Fine-Tuning Parameters of Deep Belief Networks
Using Artificial Bee Colony Algorithm

Ming-Huwi Horng
Department of Computer Science and Information Engineering, National PingTung University

Keywords: Deep belief network, Fine-tuning, Artificial bee colony Algorithm.

Abstract. The Deep Belief network has become a powerful tool in nowadays to large-scale-oriented application, however, there are several parameters need to assign in advances that is key factors of successive application. In this paper, we proposed to address the issue of properly fine-tuning parameters of Deep Belief Networks by means of Artificial Bee Colony (ABC) algorithm. Experimental results show that the proposed ABC algorithm can effectively reconstruct the original binary images.

Introduction
The fine-tuning parameters in DBN learning can be modeled as an optimization problem, in which the objective or fitness function as criterion for search for the optimal solution as well as the best parameter for constructing the DBNs. The meta-heuristic algorithms are among the most used ones for optimization problem, and such techniques can be divided into the stochastic and nature-inspired approaches. Currently, there are a few works that deal with the fine-tuning parameters of DBNs using the nature-inspired approaches such as Harmony search [5], cuckoo search [3] and firefly algorithm [4] in the literature. As far as we know, the artificial bee colony algorithm has never applied to fine-tuning the DBN parameters. In this paper, we apply the artificial bee colony algorithm to deal with the fine-tuning problem.

Deep Belief Networks
The concepts related to Deep Belief Networks, particularly, with an attention to the background of RBMs, which are the basis for DBN understanding. In essence, the Restricted Boltzmann Machine is an energy-based stochastic neural networks composed by two layers of neurons that are visible and hidden nodes, in which the learning phase is conducted by means of an unsupervised fashion. The structure of a Restricted Boltzmann Machine consists of a visible layer \( v \) with \( m \) nodes and a hidden layer \( h \) with \( n \) nodes.

The \( m \times n \) matrix \( W \) models the weights between visible and hidden layers, in which \( w_{ij} \) stands for the weight between the visible node \( v_i \) and the hidden node \( h_j \). Let us supposed that \( v \) and \( h \) are the binary and hidden units, respectively, \( v \in \{0,1\}^m \) and \( h \in \{0,1\}^n \). The energy function of RBM is given by:

\[
E(v,h) = - \sum_{i=1}^{m} a_i v_i - \sum_{j=1}^{n} b_j h_j - \sum_{i=1}^{m} \sum_{j=1}^{n} v_i h_j w_{ij}
\]

(1)

where the vector \( a \) and \( b \) are the basis vectors of visible and hidden layers, further, the probability of the a given configuration \( (v, h) \) is calculated as follows:

\[
P(v,h) = \frac{e^{-E(v,h)}}{\sum_{v,h} e^{-E(v,h)}}
\]

(2)

The denominator stands for all possible configurations \( (v, h) \), that is herein a normalization factor.

The parameters \( a, b \) and \( W \) of RBM can be optimized by using stochastic gradient scent on the log-likelihood of training data patterns. The probability of a given sample can be computed over all possible hidden vectors by (3).
\[
P(v) = \frac{\sum_y e^{-E(v,y)}}{\sum_y e^{-E(v,y)}}
\]

Stochastic gradient ascent algorithm computes the derivatives of the logarithm of \( P(v) \) with respected to \( a, b \) and \( W \), and then leading to equations as follows:

\[
\begin{align*}
W^{t+1} &= W^t + \eta \left( (h|v)v^T - P(\tilde{h}|\tilde{v})\tilde{v}^T \right) - \lambda W^t + \alpha \Delta W^{t-1} \\
a^{t+1} &= a^t + \eta (v - \tilde{v}) + \alpha \Delta a^{t-1} \\
b^{t+1} &= b^t + \eta \left( P( (h|v) - P(\tilde{h}|\tilde{v}) \right) + \alpha \Delta b^{t-1}
\end{align*}
\]

where

\[
P(h_j = 1|v) = \text{sigm}(\sum_{i=1}^{n} w_{ij} v_i + b_j)
\]

and

\[
P(v_j = 1|h) = \text{sigm}(\sum_{i=1}^{n} w_{ij} h_i + a_j)
\]

where \( \eta, n, \lambda, \alpha \) are the parameters of learning rate, the number of hidden nodes, weight decay and momentum weight. The \( \text{sigm}(.) \) stands for the logistic sigmoid function. Three learning algorithms: Contrastive Divergence (CD) and Persistent Contrastive Divergence (PCD) are applied to learning the weight matrix and the corresponding bias vectors of the visible and hidden nodes. Truly speaking, we have a DBN consisted of \( L \) layers, the \( W^i \) the weight matrix of RBM at layer \( i \). In addition, we can find the hidden units at layer \( I \) are the input units to layer \( i+1 \). The fine-tuning steps are proposed by Hinton et al. [1] to train each RBM. The proposed procedure is performed by means of a back-propagation method and Gradient descent algorithm to adjust the matrices \( W^i, i=1,2, \ldots, L \). The optimization to find the minimization of error measure considering the output of an additional layer placed on the top of the DBN after its former greedy training such as the softmax or logistic units.

**Artificial Bee Colony (ABC) Algorithm**

The artificial bee colony algorithm was proposed by Karaboga and Basturk [2]. In this algorithm, the position of a food source \( z_i \) represents a possible solution to the optimization problem and the amount of nectar in a food source corresponds to the fitness \( \text{fit}(z_i) \) to the corresponding solution. The ABC approach models each food source (candidate solution) as real-valued vector with \( 4L \) dimension in DBN fine-tuning. The \( N \) candidate solutions correspond to the \( 4L \) dimensional parameters of the DBN with \( L \) layers. The number of employed or onlooker bees is generally equal to the number of solutions of population. Initially, the ABC algorithm randomly generated a distributed population \( P \) of \( N \) solutions, in which \( N \) denoted the number of employed bees or onlooker bees. Each solution \( z_i \) is a \( D \)-dimension vector, herein; \( D \) is the number of optimization parameters. In each execution cycle, the population of the solution is updated according to the search processes of the employed, onlooker and scout bees. A employed bee modified the its possible solution depending on the amount of nets (fitness value) of the new source (new solution) by Eq. (9).

\[
z_{ij} = z_{ij} + \sigma_{ij} (z_{ij} - z_{kj})
\]

where \( z_k \) is a random solution different from the current solution \( z_i \) and the \( j \in \{1,2, \ldots, D\} \); the \( \sigma_{ij} \) is a random number between [-1, 1].

If the amount of nectar of the new solution is than the previous one, this bee will remember the new position and forget the old one, otherwise it retains the location of the previous one when all employed bees had finished their search process, they deliver the nectar information and the position of food source to the onlooker bees, each of the whom chooses a food source to onlooker.
according to a probability which is proportional to the amount of nectar in that food source. The probability \( p_i \) of selecting a food source \( z_i \) is computed using the following Eq. (10)

\[
p_i = \frac{f_i^{(z_i)}}{\sum_{z_i} f_i^{(z_i)}}
\]

(10)

In practical terms, any food source \( z_i \) sequentially generates a random number ranged from 0 to 1, if the random number is less than \( p_i \), an onlooker bee are sent to food source \( z_i \) and generates a new solution based on Eq. (9). If the fitness of the new solution is more than the old one, the looker bee memories the new one and shares information with other bees, otherwise the new solution will be discarding. The process is repeated until all onlooker bees have been distributed to the food source and produces their corresponding new solutions. If the position of food source cannot be improved through the predetermined number of the ‘limit’ of bees, the food resource \( z_i \) is abandoned and then the employed source is and \( j \in \{1,2,...,D\} \), then the scout discovers a new food source ro be replaced with \( z_i \). This operation can be defined as in Eq. (11).

\[
z_{ij} = z_{\text{min}}^j \pm \text{rand}(0,1)(z_{\text{max}}^j - z_{\text{min}}^j)
\]

(11)

where the \( z_{\text{max}}^j \) and \( z_{\text{min}}^j \) are upper bound and upper bound of the \( j \)th component of all solutions. If the new solution is better than the abandoned one, the scout will become an employed bee. The selection of employed, onlooker bees and scouts is repeated until the termination criteria are satisfied.

**Proposed Approach**

In general, the Restricted Boltzmann Machines require the setting up of four main parameters which are the learning rate \( \eta \), number of the hidden units \( n \), momentum \( \phi \) and weight decay \( \lambda \). The designed the Deep Belief Network composed by the \( L \) Layers of RBMs has \( 4L \) variables to be optimization. In this paper we used the following ranges of concerning parameters that are \( n \in [5,100] \), \( \eta \in [0.1,0.9] \), \( \lambda \in [0.1,0.9] \) and \( \phi \in [0.001,0.01] \). In short, the ABC algorithm aims to search for the set of DBN parameters that minimizes the mean squared error measure (MSE) of in the experiments, i.e.:

\[
\text{MSE} = \frac{1}{\eta} \sum_{t=1}^{N} (\tilde{I}_t - I_t)^2
\]

(12)

where \( \tilde{I}_t \) and \( I_t \) stand for \( i^{th} \) reconstructed and original images, respectively. After that, the selected set of parameters is then applied to reconstruct the test images. The training procedure of DBNs is greedy-wise, which means each layer is trained independently; only four variables are optimized in each layer.

**Experimental Results and Discussion**

In experiments we compared the ABC algorithm with other well-known Firefly algorithm, Cuckoo search and Harmony search, which are usually used meta-heuristic algorithms. In experiments, we employed 5 agents over 50 iterations for convergence considering all techniques. In experiments we have conducted a hold-out procedure with 20 randomly generated training and test sets, 10 iterations for the learning procedure of each RBM, and mini-batches of size 20 for consistent comparisons with other works listed in Table 1.

<table>
<thead>
<tr>
<th>Training methods</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firefly algorithm</td>
<td>( \gamma = 1.0, \beta_0 = 1.0, \alpha = 0.2, \text{MCN} = 100 )</td>
</tr>
<tr>
<td>Harmony search</td>
<td>( \text{HMCR}=0.7, \text{PAR}=0.7, \eta = 1.0 )</td>
</tr>
<tr>
<td>Cuckoo search</td>
<td>( \alpha = 0.1, p_a = 0.25 )</td>
</tr>
<tr>
<td>ABC algorithm</td>
<td>( N=30, \text{MCN}=100, \text{limit}=20 )</td>
</tr>
</tbody>
</table>
Table 2 presents the MSE for each optimization technique over the test set considering DBNs with one, two and three layers for the MNIST dataset. We used the Wilcoxon signed-rank test with $\alpha = 0.05$ for statistical analysis. One clearly found the ABC and Firefly algorithms obtained the best results among the four methods, but the shortcoming of the two algorithms need a large of computation times to converge.

Table 2. Average MSE over the test set with MNIST dataset.

<table>
<thead>
<tr>
<th>Technique</th>
<th>1L</th>
<th>2L</th>
<th>3L</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CD</td>
<td>PCD</td>
<td>CD</td>
</tr>
<tr>
<td>Firefly algorithm</td>
<td>0.0876</td>
<td>0.0876</td>
<td>0.0876</td>
</tr>
<tr>
<td>Harmony search</td>
<td>0.1059</td>
<td>0.1325</td>
<td>0.1059</td>
</tr>
<tr>
<td>Cuckoo search</td>
<td>0.1066</td>
<td>0.1066</td>
<td>0.1076</td>
</tr>
<tr>
<td>ABC algorithm</td>
<td><strong>0.0874</strong></td>
<td><strong>0.0874</strong></td>
<td>0.0881</td>
</tr>
</tbody>
</table>

Conclusions

In experiments of this paper, we carried out three other methods that are Firefly algorithm, harmony search and Cuckoo search method in two public databases in the context of image reconstruction using DBNs with 1, 2 and 3 layers for comparison. The experimental results show that the ABC algorithm and Firefly algorithm obtained the better results than other two methods as using much less layers. It still is an interesting topic to enhance the two algorithms for fine-tuning of deep belief network.

Acknowledgments

The author is grateful to Ministry of Science and Technology, R.O.C. under Grant No. MOST 105-2221-E-153 -012 for support of this work.

Reference


