^{(t)}$.
  - (ii) Draw $\boldsymbol{h}^{(t+1)}$ from the distribution $P_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{v})$ as defined in (2) and (3), conditioned on $\boldsymbol{\theta}^{(t)}$ and $\boldsymbol{v}^{\text{mis}}_{(t+1)}$.

- **CMLE-step**. Based on the pseudo-complete data $(\boldsymbol{h}^{(t+1)}, \boldsymbol{v}^{\text{obs}}, \boldsymbol{v}^{\text{mis}}_{(t+1)})$, do the following:

  (i) For $j = 1, 2, \ldots, M$ (in parallel), calculate $\boldsymbol{\theta}_j^{(t+1)}$ according to (7) by solving a penalized logistic regression using the coordinate descent algorithm.

  (ii) Fixed on $\{\boldsymbol{\theta}_1^{(t+1)}, \ldots, \boldsymbol{\theta}_M^{(t+1)}\}$, for $i = 1, 2, \ldots, N$ (in parallel), calculate $c_i^{(t+1)}$ according to (8) using a one-dimensional root-finding algorithm.

We tested the proposed algorithm on MATADOR [45]. which is a manually curated online database of drug-target interactions. The dataset contains 2860 protein targets, 790 drugs, and 14,964 interactions including both direct and indirect ones. We arranged the dataset into a 2860×790 binary matrix; that is, for this example, we trained a RBM with 790 visible units using $n = 2860$ independent samples. The training dataset is very sparse with the proportion of 1's, indicating presence of interactions, being only 0.66%. With such sparse signals, accurate prediction of the drug-target interactions is extremely difficult.

To test the proposed algorithm, we first randomly selected 360 rows (protein targets) and then randomly deleted 100 elements (drugs) from each of the selected 360 rows as missing data. Through this process, we generated 10 different training datasets. Our goal is to predict the missing 36,000 elements for each of the datasets. We tried a RBM with 75 hidden units for this problem. For each dataset, the proposed algorithm was run for 70 iterations, where the first 20 iterations were discarded for the burn-in process. The expected values of the missing visible units in the remaining 50 iterations were averaged as the predicted values. Each run cost about 35.5 minutes of real time on the T7610 workstation with 45 thresholds running in parallel. The total CPU time is about 897 minutes. To measure the performance of the proposed method, we calculated the area under the precision-recall curve with the precision and recall being defined as

$$\text{precision} = \frac{TP}{TP + FP}, \quad \text{recall} = \frac{TP}{TP + FN},$$

where TP, FP and FN are defined in Table 1 for outcomes of binary decision. Since the drug-target interactions in the dataset are rare, i.e., two classes are imbalanced, the precision-recall curve can therefore provide a better measure than the ROC curve for the performance of different prediction algorithms [46]. As conventional, we summarize the information of the precision-recall curve by a single number, the area under the precision-recall curve (AUPR). The results are summarized in Table 2, which reports the averaged AUPRs over the 10 datasets.

For comparison, we applied the singular value decomposition (SVD) method to this problem. SVD has been popularly used in collaborative filtering, which decomposes a matrix (with missing values) into two components U and V. The singular values have been folded into these matrices. A low-rank approximation for the original matrix can then be obtained based on the decomposition with a specified number of singular values. The SVD method has been implemented in the R package *recommenderlab* [47], which allows the existence of missing values in the matrix decomposition. For this example, we have tried different numbers of singular values with the results summarized in Table 2. The comparison indicates that the RBM-ICMLE method performs much better than the SVD method. The outperformance of the RBM-ICMLE method may be due to the non-linearity of the RBM model. Extension of the RBM-ICMLE method to general collaborative filtering problems is of great interest.

Table 1. Outcome table of binary decision.

Decision	True	False
Positive	True Positive(TP)	False Positive(FP)
Negative	False Negative(FN)	True Negative(TN)

Table 2. Comparison of RBM-ICMLE (with 75 hidden units) and SVD for prediction of drug-target interactions: the results are averaged over 10 datasets. For SVD, the default number of singular values is $k = 10$ and it allows existence of missing data. "AUPR" refers to the averaged area under the precision-recall curve, and "SD" refers to the standard deviation of the averaged area.

		SVD			
	ICMLE	k=2	k=5	K=10*	K=50
AUPR	0.787	0.148	0.171	0.095	0.073
SD	(0.006)	(0.015)	(0.017)	(0.006)	(0.004)

For this example, we have formulated the problem as a missing data problem and employed the RBM model to tackle the associated prediction problem. We have tried to compare the proposed ICMLE algorithm with the CD algorithm on this example. Unfortunately, it is unclear how the CD algorithm can be applied to the problems with missing data. At least, to the best of our knowledge, there is no a public package where the CD algorithm can be run with missing data.

5. DISCUSSION

We have proposed an innovative and fast algorithm for training RBMs by incorporating conditional maximum likelihood estimation into the ICRO algorithm, which gets around the intractable normalizing constant problem encountered by the existing CD algorithm. The numerical results indicate that the proposed algorithm can provide a drastic improvement over the CD algorithm in RBM training. In addition, the proposed algorithm has an automatic mechanism to drop out redundant connections for the RBM.

We also gave an extension of the proposed algorithm for how to cope with missing visible data in RBM training, and illustrated its application using an example on drug-target interaction predictions.

The proposed ICMLE algorithm induces two interleaved Markov chains, one for the parameters of the RBM and the other for the imputed hidden variables. The convergence of the algorithm can then be diagnosed using the methods developed for Markov chains, such as the Gelman-Rubin statistic [48], which is essentially the same as what we did in Figure 2. That is, we can run the algorithm multiple times and diagnose the convergence of these runs by comparing the paths of a summary statistic resulted from each run. The summary statistic can be reconstruction error, which is very easy to compute for RBMs, or some other summary statistics. [49] pointed out that for the CD algorithm, the reconstruction error is a poor measure of training progress as it is not the function that the CD algorithm aims to optimize. However, for the proposed ICMLE algorithm, the reconstruction error can still work as a good summary statistic for assessing the convergence of the algorithm. This is due to that the convergence of the ICML algorithm should be assessed based on multiple runs due to its Markov property, while the convergence of the CD algorithm is assessed based on a single run. In addition, the reconstruction error measures the performance of the RBM model in feature coding and extraction, the major goal of the RBM model.

Regarding computation, we note that the proposed ICMLE algorithm can be further accelerated by working with only a random subset of the full dataset at each iteration, similar to the stochastic gradient algorithm employed in deep learning. The reason why this strategy works is that the ICRO algorithm requires only an estimate of $\boldsymbol{\theta}_*^{(t)}$ obtained at each iteration instead of the exact maximizer of the pseudo-complete data likelihood. However, as expected, this strategy works at a price of high variation of the Markov chains for both the parameter estimates and imputed hidden variables.

In this paper, we considered only the RBMs with the visible units restricted to binary variables. Extension to the cases where the visible units are multinomial or Gaussian is straightforward. For the former, each parameter block $\boldsymbol{\theta}_j$ can be estimated by solving a multiclass logistic regression; and for the latter, each parameter block $\boldsymbol{\theta}_j$ can be estimated by solving a linear regression.

ACKNOWLEDGEMENTS

Liang's research was supported in part by the grants DMS-1612924, DMS/NIGMS R01-GM117597, and NIGMS R01-126089. The authors thank the editor, associate editors and two referees for their helpful comments which have led to significant improvement of this paper.

Received 17 May 2018

REFERENCES

[1] Hinton, G.E. and Salakhutdinov, R.R. (2006). Reducing the dimensionality of data with neural networks. *Science* **313** 504–507. MR2242509

[2] Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning* **2** 1–121.

[3] Dahl, G.E., Adams, R.P., and Larochelle, H. (2012). Training restricted Boltzmann machines on word observations. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)* (ed. J. Langford and J. Pineau), pp.679–686.

[4] Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* **15** 1929–1958. MR3231592

[5] Srivastava, N., Salakhutdinov, R.R., and Hinton, G.E. (2013). Modeling documents with a deep Boltzmann machine. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI 2013)* (ed. A. Nicholson and P. Smyth), pp.616–624. MR3277150

[6] Salakhutdinov, R.R., Mnih, A., and Hinton, G.E. (2007). Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)* (ed. Z. Ghahramani), pp.791-798.

[7] Hinton, G.E., Osindero, S., and Teh, W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation* **18** 1527–1554. MR2224485

[8] Carreira-Perpinán, M.A. and Hinton, G.E. (2005). On contrastive divergence learning. In *The 10th International Workshop on Artificial Intelligence and Statistics (AISTATS)* (ed. R.G. Cowell and Z. Ghahramani), pp.59–66.

[9] Bengio, Y. and Delalleau, O. (2009). Justifying and generalizing contrastive divergence. *Neural Computation* **21** 1601–1621. MR2527797

[10] Fischer, A. and Igel, C. (2010). Empirical analysis of the divergence of Gibbs sampling based learning algorithms for restricted Boltzmann machines. In *International Conference on Artificial Neural Networks (ICANN)* (ed. K. Diamantaras, W. Duch and L.S. Iliadis), pp.208–217.

[11] Tieleman, T. (2008). Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, pp.1064–1071.

[12] Tieleman, T. and Hinton, G. (2009). Using fast weights to improve persistent contrastive divergence. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, pp.1033–1040.

[13] Salakhutdinov, R.R. (2009). Learning in Markov random fields using tempered transitions. In *Advances in Neural Information Processing Systems (NIPS) 22*, pp.1598–1606.

[14] Desjardins, G., Coutville, A., Bengio, Y., Vincent, P., and Dellaleau, O. (2010). Parallel tempering for training of restricted Boltzmann machines. In *Journal of Machine Learning Research Workshop and Conference Proceedings* **9** 145–152.

[15] Cho, K., Raiko, T. and Ilin, A. (2010). Parallel tempering is efficient for learning restricted Boltzmann machines. In *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, pp.3246–3253.

[16] Schulz, H., Müller, A., and Behnke, S. (2010). Investigating convergence of restricted Boltzmann machine learning. In *NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*.

[17] Celeux, G. and Diebolt, J. (1985). The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly* **2** 73–82.

[18] Nielsen, S.F. (2000). The stochastic EM algorithm: Estimation and asymptotic results. *Bernoulli* **6** 457–489. MR1762556

[19] Andersen, E.B. (1970). Asymptotic Properties of Conditional Maximum-Likelihood Estimators. *Journal of the Royal Statistical*

Society, Series B **32** 283–301. MR0273723
[20] Tseng, P. and Yun, S. (2009). A block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization. *Journal of Optimization Theory and Applications* **140** 513–535. MR2481613
[21] Shalev-Shwartz, S. and Tewari, A. (2011). Stochastic methods for l_1-regularized loss minimization. *Journal of Machine Learning Research* **12** 1865–1892. MR2819020
[22] Dempster, A.P., Laird, N., and Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B* **39** 1–38. MR0501537
[23] Wei, G.C.G. and Tanner, M.A. (1990). A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association* **85** 699–704.
[24] Meng, X.-L. and Rubin, D.B. Maximum likelihood estimation via the ECM algorithm: a general framework. *Biometrika* **80** 267–278. MR1243503
[25] Städler, N. and Bühlmann, P. (2012). Missing values: sparse inverse covariance estimation and an extension to sparse regression. Statistics and Computings **22** 219–235. MR2865066
[26] Städler, N., Stekhoven, D.J., and Bühlmann, P. (2014). Pattern alternating maximization algorithm for missing data in high-dimensional problems. *Journal of Machine Learning Research* **15** 1903–1928. MR3231598
[27] Cai, J.-F., Candès, E., and Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization* **20** 1956–1982. MR2600248
[28] Kaplan, A., Nordman, D., and Vardeman, S. (2016). Properties and Bayesian fitting of restricted Boltzmann machines. arXiv:1612.01158.
[29] Liang, F., Jia, B., Xue, J., Li, Q., and Luo, Y. (2018). An imputation-regularized optimization algorithm for high-dimensional missing data problems and beyond. *Journal of the Royal Statistical Society, Series B* **80** 899–926.
[30] Fan, J. and Lv, J. (2008). Sure Independence Screening for Ultra-high Dimensional Feature Space. *Journal of the Royal Statistical Society, Series B* **70** 849–911. MR2530322
[31] Fan, J. and Song, R. (2010). Sure independence screening in generalized linear model with NP-dimensionality. *Annals of Statistics* **38** 3567–3604. MR2766861
[32] van der Vaart, A.W. and Wellner, J.A. (1996). *Weak Convergence and Empirical Processes*, Springer, New York. MR1385671
[33] Tibshirani, R. (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society, Series B* **58** 267–288. MR1379242
[34] Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B* **67** 301–320. MR2137327
[35] Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association* **96** 1348–1360. MR1946581
[36] Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics* **38** 894–942. MR2604701
[37] Brent, R.P. (1973). *Algorithms for Minimization without Derivatives*. Prentice-Hall: Englewood Cliffs, NJ. MR0339493
[38] Press, W.H., Teukolsky, S.A., Vetterling, W.T., and Flannery, B.P. (1992). *Numerical Recipes in C (2nd edition)* Cambridge University Press. MR1414682
[39] Rong, X. (2015). Package 'deepnet': Deep learning toolkit in R.
[40] de los Campos, G. and Rodriguez, P.P. (2015). Package 'BLR': Bayesian Linear Regression.
[41] Bleakley, K. and Yamanishi, Y. (2009). Supervised prediction of drug-target interactions using bipartite local models. *Bioinformatics* **25** 2397–2403.
[42] Cheng, F., Liu, C., Jiang, J., Lu, W., Li, W., Liu, G., Zhou, W., Huang, J., and Tang, Y. (2012). Prediction of drug-target interactions and drug repositioning via network-based inference. *PLoS Computational Biology* **8** e1002503.
[43] Wang, Y. and Zeng, J. (2013). Predicting drug-target interactions using restricted Boltzmann machines. *Bioinformatics* **29(ISMB/ECCB)** i126–i134.
[44] Ezzat, A., Wu, M., Li, X.-L., and Kwoh, C.-K. (2016). Drug-target interaction prediction via class imbalance-aware ensemble learning. *BMC Bioinformatics* **17** 509.
[45] Günther, S., Kuhn, M., Dunkel, M., Campillos, M., Senger, C., Petsalaki, E., Ahmed, J., Urdiales, E.G., Gewiess, A., Jensen, L.J., Schneider, R., Skoblo, R., Russell, R.B., Bourne, P.E., Bork, P., and Preissner, R. (2008). SuperTarget and Matador: resources for exploring drug-target relationships. *Nucleic Acids Research* **36** D919–922.
[46] Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pp.233-240.
[47] Hahsler, M. and Vereet, B. (2016). Package 'recommenderlab': Lab for developing and testing recommender algorithms.
[48] Gelman, R. and Rubin, D.B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science* **7** 457–472.
[49] Hinton, G. (2010). A practical guide to training restricted Boltzmann machines. *Department of Computer Science, University of Toronto, Toronto, Canada.*

Mingqi Wu
Shell, 150 N Dairy Ashford Rd
Houston, Texas 77079, USA
E-mail address: send2mqwu@gmail.com

Ye Luo
Faculty of Business and Economics
University of Hong Kong
Hong Kong, China
E-mail address: kurtluo@hku.hk

Faming Liang
Department of Statistics
Purdue University
West Lafayette, IN 47907, USA
E-mail address: fmliang@purdue.edu