Iterative Project Quasi-Newton Algorithm for Training RBM

Shuai Mi and **Xiaozhao Zhao**
watermelon0519@163.com
Tianjin University
No.92, Weijin Road, Nankai District, Tianjin, China

Abstract

The restricted Boltzmann machine (RBM) has been used as building blocks for many successful deep learning models, e.g., deep belief networks (DBN) and deep Boltzmann machine (DBM) etc. The training of RBM can be extremely slow in pathological regions. The second order optimization methods, such as quasi-Newton methods, were proposed to deal with this problem. However, the non-convexity results in many obstructions for training RBM, including the infeasibility of applying second order optimization methods. In order to overcome this obstruction, we introduce an em-like iterative project quasi-Newton (IPQN) algorithm. Specifically, we iteratively perform the sampling procedure where it is not necessary to update parameters, and the sub-training procedure that is convex. In sub-training procedures, we apply quasi-Newton methods to deal with the pathological problem. We further show that Newton's method turns out to be a good approximation of the natural gradient (NG) method in RBM training. We evaluate IPQN in a series of density estimation experiments on the artificial dataset and the MNIST digit dataset. Experimental results indicate that IPQN achieves an improved convergent performance over the traditional CD method.

Introduction

The restricted Boltzmann machine (RBM) is the building blocks for many successful deep learning models, e.g., deep belief networks (DBN) and deep Boltzmann machine (DBM) etc. These models have been successfully applied in computer vision (Bengio et al. 2007; Osindero and Hinton 2008; Poultney et al. 2006), natural language processing (Collobert and Weston 2008), motion-capture data (Taylor, Hinton, and Roweis 2006) and information retrieval (Hinton and Salakhutdinov 2008). A RBM is a non-convex graphical model which contains a layer of visible units and a layer of hidden units. The hidden units give rise to the unidentifiability which in turn directly results in the non-convexity of RBM.

Typically, RBM is trained using Contrastive Divergence (CD) learning, which can be considered as a stochastic gradient descent (SGD) method. This optimization method is concise. However, it is a well known fact in the optimization community that gradient descent is unsuitable for optimizing objectives that exhibit pathological curvature (Martens 2010).

Martens (2010) proposed a second order Hessian-Free optimization method, which can effectively deal with pathological curvature, to train deep auto-encoders and neural networks. Similar to the Hessian-Free method, Desjardins (2013) introduced the Metric-Free Natural Gradient algorithm for training deep Boltzmann machine (DBM). However, the non-convexity of RBM could result in many obstructions, including the infeasibility of applying 2-nd order optimization methods.

We are committed to deal with this obstruction. We introduce a novel iterative project quasi-Newton (IPQN) method for training restricted Boltzmann machine. With the iterative project method, our algorithm transforms the overall training procedure into an iterative procedure, i.e., iteratively perform the sampling procedure where it is not necessary to update parameters, and the sub-training procedure that is convex. We apply the curvature information based second order quasi-Newton methods to the sub-training procedures to deal with the pathological problem. In addition, our algorithm can deal with another training obstruction caused by the non-convexity, i.e., the saddle points whose neighborhood is similar to optima.

Furthermore, we investigate the nature of the Hessian in the RBM training, and based on this nature, two main advantages of our algorithm can be realized. Firstly, in IPQN, the Hessian of the objective function is relatively "well conditioned". Secondly, Newton's method turns out to be a good approximation of the natural gradient method (Amari 1998) in our algorithm. Besides, there are still more advantages of IPQN, i.e., Newton's method can achieve multi-times quadratically convergent stage by applying IP method, and Newton's method is independent of affine changes.

The rest of this paper is organized as follows. Section 2 describes the specific optimization environment of RBM training. Then, the basic IP method is proposed in Section 3. In Section 4, we reveal the natural advantages and motivations of our algorithm. Experimental results on RBM are presented in Section 5. Conclusions and future works are discussed in Section 6.

The Specific Optimization Problems of RBM Training

In this section, we discuss two specific optimization problems of RBM training: the pathological problem and the non-convex problem. A restricted Boltzmann machine is a two-layer network in which the stochastic, binary, visible units $v \in \{0,1\}^N$ are connected to stochastic, binary, hidden units $h \in \{0,1\}^M$. The energy of a joint configuration (**v**,**h**) of the visible and hidden units is given by:

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i \in visible} a_i v_i - \sum_{j \in hidden} b_j h_j - \sum_{i,j} v_i h_j w_{ij} \tag{1}$$

where v_i, h_j are binary state of visible unit i and hidden unit j, respectively, a_i and b_j are bias of v_i and h_j, respectively, and w_{ij} is weight between v_i and h_j. We denote model parameters as $\theta = \{\mathbf{W}, \mathbf{a}, \mathbf{b}\}$ where $\mathbf{a} = (a_1, ..., a_N), \mathbf{b} = (b_1, ..., b_N)$, and $\mathbf{W} = (w_{ij})_{N \times M}$ for $\forall i \in \{1, ..., N\}$ and $\forall j \in \{1, ..., M\}$. The assigned probability of **v** is given by the marginal probability:

$$p(\mathbf{v}) = \frac{1}{Z} \sum_h e^{-E(\mathbf{v},\mathbf{h})}. \tag{2}$$

A practical and fast procedure for estimating the gradient of $\log p(\mathbf{v}$ was proposed in (Hinton 2010):

$$\frac{\partial log p(\mathbf{v})}{\partial w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon} \tag{3}$$

where $\langle \cdot \rangle_{recon}$ represents the expectation with respect to the "reconstruction" which is obtained by initializing with data and running finite times Gibbs sampling. This fast learning procedure is called Contrastive Divergence (CD) method which could be considered as a stochastic gradient descent (SGD) method. For CD method running x times Gibbs sampling, we call it CD-x method. Without loss of generality, we will regard CD-1 as the baseline in this paper. For the sake of brevity, we will denote the objective function $\log p(\mathbf{v}; \theta)$ as $f(\theta)$ in the rest of this paper.

The Pathological Problem of RBM

Considering a "valley" area, in this area, there is one or more "valley" directions where their gradients are nearly vanished while the gradients in the directions along the "cliff" are still existent. A region like this "valley" area is called a pathological region. Typically, RBM is trained using the SGD based CD-1 method. This optimization method is concise. However, it is well known within the optimization community that gradient descent is unsuitable for optimizing objectives that exhibit pathological curvature (Martens 2010). We can use the condition number of the sublevel set of the objective function to measure the pathological degree of RBM training. The condition number has a strong effect on the efficiency of common methods for optimization (Boyd and Vandenberghe 2009). We denote an optimal parameter vector as θ^* and the corresponding optimal value, $\inf_\theta f(\theta) = f(\theta^*)$, as β. When α close to β, the sublevel set is well approximated by an ellipsoid with center θ^*, i.e., $C_\alpha \approx \{\theta | (\theta - \theta^*)^T \nabla^2 f(\theta^*)(\theta - \theta^*) \leq 2(\alpha - \beta)\}$. The condition number of an ellipsoid is the same as the Euclidean condition number of the matrix that defines it (Boyd and Vandenberghe 2009). Therefore

$$\lim_{\alpha \to \beta} cond(C_\alpha) = \kappa(\nabla^2 f(\theta^*))$$

where $\kappa(\cdot)$ represents the Euclidean condition number.

Thus, we can use the condition number of the Hessian matrix at θ^* (i.e., $\nabla^2 f(\theta^*)$) to measure the pathological degree of the objective function around θ^*. It has become apparent that in order to measure the pathological degree of RBM training, we have to evaluate the 2-nd order partial derivative of $\log p(\mathbf{v})$. According to the energy function of RBM, this 2-nd order derivative is given by,

$$\frac{\partial^2 log\, p(\mathbf{v})}{\partial w_{ij} \partial w_{rs}} = \langle v_i h_j v_r h_s \rangle_{data} - \langle v_i h_j v_r h_s \rangle_{model} + \langle v_i h_j \rangle^2_{model} - \langle v_i h_j \rangle^2_{data}$$

where $i, r \in 1, 2, ..., N$ and $j, s \in 1, 2, ..., M$.

We use this equation to evaluate the Hessian matrix at θ^*. Denoting this matrix as H^*, $\kappa(H^*)$ is the measurement of the pathological degree of the RBM training around θ^*. In our experiment on the MNIST data, even when the model contains only 10 hidden units, the condition number reached to 10^{10}(even bigger).

The condition number of the sublevel set ranging up to very large values such as 10^{10} do not adversely affect the Newton's method while the gradient method can be tolerated by a far smaller range of condition numbers (Boyd and Vandenberghe 2009). Thus, applying Newton's method (as well as quasi-Newton methods) is a reasonable way to improve the efficiency of RBM training.

The Non-convexity of RBM

A RBM contains hidden units whose states are unobservable. These hidden units give rise to two space symmetries: the interchange symmetry and the functional symmetry. For a RBM contains M hidden units, the interchange symmetry results in $M!$ equivalent optima (Bishop and others 2006). Furthermore, depending on the specific active function, the functional symmetry could also result in many equivalent optima (Kurková and Kainen 1994). From the statistic viewpoint, a RBM is an unidentifiable model since RBMs with different parameters can achieve the same stationary distribution. This unidentifiability results in the non-convexity of RBM. That is, there are many negative curvature directions in the objective function of RBM.

In the development of his Hessian-Free method, Martens used the Gauss-Newton matrix G (an approximation to the Hessian matrix), instead of the Hessian matrix, to represent the curvature information. There are many reasons for choosing G. One of the most important is that G is guaranteed to be positive semi-definite, even when un-damped, which avoids the problem of negative curvature, thus guaranteeing the optimization process will work (Martens 2010). However, it is not easy to evaluate a positive semi-definite approximation of the Hessian matrix in RBMs as the Gauss-Newton in feed-forward neural networks. However we still

find another way to apply quasi-Newton methods in RBM training. We will introduce the iterative project (IP) method in Section 3. With the iterative project method, our algorithm transforms the overall training procedure into an iterative procedure, i.e., iteratively performing the sampling procedure where it is not necessary to update parameters, and the sub-training procedure that is convex.

Iterative Project Learning for RBM

Amari (1992) proposed an iterative algorithm for training Boltzmann machines with inner layer connections and he also proposed a general iterative framework (i.e., the em framework) for training general neural networks in (Amari 1995). The em framework estimates the hidden variables from the observed or specified input-output data based on the stochastic model. Inspired by this em framework, we implement IP algorithm in RBM training. In this section, we will describe the basic IP algorithm for training RBM.

Given the current parameters θ^i of a RBM and the samples $\underline{\mathbf{v}}$ (we use "$\underline{\cdot}$" to represent samples) that generated from the underlying distribution $q(\mathbf{v})$, the iterative project method could be implemented by iteratively executing the next two procedures:

- a) sampling $\underline{\mathbf{h}}$ from RBM's conditional distribution $\mathrm{p_i}(\mathbf{h}|\mathbf{v};\theta^i)$ given $\underline{\mathbf{v}}$
- b) training a new RBM with those generated samples $(\underline{\mathbf{v}},\underline{\mathbf{h}})$, and then updating the parameter vector of RBM to be the newly trained one, denoted as θ^{i+1};

until the training comes to the convergence. Note that in b) procedures all hidden units in RBM are visible in samples $(\underline{\mathbf{v}},\underline{\mathbf{h}})$. If the observed data specifies a binary state for every unit in the RBM, the model is convex, i.e., there are no non-global optima in the parameter space. Thus, we can reasonably apply quasi-Newton methods in sub-training procedures and the saddle point problem can also be solved.

Quasi-Newton Methods in Sub-training Procedures

We have described the pathological problem of RBM in section 2. Our algorithm is robust to the pathological problem since we proposed to apply quasi-Newton methods in convex sub-training procedures. Besides, there are still more interesting properties of our algorithm. In Section 4.1, we investigate the nature of Hessian in the sub-training procedures and reveal two advantages of our algorithm. Then, we introduce two potential motivations of applying quasi-Newton methods in Section 4.2. At the end of this section, we study the details of implementing quasi-Newton methods in sub-training procedures.

Nature of Hessian in Sub-training

Eq.(3) is just a crude approximation gradient of the log probability of the training data. This "gradient" is much more closely approximating the gradient of Kullback-Liebler divergence (Hinton 2010). Denoting the optimal parameter vector as θ^*, the minimization sequence as $\{\theta^{(k)}\}, k \in 1,2,...,$, the K-L divergence is given by

$$D(\,P(\theta)\|\,Q\,(\theta^*)) = \int f\,(x;\theta)\,log\frac{f\,(x;\theta)}{f\,(x;\theta^*)}dx.$$

For simplicity, we denote the K-L divergence as $D(\theta)$ in this section. Supposing $\theta^{\#}$ is one of the minima in the neighborhood of $\theta^{(k)}$ (i.e., $D(\theta^{\#}) = min\{D(\theta)\}$ for $\forall\theta$ in the neighborhood of $\theta^{(k)}$), the training procedure locally transforms to the minimization of the distance between the distribution specified by the current parameter vector and the distribution specified by $\theta^{\#}$, i.e., finding $\theta^{\#}$ and putting $\theta^{(k+1)} = \theta^{\#}$. This distance is given by,

$$D(\,P(\theta)\|\,P\left(\theta^{\#}\right)) = \int f\,(x;\theta)\,log\frac{f\,(x;\theta)}{f\,(x;\theta^{\#})}dx.$$

Thus, the derivative of D is,

$$\frac{\partial D}{\partial\theta_i} = \int\frac{\partial f\,(x;\theta)}{\partial\theta_i}logf\,(x;\theta) + f(x;\theta)\frac{\partial logf(x;\theta)}{\partial\theta_i}dx - \int\frac{\partial f\,(x;\theta)}{\partial\theta_i}logf\left(x;\theta^{\#}\right)dx,$$

the 2-nd derivative of D is,

$$\begin{aligned}\frac{\partial^2 D}{\partial\theta_i\partial\theta_j} &= \int[\frac{\partial^2 f\,(x;\theta)}{\partial\theta_i\partial\theta_j}logf\,(x;\theta) + \frac{\partial f\,(x;\theta)}{\partial\theta_i}\frac{\partial logf\,(x;\theta)}{\partial\theta_j} \\ &\quad + \frac{\partial f\,(x;\theta)}{\partial\theta_j}\frac{\partial logf\,(x;\theta)}{\partial\theta_i} + f\,(x;\theta)\,\frac{\partial^2 logf\,(x;\theta)}{\partial\theta_i\partial\theta_j}]dx \\ &\quad - \int\frac{\partial^2 f\,(x;\theta)}{\partial\theta_i\partial\theta_j}logf\left(x;\theta^{\#}\right)dx \\ &= \int\frac{\partial^2 f\,(x;\theta)}{\partial\theta_i\partial\theta_j}[logf\,(x;\theta) - logf\left(x;\theta^{\#}\right)]dx \\ &\quad + \int[\frac{\partial f\,(x;\theta)}{\partial\theta_i}\frac{\partial logf\,(x;\theta)}{\partial\theta_j} + \frac{\partial f\,(x;\theta)}{\partial\theta_j}\frac{\partial logf\,(x;\theta)}{\partial\theta_i}]dx \\ &\quad + \int f\,(x;\theta)\,\frac{\partial^2 logf\,(x;\theta)}{\partial\theta_i\partial\theta_j}dx\end{aligned}$$

we denote the first term as X, i.e.,

$$\int\frac{\partial^2 f\,(x;\theta)}{\partial\theta_i\partial\theta_j}[logf\,(x;\theta) - logf\left(x;\theta^{\#}\right)]dx = X \quad (4)$$

the second term satisfies the equation,

$$\begin{aligned}&\int[\frac{\partial f\,(x;\theta)}{\partial\theta_i}\frac{\partial logf\,(x;\theta)}{\partial\theta_j} + \frac{\partial f\,(x;\theta)}{\partial\theta_j}\frac{\partial logf\,(x;\theta)}{\partial\theta_i}]dx \\ &= 2\mathrm{E}[\frac{\partial logf\,(x;\theta)}{\partial\theta_i}\frac{\partial logf\,(x;\theta)}{\partial\theta_j}] \quad (5)\end{aligned}$$

since

$$\frac{\partial f\,(x;\theta)}{\partial\theta_i}\frac{\partial logf\,(x;\theta)}{\partial\theta_j} = f\,(x;\theta)\,\frac{\partial logf\,(x;\theta)}{\partial\theta_i}\frac{\partial logf\,(x;\theta)}{\partial\theta_j}$$

$$\frac{\partial f\,(x;\theta)}{\partial\theta_j}\frac{\partial logf\,(x;\theta)}{\partial\theta_i} = f\,(x;\theta)\,\frac{\partial logf\,(x;\theta)}{\partial\theta_i}\frac{\partial logf\,(x;\theta)}{\partial\theta_j};$$

The last term satisfies the equation,

$$\int f(x;\theta)\frac{\partial^2 log f(x;\theta)}{\partial\theta_i\partial\theta_j}dx$$
$$= \mathrm{E}[\frac{\partial^2 log f(x;\theta)}{\partial\theta_i\partial\theta_j}] = -E[\frac{\partial log f(x;\theta)}{\partial\theta_i}\frac{\partial log f(x;\theta)}{\partial\theta_j}] \quad (6)$$

From Eq.(4), (5) and (6) we can see that for any θ in the neighborhood of $\theta^{\#}$, the relation between the Hessian matrix and the Fisher information matrix (FIM) is given by,

$$\frac{\partial^2 D}{\partial\theta_i\partial\theta_j} = E[\frac{\partial log f(x;\theta)}{\partial\theta_i}\frac{\partial log f(x;\theta)}{\partial\theta_j}] + X$$

i.e.,

$$H = F + X \quad (7)$$

where F is the FIM, and H is the Hessian matrix.

From above derivations, we can find two advantages of our algorithm:

Relatively "well conditioned" H: F is a positive semi-definite matrix (actually, in most cases, F is a positive definite matrix). Note that the manifold of the sub-training procedure is convex and the searching is in the neighborhood of the local minimum $\theta^{\#}$. Thus, X is relatively small. According to Eq.(7), H is relatively "well conditioned".

Approximation of NG: the sub-training procedures are convex and $\theta^{\#}$ is a minimum (not necessary to be a stationary point) in the neighborhood of $\theta^{(k)}$. Thus, $log f(x;\theta)$ is approaching to $log f\left(x;\theta^{\#}\right)$ once the searching comes to an area where θ is close to $\theta^{\#}$, i.e.,

$$log f(x;\theta) - log f\left(x;\theta^{\#}\right) \approx 0$$

From Eq.(4) and above derivations, we can see that X will vanish along with the searching. According to Eq.(7), we have $H \approx F$. Therefore, it turns out that Newton's method shares the same optimization direction with natural gradient method since

$$\begin{aligned} \mathrm{t_{Newon}} &= -H^{-1}\nabla D(\theta) \\ \mathrm{t_{NG}} &= -F^{-1}\nabla D(\theta) \end{aligned}$$

where t_{Newton} and t_{NG} represent the optimization direction of Newton method and natural gradient method, respectively. Newton's method turns out to be a good approximation of the natural gradient method in the RBM training. Thus, the excellent properties of the natural gradient method, e.g., the steepest descent direction and the Fisher-efficient estimation, are succeeded by Newton's method (as well as quasi-Newton methods).

Potential Motivations

Here we focus on two potential motivations of applying quasi-Newton methods in sub-training procedures.

Multi-times quadratically convergent stage: the iterations in Newton's method naturally fall into two stages. The optimization convergent very fast when it comes to the second stage which is called the quadratically convergent stage or the pure Newton stage. The second stage occurs once the gradient reduced to a threshold (i.e., the searching comes to an area that is close enough to the optima). The bound of the number of iterations in the quadratically convergent stage grows extremely slowly with required accuracy, and can be considered as a constant for practical purposes, say five or six (Boyd and Vandenberghe 2009). Note that by applying the IP method, the overall training achieves multi-times quadratically convergent stage since each sub-training procedure is a minimization of K-L divergence.

Independent of affine changes: Newton's method is independent of affine changes of coordinates. In other word, if we change the coordinates, for Newton's method, the iterates are related by the same change (Boyd and Vandenberghe 2009). It would be interesting to change the current coordinates of the model to a suitable one which could make the manifold smoother.

Implement of Quasi-Newton Method in Sub-training Procedures

Given all this, we have motivations and necessary conditions to apply quasi-Newton methods in sub-training procedures. There are many quasi-Newton methods, and in this paper we apply the Hessian-Free (HF) method introduced by Martens (Martens 2010). Denoting the initial parameter vector of the sub-training as ϵ_0 and the optimization step as t, the algorithm is given by,

---

Algorithm 1 $subtraining\_QN(\epsilon_0, samples\_vh)$ ($samples\_vh$: samples generated in the sampling procedure of IP)

---

1) Use $samples\_vh$ to estimate the gradient $\nabla D(\epsilon_i)$ via Gibbs sampling;

2) Solve the system $Ht = \nabla D(\epsilon_i)$ via the finite difference and the conjugate gradient method;

3) Update the parameters of the sub-training: $\xi_{i+1} = \xi_i + t$;

4) Repeat 1), 2), 3) until the tolerance is reached.

When the training comes to the basin of the optimum, i.e., the quadratically convergent stage occurs, we can use a fixed step size 1 (Boyd and Vandenberghe 2009). We did not adapt the learning rate via line search since in each sub-training procedure the initial distance between the initial parameter vector and the optimum is much smaller than that of the overall training. In our experiments, even using the step size 1 in the early stage, we still achieve relatively effective convergence.

Experimental Study

In this section, we experimentally investigate the IPQN algorithm in density estimation tasks for restricted Boltzmann machines. In our experiments, we used the artificial data and the MNIST digit data. We compare iterative project quasi-Newton method (IPQN) with Contrastive Divergence (CD-1), iterative project (IP) and Hessian-Free (HF), and show that in our experiments IPQN achieves better convergence.

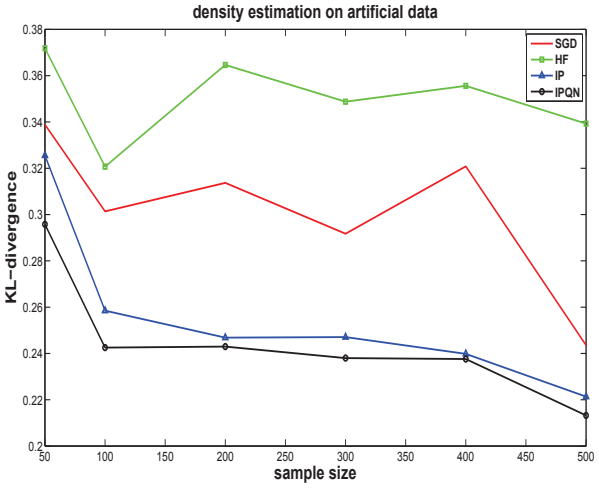

Figure 1: Density estimation for RBM

Experiments on the Artificial Data

The artificial binary dataset: we first randomly select the target distribution $q(x)$, which is randomly chosen from the open probability simplex over the n random variables using the Dirichlet prior (with parameters alpha =0.5). Then, the dataset with N samples are generated from $q(x)$. In order to accurately and effectively evaluate the contrastive divergence, the artificial dataset is set to be 10-dimensional. For experiments on the artificial data, the baseline is CD-1, and other three methods are compared:

- IPQN: we change the overall training procedure into an iterative procedure using the iterative project method and apply quasi-Newton methods in sub-training procedures;
- IP: we change the overall training procedure into an iterative procedure using the iterative project method and apply CD-1 in sub-training procedures;
- HF: we apply Hessian-Free method without changing the overall training procedure into an iterative procedure.

K-L divergence is used to evaluate the goodness-of-fit of the RBM trained by these four algorithms. For all experiments, we run 20 randomly generated distributions and report the average K-L divergence. The sample size is from 50 to 500. Note that our experiments focus on the case that variable number is relatively small ($n = 10$) in order to analytically evaluate the contrastive divergence.

The average K-L divergences between RBMs and the underlying real distributions are shown in Fig.1. Among these four algorithms, IPQN achieves better result. The performance of HF is even worse than CD-1 because of the non-convexity of RBM.

Intuitively, along with the increasing of the sample size the estimated probability distribution should be closer to the underlying distribution. However, we can see that the average performance of CD-1 fluctuates wildly. This observation illustrated that CD-1 is more likely to achieve a fluctuant convergence under some "special" distributions even when the sample is relatively sufficient. On the other hand, the performances of two algorithms involved with IP method (i.e. IP and IPQN) improve reasonably along with the increasing of the sample size. Thus, we can experimentally find that the em based IP method is more likely to achieve a stable convergence. Two reasons of the stable convergence are introduced here: first, gradient based methods, such as CD-1, are proved to be an approximation of IP method (Zhao et al. 2013). Second, the monotonicity of IP method can be theoretically guaranteed (Zhao et al. 2013).

The average performance of IPQN is better than IP especially under relatively small sample size (i.e. N = 50,100)(see Fig.1). To explain this performance difference, we have theoretically explained in Section 2 and Section 4 that quasi-Newton methods can avoid "hang-up" in pathological curvature regions and achieve a Fisher-efficient optimum. Fig.2 shows the training trajectories (which are averaged over 20 distributions) of the initial a few seconds of IPQN procedure and IP procedure. Note that we focus on the initial a few seconds because we want to obtain the average trajectory while the training on different distributions would reach the convergence after different number of steps. From Fig.2, we can see that IPQN is much faster than IP when the sample size is relatively small and this difference is less significant when the sample is relatively sufficient. Two reasons result in this observation:

- A RBM can be considered as a non-linear function. A function with relatively less constraints is more likely to be ill-posed. In the same way, a RBM specified by relatively insufficient sample is more likely to be pathological.
- IPQN algorithm pass through the pathological curvature regions rapidly while IP exploring slowly.

Experiments on the MNIST Digit Data

In this subsection, we experimentally investigate the performance of IPQN on real-world datasets in the context of density estimation. We use log-likelihood to evaluate the goodness-of-fit of the training since it is not easy to compute the contrastive divergence due to the high dimensionality. We used a RBM contains 20 hidden units to learn the distribution density over the MNIST digit data. In our experiments, the training set consists of 10000 cases and the test set consists of 10000 cases. Four algorithms are compared: CD-1, HF, IP and IPQN.

CD-1 and HF takes less time to achieve convergence, but from Fig.3 we can see that IPQN achieves better log-likelihood(-175.243) than CD-1(-183.792) and HF(-192.233). Furthermore, we can see that the trajectories of IPQN and IPs are stable while the trajectory of CD-1 is extremely fluctuant. This observations agree with the experiments on the artificial data.

We compare IPQN with IPs under different settings (i.e. learning rate is set to be 0.5, 0.05 and 0.005). We can see from Fig.3 that IP(0.5) converges quickly, but the corresponding convergent value is relatively low. Even the lower learning rate IP(0.05), which is much slower, cannot reach

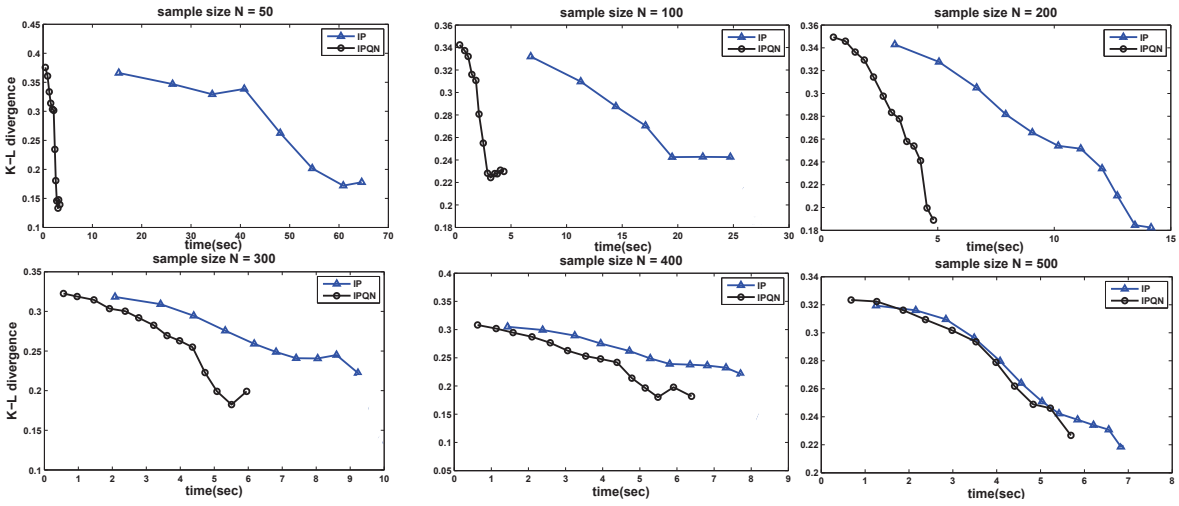

Figure 2: Trajectory of RBM training

the same convergent value as IPQN. Meanwile, the IP with learning rate 0.005 is "hang-up" because of the under-fitting.

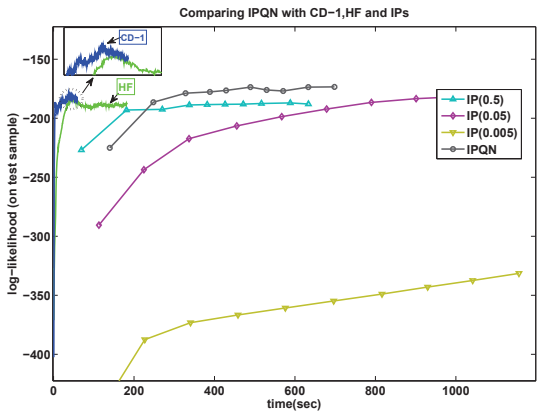

Figure 3: Trajectoris of MNIST experiments.

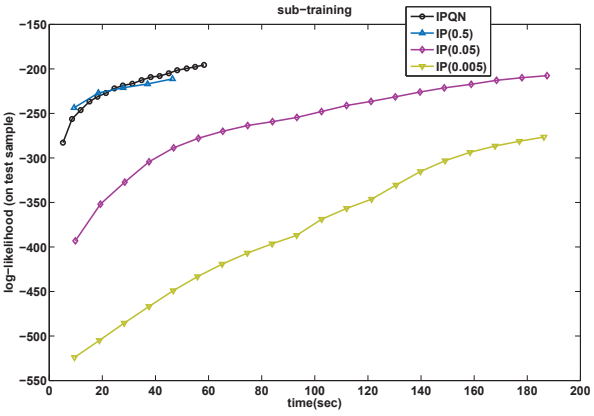

Figure 4: The first sub-training procedure

In order to investigate the superiority of IPQN in pathological regions, we also observed the trajectories of IPQN and IP in sub-training procedures. Without loss of generality, we show trajectories of the first sub-training in Fig.4. We can see that IP(0.05), which achieves the best convergence in these three IPs, takes about 50 seconds to achieve the same log-likelihood as IPQN achieves after the first ten seconds. After this, IP(0.05) can only obtain a small per-epoch improvement since the learning rate is set to be relatively low to pass through the pathological regions. However, IPQN still obtain a relatively larger improvement in the pathological regions. This observation illustrates that IPQN is more efficient in the pathological problem.

Conclusions and Future Works

We propose an IPQN algorithm for training RBM. Specifically, we implement IP method to deal with the non-convex problem, and in the sub-training procedures of IP we apply quasi-Newton methods to deal with the pathological problem. Our main contributions are following three folds. First, we reveal the fact that in the RBM training, the Hessian matrix consists of the fisher information matrix and a matrix X. Second, in our algorithm, Hessian of the objective function is relatively "well conditioned" since the X is relatively small in the convex sub-training procedures. Third, we find that when the optimization approaching to the minimum of the neighborhood of the current parameter, Newton's method turns out to be a good approximation of the natural gradient method. Furthermore, our algorithm can achieve multi-times quadratically convergent stage, and be independent of affine coordinate changes. Finding a suitable coordinate will be left as future works.

On the experimental side, we compared IPQN with CD-1, HF and IP. In summary, IPQN achieves better convergence than CD-1, HF and IP. Directly applying HF to RBM training cannot fully stretch the advantages of quasi-Newton methods because of the non-convexity. Comparing IP based methods (i.e. IP and IPQN) with CD-1, IP based methods

achieve a stabler convergence and a better convergent value.

References

Amari, S.-I.; Kurata, K.; and Nagaoka, H. 1992. Information geometry of boltzmann machines. *Neural Networks, IEEE Transactions on* 3(2):260–271.

Amari, S.-I. 1995. Information geometry of the em and em algorithms for neural networks. *Neural networks* 8(9):1379–1408.

Amari, S.-I. 1998. Natural gradient works efficiently in learning. *Neural computation* 10(2):251–276.

Bengio, Y.; Lamblin, P.; Popovici, D.; and Larochelle, H. 2007. Greedy layer-wise training of deep networks. *Advances in neural information processing systems* 19:153.

Bishop, C. M., et al. 2006. *Pattern recognition and machine learning*, volume 1. springer New York.

Boyd, S., and Vandenberghe, L. 2009. *Convex optimization*. Cambridge university press.

Collobert, R., and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, 160–167. ACM.

Desjardins, G.; Pascanu, R.; Courville, A.; and Bengio, Y. 2013. Metric-free natural gradient for joint-training of boltzmann machines. *arXiv preprint arXiv:1301.3545*.

Hinton, G. E., and Salakhutdinov, R. 2008. Using deep belief nets to learn covariance kernels for gaussian processes. In *Advances in neural information processing systems*, 1249–1256.

Hinton, G. 2010. A practical guide to training restricted boltzmann machines. *Momentum* 9(1):926.

Kurková, V., and Kainen, P. C. 1994. Functionally equivalent feedforward neural networks. *Neural Computation* 6(3):543–558.

Martens, J. 2010. Deep learning via hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 735–742.

Osindero, S., and Hinton, G. E. 2008. Modeling image patches with a directed hierarchy of markov random fields. In *Advances in neural information processing systems*, 1121–1128.

Poultney, C.; Chopra, S.; Cun, Y. L.; et al. 2006. Efficient learning of sparse representations with an energy-based model. In *Advances in neural information processing systems*, 1137–1144.

Taylor, G. W.; Hinton, G. E.; and Roweis, S. 2006. Modeling human motion using binary latent variables. In *Advances in Neural Information Processing Systems*, 2007. MIT Press.

Zhao, X.; Hou, Y.; Yu, Q.; Song, D.; and Li, W. 2013. Understanding boltzmann machine and deep learning via a confident information first principle. *arXiv preprint arXiv:1302.3931*.