Belief Propagation in Conditional RBMs for Structured Prediction

Wei Ping
Computer Science, UC Irvine
weiping.thu@gmail.com

Alexander Ihler
Computer Science, UC Irvine
ihler@ics.uci.edu

Abstract

Restricted Boltzmann machines (RBMs) and conditional RBMs (CRBMs) are popular models for a wide range of applications. In previous work, learning on such models has been dominated by contrastive divergence (CD) and its variants. Belief propagation (BP) algorithms are believed to be slow for structured prediction on conditional RBMs (e.g., Mnih et al. [2011]), and not as good as CD when applied in learning (e.g., Larochelle et al. [2012]). In this work, we present a matrix-based implementation of belief propagation algorithms on CRBMs, which is easily scalable to tens of thousands of visible and hidden units. We demonstrate that, in both maximum likelihood and max-margin learning, training conditional RBMs with BP as the inference routine can provide significantly better results than current state-of-the-art CD methods on structured prediction problems. We also include practical guidelines on training CRBMs with BP, and some insights on the interaction of learning and inference algorithms for CRBMs.

1 INTRODUCTION

A restricted Boltzmann machine (RBM) is a two-layer latent variable model that uses a layer of hidden units $\boldsymbol{h}$ to model the distribution of visible units $\boldsymbol{v}$. RBMs are widely used as building blocks for deep generative models, such as deep belief networks [Hinton et al., 2006] and deep Boltzmann machines [Salakhutdinov and Hinton, 2009]. Due to the intractability of the partition function in maximum likelihood estimation (MLE), RBMs are usually learned using the contrastive divergence (CD) algorithm [Hinton, 2002], which approximates the gradient of the log-partition function using a k-step Gibbs sampler (referred to as CD-k). To speed up the convergence of the Markov chain, a critical trick in CD-k is to initialize the state of the Markov chain with each training instance. Although it has been shown that CD-k does not follow the gradient of any objective function [Sutskever and Tieleman, 2010], it works well in many practical applications [Hinton, 2010]. An important variant of CD-k is persistent CD (PCD) [Tieleman, 2008]. PCD uses a persistent Markov chain during learning, where the Markov Chain is not reset between parameter updates. Because the learning rate is usually small and the model changes only slightly between parameter updates, the long-run persistent chain in PCD usually provides a better approximation to the target distribution than the limited step chain in CD-k.

A conditional RBM (CRBM) is the discriminative extension of RBM to include observed features $\boldsymbol{x}$; CRBM is used in deep probabilistic model for supervised learning [Hinton et al., 2006], and also provides a stand-alone solution to a wide range of problems such as classification [Larochelle and Bengio, 2008], human motion capture [Taylor et al., 2006], collaborative filtering [Salakhutdinov et al., 2007], and structured prediction [Mnih et al., 2011, Yang et al., 2014]. For structured prediction, a CRBM need not make any explicit assumptions about the structure of the output variables (visible units $\boldsymbol{v}$). This is especially useful in many applications where the structure of the outputs is challenging to describe (e.g., multi-label learning [Li et al., 2015]). In image denoising or image segmentation, the hidden units can encode higher-order correlations of visible units (e.g. shapes, or parts of object), which play the same role as high-order potentials but can improve the statistical efficiency.

In contrast to the success of CD methods for RBMs, it has been noted that both CD-k and PCD may not be well suited to learning conditional RBMs [Mnih et al., 2011]. In particular, PCD is not appropriate for learning such conditional models, because the observed fea-

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

tures $\boldsymbol{x}$ greatly affect the model potentials. This means we need to run a separate persistent chain for every training instance, which is costly for large datasets. To make things worse, as we revisit a training instance in stochastic gradient descent (SGD) (which is standard practice for large datasets), the model parameters will have changed substantially, making the persistent chain for this instance far from the target distribution. Also, given the observed features, CRBMs tend to be more peaked than RBMs in a purely generative setting. CD methods may make slow progress because it is difficult for the sampling procedure to explore these peaked but multi-modal distributions. It was also observed that the important trick in CD-k, which initializes the Markov chain using the training data, does not work well for CRBMs in structured prediction [Mnih et al., 2011]. In contrast, starting the Gibbs chain with a random state (which resembles the original learning algorithm for Boltzmann machines [Ackley et al., 1985]) provides better results.

Approximate inference methods, such as mean field (MF) and belief propagation (BP), can be employed as inference routines in learning as well as for making predictions after the CRBM has been learned [Welling and Teh, 2003, Yasuda and Tanaka, 2009]. Although loopy BP usually provides a better approximation of marginals than MF [Murphy et al., 1999], it was found to be slow on CRBMs for structured prediction and only considered practical on problems with few visible and hidden nodes [Mnih et al., 2011, Mandel et al., 2011]. This inefficiency prevents it from being widely applied to conditional RBMs for structured prediction, in which the CRBMs may have thousands of visible and hidden units. More importantly, there is a pervasive opinion that belief propagation does not work well on RBM-based models, especially for learning [Goodfellow et al., 2016, Chapter 16].

In this work, we present an efficient implementation of belief propagation algorithms for conditional RBMs. It takes advantage of the bipartite graph structure and is scalable to tens of thousands of visible units and hidden units. [1] Our algorithm uses a compact representation and only depends on matrix product and element-wise operations, which are typically highly optimized in modern high-performance computing architectures. We demonstrate that, in the conditional setting, learning RBM-based models with belief propagation and its variants can provide much better results than the state-of-the-art CD methods. We also show that the marginal structured SVM (MSSVM; [Ping et al., 2014]) can provide improvements for max-margin learning of CRBMs [Yang et al., 2014]. We include practical guidelines on training CRBMs, and some insights on the interaction of learning and message-passing algorithms for CRBMs.

We organize the rest of the paper as follows. Section 2 discusses some connections to related work. We review the RBM model and conditional RBMs in Section 3 and discuss the learning algorithms in Section 4. In Section 5, we provide our efficient inference procedure. We report experimental results in Section 6 and conclude the paper in Section 7.

2 RELATED WORK

Mnih et al. [2011] proposed the CD-PercLoss algorithm for conditional RBMs, which uses a CD-like stochastic search procedure to minimize the perceptron loss on training data. Given the observed features of the training instance, CD-PercLoss starts the Gibbs chain using the logistic regression component of the CRBM. Yang et al. [2014] trained CRBMs using a latent structured SVM (LSSVM) objective [Yu and Joachims, 2009], and used a greedy search (i.e., iterated conditional modes) for joint maximum a posteriori (MAP) inference over hidden and visible units.

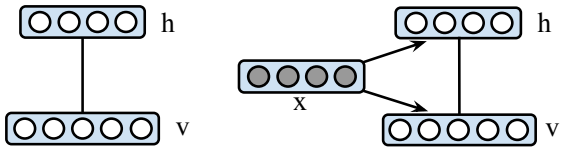
It is also feasible to apply the mean-field (MF) approximation for the partition function in MLE learning of RBMs and CRBMs [Peterson and Anderson, 1987]. Although efficient, this is conceptually problematic in the sense that it effectively maximizes an upper bound of the log-likelihood in learning. In addition, MF uses a unimodal proposal to approximate the multi-modal distribution, which may lead to unsatisfactory results.

Although belief propagation (BP) and its variants have long been used to learn conditional random fields (CRFs) with hidden variables [Quattoni et al., 2007, Ping et al., 2014], they are mainly applied on sparsely connected graphs (e.g., chains and grids) and were believed to be ineffective and slow on very dense graphs like CRBMs [Mnih et al., 2011, Goodfellow et al., 2016]. A few recent works impose particular assumptions on the type of edge potentials and provide efficient inference algorithms for fully connected CRFs. For example, the edge potentials in [Krähenbühl and Koltun, 2012] are defined by a linear combination of Gaussian kernels. In this work, however, we propose to speed up general belief propagation on conditional RBMs without any potential function restrictions.

3 MODELS

In this section, we review background on RBMs and conditional RBMs. We also discuss structured prediction with CRBMs.

[1] For random RBMs with $10,000$ visible units and $2,000$ hidden units, our Matlab implementation converges within a few seconds on a desktop with Intel Core i7 (3.6 GHz).



(a) RBM (b) conditional RBM

Figure 1: Graphical illustration of (a) RBM with $|\boldsymbol{v}| = 5$ visible units and $|\boldsymbol{h}| = 4$ hidden units, and (b) the extended CRBM with $\boldsymbol{v}$ as output variables, $\boldsymbol{x}$ as observed input features.

3.1 Restricted Boltzmann Machine

An RBM is a undirected graphical model (see Figure 1(a)) that defines a joint distribution over the vectors of visible units $\boldsymbol{v} \in \{0,1\}^{|\boldsymbol{v}|\times 1}$ and hidden units $\boldsymbol{h} \in \{0,1\}^{|\boldsymbol{h}|\times 1}$,

$$p(\boldsymbol{v},\boldsymbol{h}|\theta) = \frac{1}{Z(\theta)} \exp\big(-E(\boldsymbol{v},\boldsymbol{h};\theta)\big), \qquad (1)$$

where $|\boldsymbol{v}|$ and $|\boldsymbol{h}|$ are the dimensions of $\boldsymbol{v}$ and $\boldsymbol{h}$ respectively; $E(\boldsymbol{v},\boldsymbol{h};\theta)$ is the energy function,

$$E(\boldsymbol{v},\boldsymbol{h};\theta) = -\boldsymbol{v}^\top W^{vh}\boldsymbol{h} - \boldsymbol{v}^\top \boldsymbol{b}^1 - \boldsymbol{h}^\top \boldsymbol{b}^2;$$

and $\theta = \{W^{vh}, \boldsymbol{b}^1, \boldsymbol{b}^2\}$ are the model parameters, including pairwise interaction terms $W^{vh} \in \mathbb{R}^{|\boldsymbol{v}|\times|\boldsymbol{h}|}$, and bias terms $\boldsymbol{b}^1 \in \mathbb{R}^{|\boldsymbol{v}|\times 1}$ for visible units and $\boldsymbol{b}^2 \in \mathbb{R}^{|\boldsymbol{h}|\times 1}$ for hidden units. The function $Z(\theta)$ is the normalization constant, or *partition function*,

$$Z(\theta) = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} \exp\big(-E(\boldsymbol{v},\boldsymbol{h};\theta)\big),$$

which is typically intractable to calculate.

3.2 Conditional RBM

The conditional RBM (CRBM) extends RBMs to include observed features $\boldsymbol{x}$ (see Figure 1(b) for an illustration [Mnih et al., 2011]), and defines a joint conditional distribution over $\boldsymbol{v}$ and $\boldsymbol{h}$ given features $\boldsymbol{x} \in \mathbb{R}^{|\boldsymbol{x}|\times 1}$,

$$p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{x};\theta) = \frac{1}{Z(\boldsymbol{x};\theta)} \exp\big(-E(\boldsymbol{v},\boldsymbol{h},\boldsymbol{x};\theta)\big), \qquad (2)$$

where the energy function E is defined as,

$$E(\boldsymbol{v},\boldsymbol{h},\boldsymbol{x};\theta) = -\boldsymbol{v}^\top W^{vh}\boldsymbol{h} - \boldsymbol{v}^\top W^{vx}\boldsymbol{x} - \boldsymbol{h}^\top W^{hx}\boldsymbol{x} - \boldsymbol{v}^\top \boldsymbol{b}^v - \boldsymbol{h}^\top \boldsymbol{b}^h,$$

and $\theta = \{W^{vh}, W^{vx}, W^{hx}, b^v, b^h\}$ are model parameters. $Z(\boldsymbol{x};\theta)$ is the $\boldsymbol{x}$-dependent partition function,

$$Z(\boldsymbol{x};\theta) = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} \exp\big(-E(\boldsymbol{v},\boldsymbol{h},\boldsymbol{x};\theta)\big).$$

One can view an RBM as a special CRBM with $\boldsymbol{x} \equiv 0$.

Conditional Distribution Because CRBMs have a bipartite structure given the observed features, the conditional distributions $p(\boldsymbol{v}|\boldsymbol{h},\boldsymbol{x})$ and $p(\boldsymbol{h}|\boldsymbol{v},\boldsymbol{x})$ are fully factored and can be written as,

$$p(\boldsymbol{v}|\boldsymbol{h},\boldsymbol{x}) = \prod_i p(v_i|\boldsymbol{h},\boldsymbol{x}), \quad p(\boldsymbol{h}|\boldsymbol{v},\boldsymbol{x}) = \prod_j p(h_j|\boldsymbol{v},\boldsymbol{x})$$
$$\text{with } p(v_i = 1|\boldsymbol{h},\boldsymbol{x}) = \sigma\big(W^{vh}_{i\bullet}\boldsymbol{h} + W^{vx}_{i\bullet}\boldsymbol{x} + b^v_i\big),$$
$$p(h_j = 1|\boldsymbol{v},\boldsymbol{x}) = \sigma\big(\boldsymbol{v}^T W^{vh}_{\bullet j} + W^{hx}_{j\bullet}\boldsymbol{x} + b^h_j\big), \quad (3)$$

where $\sigma(u) = 1/(1+\exp(-u))$ is the logistic function, $W^{vh}_{i\bullet}$ and $W^{vh}_{\bullet j}$ are the i-th row and j-th column of W^{vh} respectively, $W^{vx}_{i\bullet}$ is the i-th row of of W^{vx}, and $W^{hx}_{j\bullet}$ is the j-th row of W^{hx}. Eq. (3) allows us to derive a blocked Gibbs sampler that iteratively alternates between drawing $\boldsymbol{v}$ and $\boldsymbol{h}$.

Marginal Distribution The marginal distribution of visible units $\boldsymbol{v}$ given observed features $\boldsymbol{x}$ is,

$$p(\boldsymbol{v}|\boldsymbol{x}) = \textstyle\sum_{\boldsymbol{h}} p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x};\theta)} \exp\big[-F(\boldsymbol{v},\boldsymbol{x};\theta)\big] \quad (4)$$

where the negative energy function has analytic form,

$$-F(\boldsymbol{v},\boldsymbol{x};\theta) = \sum_{j=1}^{|\boldsymbol{h}|} \log\big[1 + \exp\big(\boldsymbol{v}^\top W^{vh}_{\bullet j} + W^{hx}_{j\bullet}\boldsymbol{x} + b^h_j\big)\big] + \boldsymbol{v}^\top W^{vx}\boldsymbol{x} + \boldsymbol{v}^\top \boldsymbol{b}^v.$$

Note, after marginalizing out hidden variables, the log-linear model (2) becomes a non-linear model (4), which can capture high-order correlations among visible units. This property is essentially important in many applications of CRBMs with structured output [e.g., Salakhutdinov et al., 2007, Mnih et al., 2011].

3.3 Structured Prediction with CRBMs

In structured prediction, the visible units $\boldsymbol{v}$ typically represent output variables, while the observed $\boldsymbol{x}$ represent input features, and the hidden units $\boldsymbol{h}$ facilitate the modeling of output variables given observed features. To make predictions, one choice is to infer the modes of the singleton marginals, $p(v_i|\boldsymbol{x}) = \sum_{\boldsymbol{v}_{\setminus i}} \sum_{\boldsymbol{h}} p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{x})$. This marginalization inference is intractable and is closely related to calculating the partition function. One can also decode the output $\boldsymbol{v}$ by performing joint maximum *a posteriori* (MAP) inference [e.g., Yu and Joachims, 2009, Yang et al., 2014],

$$(\hat{\boldsymbol{v}},\hat{\boldsymbol{h}}) = \operatorname*{argmax}_{\boldsymbol{v},\boldsymbol{h}} \; p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{x}),$$

which gives a prediction for the pair $(\boldsymbol{v},\boldsymbol{h})$; one obtains a prediction of $\boldsymbol{v}$ by simply discarding the $\boldsymbol{h}$ component. Intuitively, the joint MAP prediction is "over-confident", since it deterministically assigns the

hidden units to their most likely states, and is not robust when the uncertainty of the hidden units is high. One promising alternative for CRBM is marginal MAP prediction [Ping et al., 2014],

$$\tilde{\boldsymbol{v}} = \operatorname*{argmax}_{\boldsymbol{v}} p(\boldsymbol{v}|\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{v}} \sum_{\boldsymbol{h}} \exp\big(-E(\boldsymbol{v},\boldsymbol{h},\boldsymbol{x};\theta)\big),$$

which explicitly takes into account the uncertainty of the hidden units by marginalizing them out. In general, these predictions are intractable in CRBMs, and one must use approximate inference methods, such as mean field or belief propagation.

4 LEARNING

In this section, we discuss different learning methods for conditional RBMs.

4.1 MLE and Related Algorithms

Assume we have a training set $\{\boldsymbol{v}^n, \boldsymbol{x}^n\}_{n=1}^N$; then, the log-likelihood can be written as,

$$\sum_{n=1}^{N} \Big\{ \log \sum_{\boldsymbol{h}} \exp\big(-E(\boldsymbol{v}^n,\boldsymbol{h},\boldsymbol{x}^n;\theta)\big) - \log Z(\boldsymbol{x}^n;\theta) \Big\}.$$

To efficiently maximize the objective function, stochastic gradient descent (SGD) is usually applied. Given a randomly chosen instance $\{\boldsymbol{v}^n, \boldsymbol{x}^n\}$, one can show that the gradient of log-likelihood w.r.t. W^{vh} is,

$$\frac{\partial \log p(\boldsymbol{v}^n|\boldsymbol{x}^n)}{\partial W^{vh}} = \boldsymbol{v}^n(\boldsymbol{\mu}^n)^\top - \mathbb{E}_{p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{x}^n)}\big[\boldsymbol{v}\boldsymbol{h}^\top\big], \quad (5)$$

where $\boldsymbol{\mu}^n = \sigma({W^{vh}}^\top \boldsymbol{v}^n + W^{hx}\boldsymbol{x}^n + \boldsymbol{b}^h)$ and the logistic function σ is applied in an element-wise manner. The positive part of the gradient can be calculated exactly. The negative part arises from the derivatives of the log-partition function and is intractable to calculate. The gradients of log-likelihood w.r.t. other parameters are analogous to Eq. (5), and can be found in Appendix A.

CD-k initializes the Gibbs chain by instance $\boldsymbol{v}^n$, and performs k-step Gibbs sampling by Eq. (3). Then, the empirical moment is used as a substitute for the intractable expectation $\mathbb{E}_{p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{x}^n)}\big[\boldsymbol{v}\boldsymbol{h}^\top\big]$. Although this works well on RBMs, it gives unsatisfactory results on CRBMs. In practice, the conditional distributions $p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{x}^n)$ are strongly influenced by the observed features $\boldsymbol{x}^n$, and usually more peaked than generative RBMs. It is usually difficult for a Markov chain with few steps (e.g., 10) to explore these peaked and multimodal distributions. PCD uses a long-run persistent Markov chain to improve convergence, but is not suitable for CRBMs as discussed in Section 1.

Sum-product BP and mean field methods provide pseudo-marginals as substitutes for the intractable expectations in Eq. (5). These deterministic gradient estimates have the advantage that a larger learning rate can be used. BP tends to give a more accurate estimate of $\log Z$ and marginals, but is reported to be slow on CRBMs and is impractical on problems with large output dimension and hidden layer sizes in structured prediction [Mnih et al., 2011].

More importantly, it was observed that belief propagation usually gives unsatisfactory results when learning vanilla RBMs. This is mainly because the parameters' magnitude gradually increases during learning; the RBM model eventually undergoes a "phase transition" after which BP has difficulty converging [Ihler et al., 2005, Mooij and Kappen, 2005]. If BP does not converge, it can not provide a meaningful gradient direction to update the model, and the leaning becomes stuck. However, CRBMs appear to behave quite differently, due to operating in the "high signal" regime provided by an informative observation $\boldsymbol{x}$. This improves the convergence behaviour of BP, which may not be surprising since loopy BP is widely accepted as useful in learning other conditional models (e.g., grid CRFs for image segmentation). In addition, given N training instances for learning the CRBM, BP is actually performed on N different RBMs corresponding to different features $\boldsymbol{x}^n$. During any particular phase of learning, BP may have trouble converging on some training instances, but we can still make progress as long as BP converges on the majority of instances. We demonstrate this behavior in our experiments.

4.2 Max-Margin Learning

Another by-product of using BP is that it enables us to apply the marginal structured SVM (MSSVM) [Ping et al., 2014] framework for max-margin learning of CRBMs,

$$\min_{\theta} \sum_{n=1}^{N} \Big\{ \max_{\boldsymbol{v}} \log \sum_{\boldsymbol{h}} \exp\Big(\Delta(\boldsymbol{v},\boldsymbol{v}^n) - E(\boldsymbol{v},\boldsymbol{h},\boldsymbol{x};\theta)\Big) - \log \sum_{\boldsymbol{h}} \exp\Big(-E(\boldsymbol{v}^n,\boldsymbol{h},\boldsymbol{x}^n;\theta)\Big) \Big\}, \quad (6)$$

where the loss function $\Delta(\boldsymbol{v},\boldsymbol{v}^n) = \sum_i \Delta(v_i, v_i^n)$ is decomposable (e.g., Hamming loss). In contrast to LSSVM [Yu and Joachims, 2009, Yang et al., 2014], MSSVM marginalizes over the uncertainty of hidden variables, and can significantly outperform LSSVM when that uncertainty is large [Ping et al., 2014]. Experimentally, we find that MSSVM improves performance of max-margin CRBMs, likely because there is usually non-trivial uncertainty in the hidden units. Given an instance $\{\boldsymbol{v}^n, \boldsymbol{x}^n\}$, the stochastic gradient of

Eq. (6) w.r.t. W^{vh} is,

$$\frac{\partial l(\boldsymbol{v}^n, \boldsymbol{x}^n)}{\partial W^{vh}} = \mathbb{E}_{p(\boldsymbol{h}|\hat{\boldsymbol{v}}, \boldsymbol{x}^n)}\left[\hat{\boldsymbol{v}}\boldsymbol{h}^T\right] - \boldsymbol{v}^n(\boldsymbol{\mu}^n)^\top, \tag{7}$$

where $\boldsymbol{\mu}^n$ is defined as in Eq. (5); $\hat{\boldsymbol{v}}$ is the loss-augmented marginal MAP prediction,

$$\hat{\boldsymbol{v}} = \operatorname*{argmax}_{\boldsymbol{v}} \sum_{\boldsymbol{h}} \exp\Big(\Delta(\boldsymbol{v}, \boldsymbol{v}^{(n)}) - E(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{x}^n; \theta)\Big);$$

and "mixed-product" belief propagation [Liu and Ihler, 2013] or dual-decomposition method [Ping et al., 2015] for marginal MAP can provide pseudo-marginals to estimate the intractable expectation. (The gradients for other parameters are analogous.)

5 APPROXIMATE INFERENCE

In this section, we present a matrix-based implementation of sum-product and mixed-product BP algorithms for RBMs. Given a particular $\boldsymbol{x}^n$ in CRBM (2), we obtain a $\boldsymbol{x}^n$-dependent RBM model,

$$p(\boldsymbol{v}, \boldsymbol{h}|\boldsymbol{x}^n) = \frac{1}{Z(\theta(\boldsymbol{x}^n))} \exp\left(\boldsymbol{v}^\top W^{vh}\boldsymbol{h} + \boldsymbol{v}^\top \boldsymbol{b}^1 + \boldsymbol{h}^\top \boldsymbol{b}^2\right),$$

where the bias terms $\boldsymbol{b}^1 = \boldsymbol{b}^v + W^{vx}\boldsymbol{x}^n$, $\boldsymbol{b}^2 = \boldsymbol{b}^h + W^{hx}\boldsymbol{x}^n$, and thus we can directly apply the algorithm to CRBMs.

5.1 Message-passing in RBMs

We first review the standard message-passing form in RBMs. On a dense graphical models like RBMs, to reduce the amount of calculation, one should always pre-compute the product of incoming messages (or the beliefs) on the nodes, and reuse them to perform updates of all outgoing messages. In sum-product BP, we write the fixed-point update rule for the message sent from hidden unit h_j to visible unit v_i as,

$$m_{j\to i}(v_i) \propto \sum_{h_j} \exp\left(v_i W_{ij}^{vh} h_j\right) \cdot \frac{\tau(h_j)}{m_{i\to j}(h_j)}, \tag{8}$$

where the belief on h_j is

$$\tau(h_j) \propto \exp\left(h_j b_j^2\right) \cdot \prod_{k=1}^{|\boldsymbol{v}|} m_{k\to j}(h_j). \tag{9}$$

The update rule for the message sent from v_i to h_j is,

$$m_{i\to j}(h_j) \propto \sum_{v_i} \exp\left(v_i W_{ij}^{vh} h_j\right) \cdot \frac{\tau(v_i)}{m_{j\to i}(v_i)}, \tag{10}$$

where the belief on v_i is,

$$\tau(v_i) \propto \exp\left(v_i b_i^1\right) \cdot \prod_{k=1}^{|\boldsymbol{h}|} m_{k\to i}(v_i). \tag{11}$$

In mixed-product BP, the message sent from hidden unit to visible unit is the same as Eq. (8). The message sent from visible unit v_i to hidden unit h_j is

$$\tilde{m}_{i\to j}(h_j) \propto \exp\left(\tilde{v}_i W_{ij}^{vh} h_j\right) \cdot \frac{\tau(\tilde{v}_i)}{m_{j\to i}(\tilde{v}_i)}, \tag{12}$$

where $\tilde{v}_i = \operatorname{argmax}_{v_i} \tau(v_i)$, and $\tau(v_i)$ is defined in Eq. (11). These update equations are repeatedly applied until the values converge (hopefully), or a stopping criterion is satisfied. Then, the pairwise belief on (v_i, h_j) is calculated as,

$$\tau(v_i, h_j) \propto \exp\left(v_i W_{ij}^{vh} h_j\right) \cdot \frac{\tau(v_i)}{m_{j\to i}(v_i)} \cdot \frac{\tau(h_j)}{m_{i\to j}(h_j)}.$$

It is well known that BP on loopy graphs is not guaranteed to converge, although in practice it usually does [Murphy et al., 1999].

5.2 Matrix-based BP Algorithms

Our algorithms use a compact matrix representation. We denote the "free" belief vectors and matrices as,

$$\begin{aligned}
&\boldsymbol{\tau}^v \in \mathbb{R}^{|\boldsymbol{v}|\times 1}, \text{ where } \tau_i^v = \tau(v_i = 1),\\
&\boldsymbol{\tau}^h \in \mathbb{R}^{|\boldsymbol{h}|\times 1}, \text{ where } \tau_j^h = \tau(h_j = 1),\\
&\Gamma \in \mathbb{R}^{|\boldsymbol{v}|\times|\boldsymbol{h}|}, \text{ where } \Gamma_{ij} = \tau(v_i = 1, h_j = 1).
\end{aligned}$$

Other beliefs can be represented by these "free" beliefs:

$$\begin{aligned}
&\tau(v_i = 0) = 1 - \tau_i^v, \quad \tau(h_j = 0) = 1 - \tau_j^h,\\
&\tau(v_i = 1, h_j = 0) = \tau_i^v - \Gamma_{ij},\\
&\tau(v_i = 0, h_j = 1) = \tau_j^h - \Gamma_{ij},\\
&\tau(v_i = 0, h_j = 0) = 1 + \Gamma_{ij} - \tau_i^v - \tau_j^h.
\end{aligned}$$

We similarly define the normalized message matrices,

$$\begin{aligned}
&M^{vh} \in \mathbb{R}^{|\boldsymbol{v}|\times|\boldsymbol{h}|}, \quad M_{ij}^{vh} = m_{j\to i}(v_i = 1),\\
&M^{hv} \in \mathbb{R}^{|\boldsymbol{h}|\times|\boldsymbol{v}|}, \quad M_{ji}^{hv} = m_{i\to j}(h_j = 1).
\end{aligned}$$

Thus, M^{vh} represents all the messages sent from $\boldsymbol{h}$ to $\boldsymbol{v}$, and M^{hv} represents all the messages from $\boldsymbol{v}$ to $\boldsymbol{h}$.

One can show (see Appendix B.1) that the update equation for message matrix M^{vh} in both sum-product and mixed-product BP is

$$\begin{aligned}
M^{vh} = \ & \sigma\Big(\log\Big(\frac{\exp(W^{vh}) \circ \Lambda_1^{vh} + \Lambda_2^{vh}}{\Lambda_1^{vh} + \Lambda_2^{vh}}\Big)\Big), \\
\text{where} \quad & \Lambda_1^{vh} = (\mathbf{1}^{hv} - M^{hv})^\top \cdot \operatorname{diag}(\boldsymbol{\tau}^h),\\
& \Lambda_2^{vh} = {M^{hv}}^\top \cdot \operatorname{diag}(\mathbf{1}^h - \boldsymbol{\tau}^h),
\end{aligned} \tag{13}$$

where $\mathbf{1}^{hv}$ is a $|\boldsymbol{h}|\times|\boldsymbol{v}|$ matrix of ones, $\mathbf{1}^h$ is a $|\boldsymbol{h}|\times 1$ vector of ones, $\circ$ is the element-wise Hadamard product,

and diag(·) extracts the elements in a vector to form a diagonal matrix. The logarithm, fraction and logistic function are all applied in an element-wise manner. Similarly, the update equation for message matrix M^{hv} in sum-product BP is

$$M^{hv} = \sigma\Big(\log\Big(\frac{\exp(W^{vh\top})\circ\Lambda_1^{hv}+\Lambda_2^{hv}}{\Lambda_1^{hv}+\Lambda_2^{hv}}\Big)\Big), \quad (14)$$

$$\text{where} \quad \Lambda_1^{hv} = (\mathbf{1}^{vh} - M^{vh})^\top \cdot \mathrm{diag}(\boldsymbol{\tau}^v),$$

$$\Lambda_2^{hv} = M^{vh\top} \cdot \mathrm{diag}(\mathbf{1}^v - \boldsymbol{\tau}^v),$$

with $\mathbf{1}^{vh}$ a $|\boldsymbol{v}| \times |\boldsymbol{h}|$ matrix of ones, and $\mathbf{1}^v$ a $|\boldsymbol{v}| \times 1$ vector of ones. In mixed-product BP, the update equation for message matrix M^{vh} is

$$M^{hv} = \sigma\Big(W^{vh\top} \cdot \mathrm{diag}(\tilde{\boldsymbol{v}})\Big), \quad (15)$$

where $\tilde{v}_i = \mathrm{argmax}_{v_i}\, \tau^v(v_i)$ for all v_i. In addition, one can show (see Appendix B.2) that the belief vectors $\boldsymbol{\tau}^v$ and $\boldsymbol{\tau}^h$ can be calculated as,

$$\boldsymbol{\tau}^v = \sigma\Big(\boldsymbol{b}^1 + \log\Big(\frac{M^{vh}}{\mathbf{1}^{vh} - M^{vh}}\Big)\cdot \mathbf{1}^h\Big), \quad (16)$$

$$\boldsymbol{\tau}^h = \sigma\Big(\boldsymbol{b}^2 + \log\Big(\frac{M^{hv}}{\mathbf{1}^{hv} - M^{hv}}\Big)\cdot \mathbf{1}^v\Big), \quad (17)$$

where $\cdot$ is the matrix product. These update equations are repeatedly applied until the stopping criterion is satisfied. After that, the pairwise belief matrix Γ can be calculated as,

$$\Gamma = \frac{\Gamma^{11}}{\Gamma^{11}+\Gamma^{01}+\Gamma^{10}+\Gamma^{00}}, \text{ where} \quad (18)$$

$$\Gamma^{11} = \exp(W^{vh}) \circ (\boldsymbol{\tau}^v \cdot \boldsymbol{\tau}^{h\top}) \circ (\mathbf{1}^{vh} - M^{vh}) \circ (\mathbf{1}^{hv} - M^{hv})^\top,$$

$$\Gamma^{01} = \big((\mathbf{1}^v - \boldsymbol{\tau}^v)\cdot \boldsymbol{\tau}^{h\top}\big) \circ M^{vh} \circ (\mathbf{1}^{hv} - M^{hv})^\top,$$

$$\Gamma^{10} = \big(\boldsymbol{\tau}^v \cdot (\mathbf{1}^h - \boldsymbol{\tau}^h)^\top\big) \circ (\mathbf{1}^{vh} - M^{vh}) \circ M^{hv\top},$$

$$\Gamma^{00} = \big((\mathbf{1}^v - \boldsymbol{\tau}^v)\cdot (\mathbf{1}^h - \boldsymbol{\tau}^h)\big) \circ M^{vh} \circ M^{hv\top}.$$

We summarize the matrix-based sum-product BP and mixed-product BP in Algorithm 1. It is well known that asynchronous (sequential) BP message updates usually converge much faster than synchronous updates [e.g., Wainwright et al., 2003, Gonzalez et al., 2009]; in Algorithm 1, although messages are sent in parallel from all hidden units to visible units, the bipartite graph structure ensures that these are actually *asynchronous* updates, which helps convergence in practice. Our method is also related to message-passing algorithms designed for other binary networks, such as binary LDPC codes [Kschischang et al., 2001], which parametrize each message by a single real number using a hyperbolic tangent transform. Our algorithm is specially designed for RBM-based models, and significantly speeds up BP by taking advantage of the RBM structure and using only matrix operations.

Algorithm 1 Sum(mixed)-product BP on RBM

Input: $\{W^{vh}, \boldsymbol{b}^1, \boldsymbol{b}^2\}$, number of iterations T
Output: beliefs $\{\boldsymbol{\tau}^v, \boldsymbol{\tau}^h, \Gamma\}$
initialize message matrices:
$M^{vh} = 0.5 \times \mathbf{1}^{vh}, \quad M^{hv} = 0.5 \times \mathbf{1}^{hv}$;
initialize beliefs: $\boldsymbol{\tau}^v = \sigma(\boldsymbol{b}^1), \quad \boldsymbol{\tau}^h = \sigma(\boldsymbol{b}^2)$;
for $t = 1$ **to** T **do**
  send messages from $\boldsymbol{h}$ to $\boldsymbol{v}$:
  $\Lambda_1^{vh} = (\mathbf{1}^{hv} - M^{hv})^\top \cdot \mathrm{diag}(\boldsymbol{\tau}^h)$;
  $\Lambda_2^{vh} = M^{hv\top} \cdot \mathrm{diag}(\mathbf{1}^h - \boldsymbol{\tau}^h)$;
  $M^{vh} = \sigma\Big(\log\Big(\frac{\exp(W^{vh})\circ\Lambda_1^{vh}+\Lambda_2^{vh}}{\Lambda_1^{vh}+\Lambda_2^{vh}}\Big)\Big)$; (13)
  $\boldsymbol{\tau}^v = \sigma\Big(\boldsymbol{b}^1 + \log\Big(\frac{M^{vh}}{\mathbf{1}^{vh}-M^{vh}}\Big)\cdot\mathbf{1}^h\Big)$; (16)
  send messages from $\boldsymbol{v}$ to $\boldsymbol{h}$
  in sum-product BP:
  $\Lambda_1^{hv} = (\mathbf{1}^{vh} - M^{vh})^\top \cdot \mathrm{diag}(\boldsymbol{\tau}^v)$;
  $\Lambda_2^{hv} = M^{vh\top} \cdot \mathrm{diag}(\mathbf{1}^v - \boldsymbol{\tau}^v)$;
  $M^{hv} = \sigma\Big(\log\Big(\frac{\exp(W^{vh\top})\circ\Lambda_1^{hv}+\Lambda_2^{hv}}{\Lambda_1^{hv}+\Lambda_2^{hv}}\Big)\Big)$; (14)
  or, in mixed-product BP:
  $M^{hv} = \sigma\Big(W^{vh\top} \cdot \mathrm{diag}(\tilde{\boldsymbol{v}})\Big)$; (15)
  $\boldsymbol{\tau}^h = \sigma\Big(\boldsymbol{b}^2 + \log\Big(\frac{M^{hv}}{\mathbf{1}^{hv}-M^{hv}}\Big)\cdot\mathbf{1}^v\Big)$; (17)
end for
$\Gamma = \frac{\Gamma^{11}}{\Gamma^{11}+\Gamma^{01}+\Gamma^{10}+\Gamma^{00}}$ as defined in Eq. (18);

In practice, our matrix implementation runs orders of magnitude faster than standard implementation of belief propagation, e.g., the C++ factor graph package libDAI [Mooij, 2010], which has been used for RBM assessments [e.g., Hadjis and Ermon, 2015]. For an RBM with 1000 visible and 500 hidden units, 10 iterations of BP in our Matlab implementation takes 0.5 seconds on a laptop with Intel Core i5 (2.5GHz). In libDAI (with gcc -O3, i.e., fully optimized for speed), 10 iterations of BP takes 297.4 seconds, approximately 600× slower. This is mainly because matrix operations are highly optimized in modern computer architectures, e.g., they are performed in parallel in the instruction pipeline, and no pointers (to messages, neighbors, etc.) need to be dereferenced.

6 Experiments

In this section, we compare our methods with state-of-the-art algorithms for learning CRBMs on two datasets: MNIST and Caltech101 Silhouettes.

Datasets: The MNIST database [LeCun et al., 1998] contains 60,000 images in the training set and 10,000 test set images. We randomly select 10,000 images from training as the validation set. Each image is 28 × 28 pixels, thus $|\boldsymbol{v}| = 784$. We binarize the grayscale images by thresholding the pixels at 127, to

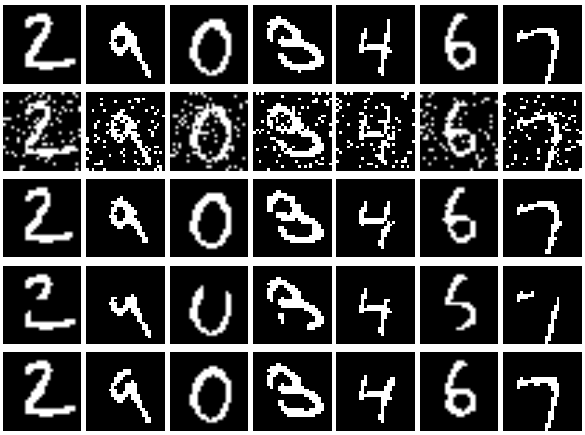


Figure 2: (Row 1) 7 original images from the test set. (Row 2) The noisy (10%) images. (Row 3) The images predicted from noisy images. (Row 4) The occluded (8×8) images. (Row 5) The images predicted from the occluded images. Rows 3 and 5 use our MLE-BP for learning.

obtain the clean image $\boldsymbol{v}$. We test two types of structured prediction tasks in our experiment. The first task is image denoising and denoted "*noisy* MNIST", where the noisy image $\boldsymbol{x}$ is obtained by flipping either 10% or 20% of the entries in $\boldsymbol{v}$. The second task is image completion, denoted *occluded* MNIST, where the occluded image $\boldsymbol{x}$ is obtained by setting a random patch within the image $\boldsymbol{v}$ to 0. The patch size is either 8×8 or 12×12 pixels. See Figure 2 for an illustration. The Caltech101 Silhouettes dataset [Marlin et al., 2010] has $8,671$ images with 28×28 binary pixels, where each image represents object silhouette. The dataset is divided into three subsets: $4,100$ examples for training, $2,264$ for validation and $2,307$ for testing. We test both image denoising and image completion tasks. The noisy image $\boldsymbol{x}$ in *noisy* Caltech101 is obtained by flipping 20% of the pixels from the clean $\boldsymbol{v}$, and the occluded image in *occluded* Caltech101 is obtained by setting a random 12×12 patch to 1.

Model: Following [Mnih et al., 2011], we structured the CRBM model with 256 hidden units, giving 1 million parameters in the model. All the learning algorithms are applied to learn this CRBM model. The logistic regression method can be viewed as learning this CRBM with only W^{vx} and $\boldsymbol{b}^v$ non-zero.

Algorithms: We train several CRBMs using the state-of-the-art CD methods, including CD-1, CD-10 and CD-PercLoss. We also train models to optimize likelihood (MLE) using mean field (MLE-MF) and sum-product BP (MLE-BP).[2] Finally, we train MSSVM CRBMs using mixed-product BP, and LSSVM CRBMs using max-product BP. A fixed learning rate is selected from the set $\{0.05, 0.02, 0.01, 0.005\}$ using the validation set, and the mini-batch size is selected from the set $\{10, 20, 40, 80, 160\}$. The CD-PercLoss algorithm uses 10-step Gibbs sampling in the stochastic search process. All the CD methods use 200 epochs in training. In contrast, MLE-MF, MLE-BP, MSSVM and LSSVM use 50 epochs, because BP and MF provide a deterministic gradient estimate and larger learning rates can be applied. Early stopping based on the validation error is also used for all methods.[3] We test the learned models of the CD methods and MLE-MF with mean-field predictions; the learned model of MLE-BP with sum-product BP predictions; MSSVM with mixed-product BP; and LSSVM with max-product BP.

Table 1: Average test error (%) for image denoising on *noisy* MINIST. All denotes the percentage incorrectly labeled pixels among all pixels. Changed denotes the percentage of errors among pixels that were changed by the noise process.

Dataset	Noisy (10%)		Noisy (20%)	
Method	All	Changed	All	Changed
LR	1.960	12.531	4.088	12.609
CD-1	1.925	12.229	4.012	12.597
CD-10	1.816	11.103	3.995	11.271
CD-PercLoss	1.760	11.121	3.970	10.876
MLE-MF	1.862	11.319	3.917	10.939
MLE-BP	**1.688**	**10.718**	**3.691**	**10.409**
LSSVM	1.807	11.565	3.910	11.175
MSSVM	1.751	11.023	3.804	10.627

Table 2: Average test error (%) for image completion on *occluded* MINIST.

Dataset	Occluded (8×8)		Occluded (12×12)	
Method	All	Changed	All	Changed
LR	1.468	61.304	3.498	53.971
CD-1	1.814	63.130	3.983	58.376
CD-10	1.707	67.925	3.921	63.237
CD-PercLoss	1.394	45.684	3.483	35.755
MLE-MF	1.492	49.553	3.477	40.703
MLE-BP	**1.329**	**39.785**	**3.117**	36.233
LSSVM	1.496	44.037	3.468	39.140
MSSVM	1.391	41.829	3.273	**35.712**

Results: Table 1 shows the percentage of incorrectly labeled pixels on the *noisy* MNIST for different methods. "All" denotes the errors among all pixels and is the main measurement. We also report the "Changed" errors among the pixels that were changed by the noise/occlusion process. MLE-BP works best and provides 4% and 7% relative improvement over CD-

[2] In previous work [Mnih et al., 2011], MLE-BP was considered impractical on this task due to the efficiency issue.

[3] In experiments, we found that early stopping always worked better than the Frobenius norm regularization.

Table 3: Average test error (%) for image denoising & completion on Caltech101 Silhouettes dataset.

Dataset / Method	Noisy (20%) All	Noisy (20%) Changed	Occluded (12 × 12) All	Occluded (12 × 12) Changed
LR	5.653	11.460	4.771	16.587
CD-1	5.876	12.423	5.033	20.300
CD-10	5.736	12.013	5.149	21.087
CD-PercLoss	5.622	10.808	5.081	15.102
MLE-MF	5.617	11.083	4.692	15.995
MLE-BP	**5.445**	**10.731**	4.548	16.541
LSSVM	5.628	11.468	4.703	16.014
MSSVM	5.549	11.389	**4.534**	**14.918**

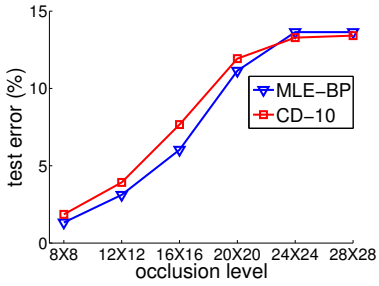


Figure 3: Average test error (%) for image completion on *occluded* MINIST under different occlusion levels.

PercLoss on two datasets with different noise levels. Table 2 shows the results on *occluded* MNIST. Here MLE-BP provides 4% and 10% relative improvement over CD-PercLoss on the two datasets, respectively. CD-k gives unsatisfactory results in both cases. Here MSSVM performs worse than MLE-BP, but better than the other methods in Table 1 and 2. The image completion task is viewed as more difficult on Changed pixels. However, again training the CRBM with MLE-BP gives very good results; see the last two rows of images in Figure 2. Table 3 demonstrate the results on Caltech101 Silhouettes; in this setting, MLE-BP and MSSVM perform the best for image denoising and image completion, respectively.

Figure 3 shows the results for image completion under different occlusion levels. MLE-BP works better than CD-10, unless the images are almost fully occluded. Note, the full occlusion (28×28) corresponds to no conditioning (i.e., $\boldsymbol{x} \equiv 0$), and the CRBM models are reduce to vanilla RBMs.

Discussion: We include several observations on the interaction of learning and inference algorithms for CRBMs: (1) Early on in learning, message passing is fast to converge, typically within ≈ 7 iterations. As learning continues, the magnitudes of the parameters gradually increase, and it becomes harder for BP to converge quickly. One simple but effective strategy is to set the number of iterations to $T = 7 + \text{epoch}$ (e.g., at epoch 10, $T = 17$). See Figure 4 for an illustration

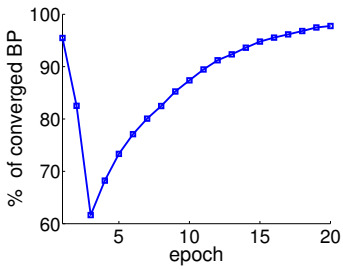


Figure 4: Percentage of converged BP in each epoch during MLE-BP training on *occluded* (8×8) MNIST.

of the convergence behavior of BP using this strategy during training. We set the convergence tolerance $\epsilon = 0.001$. The model undergoes a change of convergence behaviour around epoch 3, but we can still make progress as BP converges on the majority of training instances. (2) No damping is better. Although message damping can improve the convergence of BP, it always requires more iterations of message-passing and effectively slows down the progress of learning CRBMs. (3) The approximate inference algorithms used in learning and test should be matched, which means the inference method (BP or mean-field), number of iterations etc., should be the same. Otherwise, we see unsatisfactory results. (4) Learning CRBMs and vanilla RBMs are quite different in practice. As the literature suggests, in vanilla RBMs we also find that CD methods work better than MLE-BP.

7 Conclusions and Future Work

In contrast to past work, we argue that belief propagation can be an excellent choice for learning and inference with RBM-based models in the conditional setting. We present a matrix-based expression of the BP updates for CRBMs, which is scalable to tens of thousands of visible and hidden units. Our implementation takes advantage of the bipartite graphical structure and uses a compact representation of messages and beliefs. Since it uses only matrix product and element-wise operations, it is highly suited to GPU acceleration. We demonstrate that learning CRBMs with sum-product BP (MLE) and mixed-product BP (MSSVM) can provide significantly better results than the state-of-the-art CD methods on structured prediction problems. Future directions include a GPU-based implementation and applying the method to deep probabilistic models, such as deep Boltzmann machines.

Acknowledgements

This work is sponsored in part by NSF grants IIS-1254071 and CCF-1331915. It is also funded in part by the United States Air Force under Contract No. FA8750-14-C-0011 under the DARPA PPAML program.

References

D. Ackley, G. Hinton, and T. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive science*, 1985.

J. Gonzalez, Y. Low, and C. Guestrin. Residual splash for optimally parallelizing belief propagation. In *AISTATS*, 2009.

I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. Book in preparation for MIT Press, 2016.

S. Hadjis and S. Ermon. Importance sampling over sets: A new probabilistic inference scheme. In *UAI*, pages 355–364, 2015.

G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 2006.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 2002.

G. E. Hinton. A practical guide to training restricted Boltzmann machines. *UTML TR 2010-003*, 2010.

A. T. Ihler, W. F. John III, and A. S. Willsky. Loopy belief propagation: Convergence and effects of message errors. *JMLR*, 2005.

P. Krähenbühl and V. Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *NIPS*, 2012.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 2001.

H. Larochelle and Y. Bengio. Classification using discriminative restricted Boltzmann machines. In *ICML*, 2008.

H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio. Learning algorithms for the classification restricted Boltzmann machine. *JMLR*, 2012.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

X. Li, F. Zhao, and Y. Guo. Conditional restricted Boltzmann machines for multi-label learning with incomplete labels. In *AISTATS*, 2015.

Q. Liu and A. Ihler. Variational algorithms for marginal MAP. *JMLR*, 2013.

M. Mandel, R. Pascanu, H. Larochelle, and Y. Bengio. Autotagging music with conditional restricted Boltzmann machines. *arXiv:1103.2832*, 2011.

B. M. Marlin, K. Swersky, B. Chen, and N. de Freitas. Inductive principles for restricted boltzmann machine learning. In *AISTATS*, 2010.

V. Mnih, H. Larochelle, and G. E. Hinton. Conditional restricted Boltzmann machines for structured output predition. In *UAI*, 2011.

J. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *JMLR*, 2010.

J. Mooij and H. Kappen. On the properties of the bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2005.

K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, 1999.

C. Peterson and J. R. Anderson. A mean field theory learning algorithm for neural networks. *Complex Systems*, 1987.

W. Ping, Q. Liu, and A. Ihler. Marginal structured SVM with hidden variables. In *ICML*, 2014.

W. Ping, Q. Liu, and A. Ihler. Decomposition bounds for marginal MAP. In *NIPS*, 2015.

A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell. Hidden conditional random fields. *IEEE Transactions on PAMI*, 2007.

R. Salakhutdinov and G. E. Hinton. Deep Boltzmann machines. In *AISTATS*, 2009.

R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. In *ICML*, pages 791–798, 2007.

I. Sutskever and T. Tieleman. On the convergence properties of contrastive divergence. In *AISTATS*, 2010.

G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. In *NIPS*, 2006.

T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *ICML*, 2008.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on information theory*, 2003.

M. Welling and Y. W. Teh. Approximate inference in Boltzmann machines. *Artificial Intelligence*, 2003.

J. Yang, S. Safar, and M.-H. Yang. Max-margin Boltzmann machines for object segmentation. In *CVPR*, 2014.

M. Yasuda and K. Tanaka. Approximate learning algorithm in Boltzmann machines. *Neural computation*, 2009.

C.-N. J. Yu and T. Joachims. Learning structural SVMs with latent variables. In *ICML*, 2009.