PRML 2018 spring
Solution to Assignment #4

June 10, 2018

Ziyuan Feng
15307130194
zyfeng15@fudan.edu.cn

Problem

Model MNIST dataset with Restricted Boltzman Machine. Train parameters and then generate samples.

Model Definition

Given a dataset $\{x_i\}_{i=1}^N$ where x_i is a m dimensional vector. Suppose there is a Restricted Boltzman Machine with m visible random variables and n latent random variables. Therefore, we have

- visible random vector $\mathbf{v} \in \{0,1\}^m$
- latent random vector $\mathbf{h} \in \{0,1\}^n$
- weight matrix $\mathbf{W} \in \Re^{m \times n}$
- visible bias $\mathbf{a} \in \Re^m$
- latent bias $\mathbf{b} \in \Re^n$

Thus, the energy of the RBM given $\mathbf{v}$ and $\mathbf{h}$ is

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{a}^T\mathbf{v} - \mathbf{b}^\mathbf{h} - \mathbf{v}^T\mathbf{W}\mathbf{h} \tag{1}$$

The probability distribution of latent variables given visible variables is

$$p(\mathbf{h} = 1|\mathbf{v}) = \sigma(\mathbf{W}^T\mathbf{v} + \mathbf{b}) \tag{2}$$

where σ is the element-wise sigmoid function.
The probability distribution of visible variables given latent variables is

$$p(\mathbf{v} = 1|\mathbf{h}) = \sigma(\mathbf{W}\mathbf{h} + \mathbf{a}) \tag{3}$$

Parameter Learning

Constractive divergence with one step (**CD-1**) is used to train RBM parameters.

1. pick up a training sample as visible input $\mathbf{v_0}$
2. compute latent variable distribution $p(\mathbf{h_0} = 1|\mathbf{v_0})$ with (2)
3. sample $\mathbf{h_0}$ from $p(\mathbf{h_0} = 1|\mathbf{v_0})$
4. compute visible variable distribution $p(\mathbf{v_1} = 1|\mathbf{h_0})$ with (3)
5. sample reconstructed visible variables $\mathbf{v_1}$ from $p(\mathbf{v_1} = 1|\mathbf{h_0})$
6. compute $p(\mathbf{h_1} = 1|\mathbf{v_1})$ with (2)
7. sample $\mathbf{h_1}$ from $p(\mathbf{h_1} = 1|\mathbf{v_1})$
8. update $\mathbf{W}$ with $\mathbf{W} + \alpha(\mathbf{v_0}\mathbf{h_0}^T - \mathbf{v_1}\mathbf{h_1}^T)$
9. update $\mathbf{a}$ with $\mathbf{a} + \alpha(\mathbf{v_0} - \mathbf{v_1})$
10. update $\mathbf{b}$ with $\mathbf{b} + \alpha(\mathbf{h_0} - \mathbf{h_1})$
11. repeat 1-10 over all training samples
12. repeat 1-11 for T epochs

continued on next page...

Sampling

The process of Gibbs sampling on RBM is described below.

1. define initial visible input $\mathbf{v_0}$
2. repeat 3-6 for T times, with $t = 1, \dots T$
3. compute latent variable distribution $p(\mathbf{h_{t-1}} = 1|\mathbf{v_{t-1}})$ with (2)
4. sample $\mathbf{h_{t-1}}$ from $p(\mathbf{h_{t-1}} = 1|\mathbf{v_{t-1}})$
5. compute visible variable distribution $p(\mathbf{v_t} = 1|\mathbf{h_{t-1}})$ with (3)
6. sample $\mathbf{v_t}$ from $p(\mathbf{v_t} = 1|\mathbf{h_{t-1}})$
7. return $\mathbf{v_T}$ and $\mathbf{h_{T-1}}$

In my RBM implementation, step 3 and 4 are called a "forward" pass in which data flows from visible units to hidden units, and step 4 and 5 are called a "backward" pass as reverse.
Essentially, the sampling of RBM is iteratively applying forward and backward passes, while the CD-1 method contains two forward passes and one backward pass.
One more detail: the way to sample binary vectors from a given distribution is first generating a random vector from uniform distribution and then setting each of its elements to 0 or 1 by comparing with the given probability. That is what step 4 and 6 do in the codes.

Experiments

MNIST dataset consists of 60000 images of shape 28×28. To model the dataset distribution, a restricted Boltzman machine with 28×28 visible units is used. However, the number of hidden units is an essential hyper-paramter. If it is too small, the model may not be capable to learn the distribution. If it is too large, it will take quite a long time to train.

Efficiency

For RBM, training takes much longer time than sampling. In fact, the efficiency of training is bottlenecked by matrix multiplication from equation (2) and (3). It is not easy to develop algorithmic parallelism to optimize (2) and (3). Hence, **multithreading** is applied to achieve data-level parallelism. Experiments have shown a massive speedup in training.

Parameter Setting 1

hidden units n	2
training epochs T	8

With 4 threads, the average training time is 53.39s per epoch.

continued on next page...

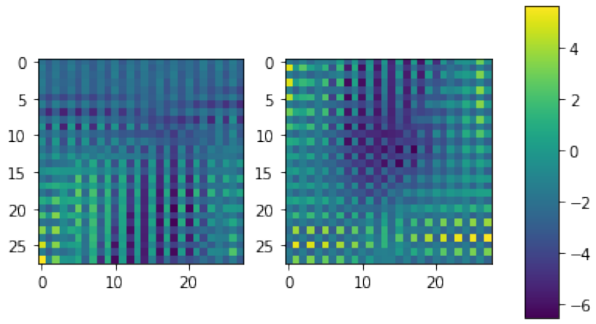

Figure 1: Visualization of two weight matrices: $\mathbf{W}[:, 0]$ (left) and $\mathbf{W}[:, 1]$(right)

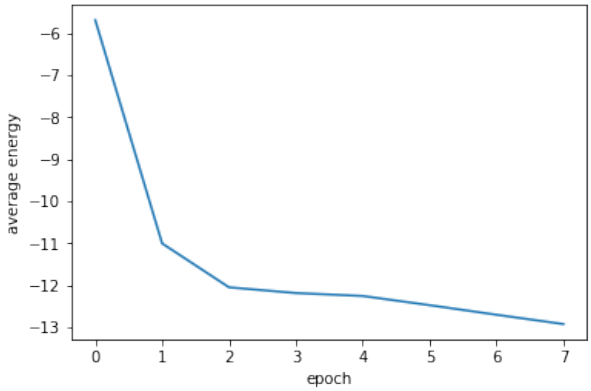

Figure 2: The average energy of RBM decreases.

Figure 1 shows how weight matrices of RBM turn activated after training. It turns out that they both have some dark colored regions where activations from visible units tend to be ignored. However, $\mathbf{h}[1]$ received activations from the edge of the matrix.
Figure 2 shows the decrease of average energy over all training samples during 8 epochs. The curve seems not converged but it has dropped much slower after the third epoch.

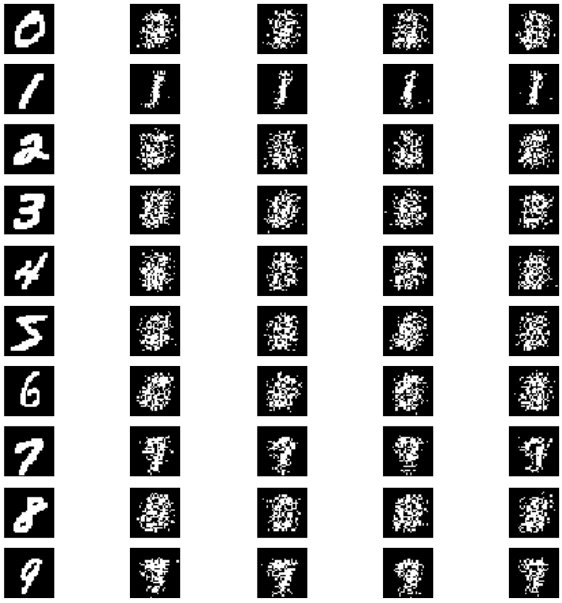

Figure 3: Generate digit 0 to 9. The first column is the initial visible input from MNIST. The other columns are outputs of Gibbs sampling.

Figure 3 shows the sampling from RBM. All these results are not distinguishable from other digits except the digit 1. It shows that the RBM with only 2 hidden units is not capable to model MNIST data.

continued on next page...

Parameter setting 2

hidden units n	100
training epochs T	10

With 5 threads, the average training time is 175.15s per epoch.

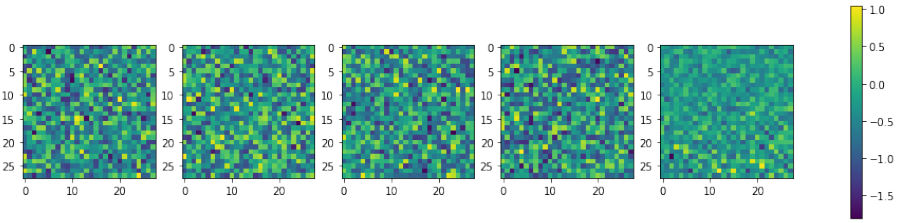

Figure 4: Visualization of five weight matrices: $\mathbf{W}[:, i], i = 0 \dots 4$

Figure 4 shows the weight matrices present no particular distributions. Activated pixels are uniformly or randomly distributed. That means the model has learnt an abstract representation of those images.

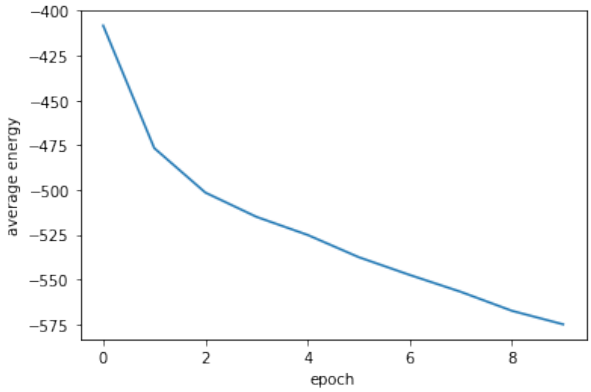

Figure 5: The average energy of RBM decreases.

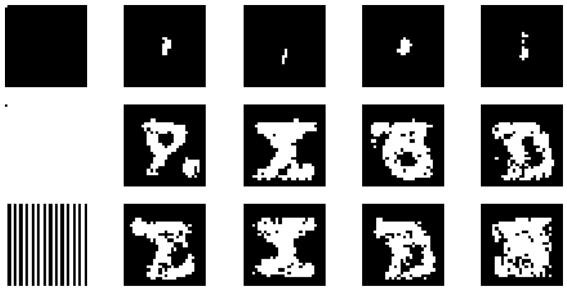

Figure 6: Sampling with special inputs (the first column).

It is found that the sampling results from RBM is influenced by the initial visible input. Figure 6 shows the results from specific initial inputs. For example, in the first row, the model is trying to reproduce the black image with as less white pixels as possible.

continued on next page...

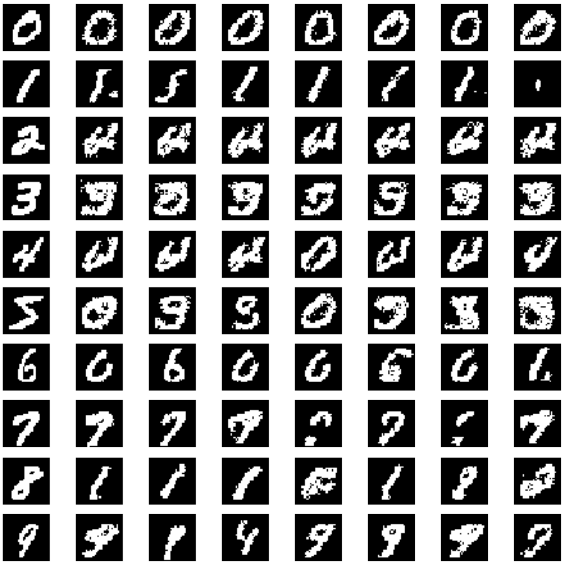

Figure 7: Generate digit 0 to 9. The first column is the initial visible input from MNIST. The other columns are outputs of Gibbs sampling.

Figure 7 shows that given such inputs, the model can well generate digit 0, 1, 6, 7, 9. But it cannot model the others. It fails to distinguish between digit 2 and 4, digit 3 and 9, etc.

Parameter setting 3

hidden units n	200
training epochs T	50

Only 1000 training samples from MNIST are used in this setting.

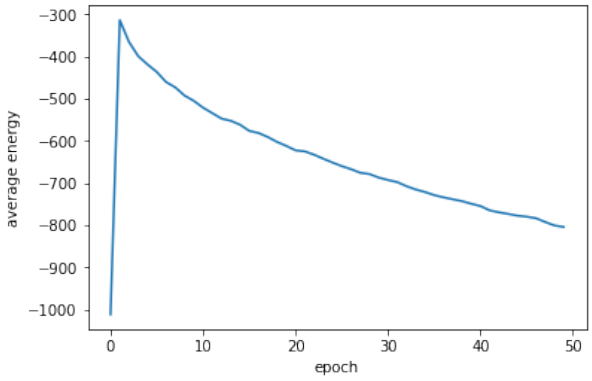

Figure 8: The average energy of RBM decreases, except the first epoch. It may be caused by random initialization of weight matrices.

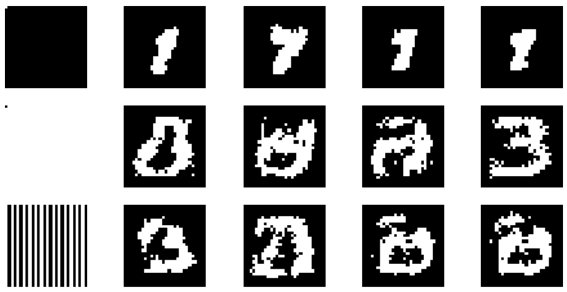

Figure 9: Sampling with special inputs (the first column). In the second and third row, the model tries to produce as many white pixels as possible.

continued on next page...

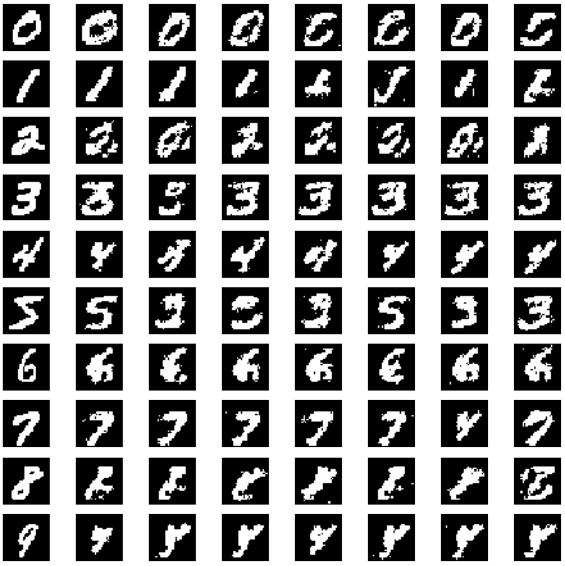

Figure 10: Generate digit 0 to 9. The first column is the initial visible input from MNIST. The other columns are outputs of Gibbs sampling.

With such setting, the model can learn most of the digits quite well, such as digit 2, 3, 4, and 5. But it cannot sample digit 8 correctly.
This result is achieved with less training samples but more epochs, which is a balance between efficiency and accuracy.

References

[1] Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.

[2] Xipeng Qiu. Deep Learning and Neural Network. https://nndl.github.io. 2018.