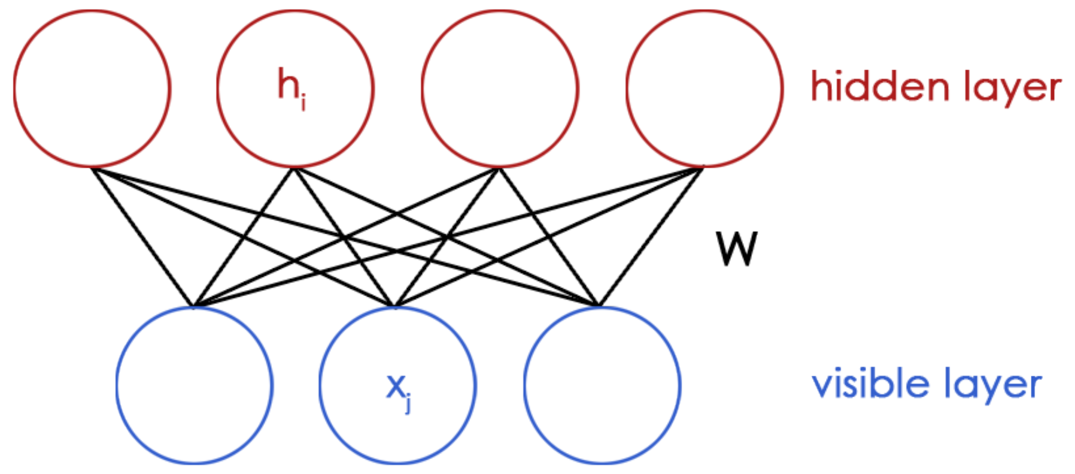
Restricted Boltzmann Machine

{ Chaïmaa Kadaoui, Othman Sbai and Xi Shen }   MVA - ENS Paris Saclay

Definition

A Restricted Boltzman Machine is an undirected graph which can be decomposed in two layers.



It is *restricted* because no connection between nodes of the same layer is supposed. The matrix W characterizes the connections (weights) between the two layers. For simplification, we suppose that the variables $\mathbf{x}$ and $\mathbf{h}$ are binary. This model is called Bernoulli RBM.
The hidden variables can be seen as features and the goal of the RMB is to represent meaningful features.

Energy and Probability

We introduce the two bias vectors c and b to define the energy of this graph:

$$E(x,h) = -h^\top W x - c^\top x - b^\top h$$

We can write the joint probability of $\mathbf{x}$ and $\mathbf{h}$ as:

$$p(x,h) = \exp(-E(x,h))/Z$$

Z is the partition parameter which can be computed by calculating all the possibles values of $\mathbf{x}$ and $\mathbf{h}$. In practice, this parameter is intractable.
Inference We can prove that given $\mathbf{x}$, h_j follows a Bernoulli:

$$p(h_j = 1|x) = sigm(b_j + W_{j.}x)$$

$W_{j.}$ is the j-th row of W and $sigm$ defines the sigmoid function. We can prove a similar result for $p(x_i = 1|h)$.

Contrastive Divergence

To train the RBM with our data $\mathbf{x}^{(t)}$, we would like to minimize the average loss: $\frac{1}{T}\sum_t l(f(\mathbf{x}^{(t)})) = \frac{1}{T}\sum_t -\log p(\mathbf{x}^{(t)})$
We will apply a stochastic gradient descent. We derive each term of the sum with respect to our model parameter θ as follow (where $\mathbb{E}_h$ and $\mathbb{E}_{x,h}$ are expectations of h and (x,h) respectively):

$$\frac{\partial -\log p(\mathbf{x}^{(t)})}{\partial\theta} = \mathbb{E}_h\left[\frac{\partial E(\mathbf{x}^{(t)},h)}{\partial\theta}|\mathbf{x}^{(t)}\right] - \mathbb{E}_{x,h}\left[\frac{\partial E(\mathbf{x},h)}{\partial\theta}\right]$$

The first term is called the *positive phase* and the second term the *negative phase*. Because of the difficulty to compute the second term, we will use an algorithm called *Contrastive Divergence*. The key points of the algorithm are:

- We estimate the expectation $\mathbb{E}_{x,h}$ by sampling a single point $\tilde{\mathbf{x}}$.
- To do so, we use Gibbs sampling in chain (we apply it k times).
- We initialize our Gibbs sampling with $\mathbf{x^{(t)}}$.
- Update parameters W, b, c; with $h(x) = p(h|x) = sigm(b + Wx)$ and α the learning rate.
  - $W = W + \alpha\left(h(\mathbf{x^{(t)}})\mathbf{x^{(t)}}^\top - h(\tilde{\mathbf{x}})\tilde{\mathbf{x}}^\top\right)$
  - $b = b + \alpha\left(h(\mathbf{x^{(t)}}) - h(\tilde{\mathbf{x}})\right)$
  - $c = c + \alpha\left(\mathbf{x^{(t)}} - \tilde{\mathbf{x}}\right)$

Applications of RBM

- RBM is a generative model, but it can be used to improve classification tasks: instead of training on the raw data $\mathbf{x}$, we can use the values of the hidden layer $\mathbf{h}$.
- RBM are also used for collaborative filtering: given a set of N users and M movies, we would like to recommend movies to the users.
- RBM are helpful for computer vision tasks such as object recognition, image denoising and inpainting.

References

[1] Asja Fischer and Christian Igel. Training restricted boltzmann machines: An introduction. *Pattern Recognition*, 47(1):25–39, 2014.

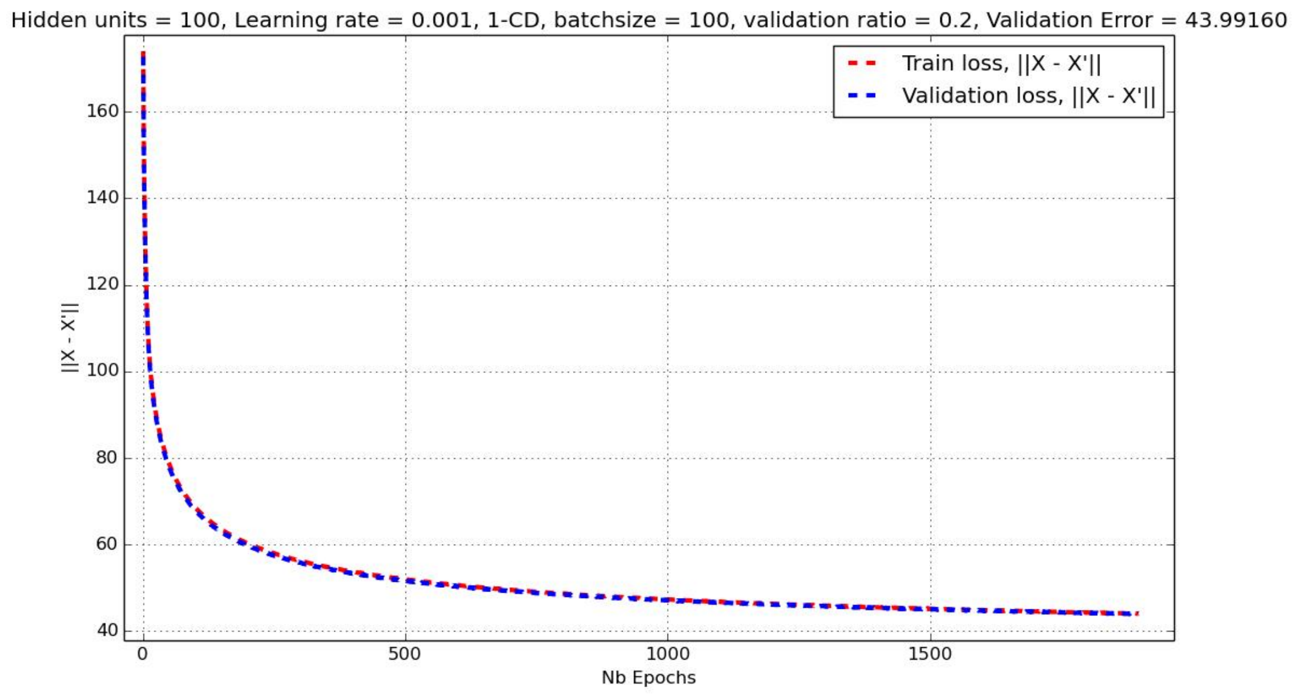
[2] Geoffrey Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 9(1):926, 2010.

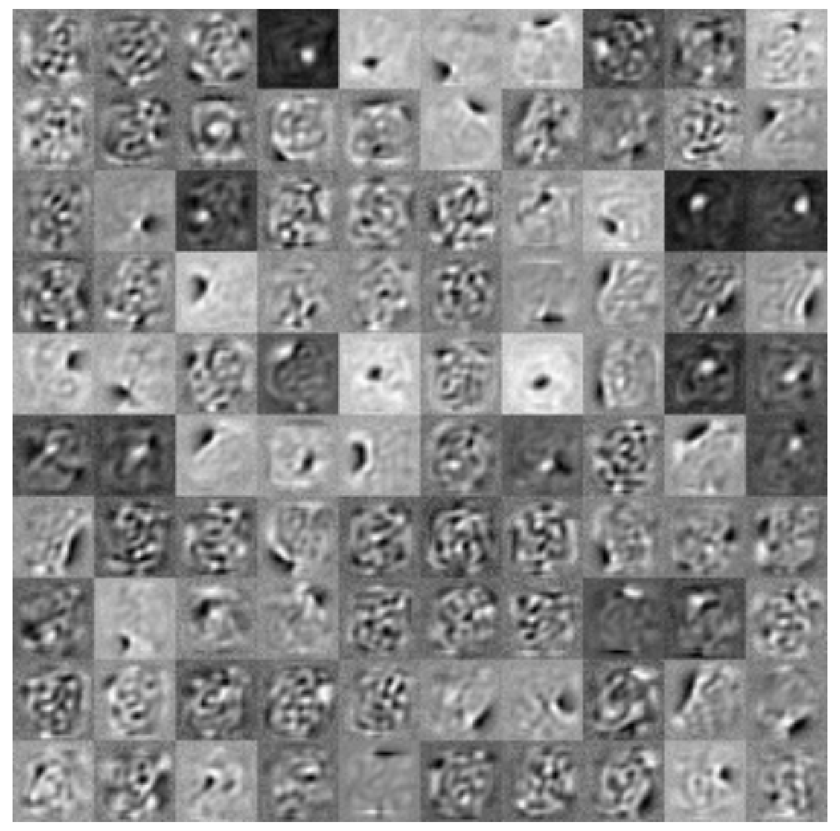
[3] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071. ACM, 2008.

Results On MNIST data

We implemented the RBM training algorithm both using numpy and using tensorflow, we applied it to the MNIST dataset (images of handwritten digits). In the case of image data, each pixel is considered as a node; the intensity $p \in [0,1]$ of a pixel is considered as a probability.

The left figure is the evolution of the average stochastic reconstruction $\|\mathbf{x^{(t)}} - \tilde{\mathbf{x}}\|^2$ for both the training and the test dataset at each iteration. The right figure is the visualization of the weights between one hidden unit and each element of the input vector.

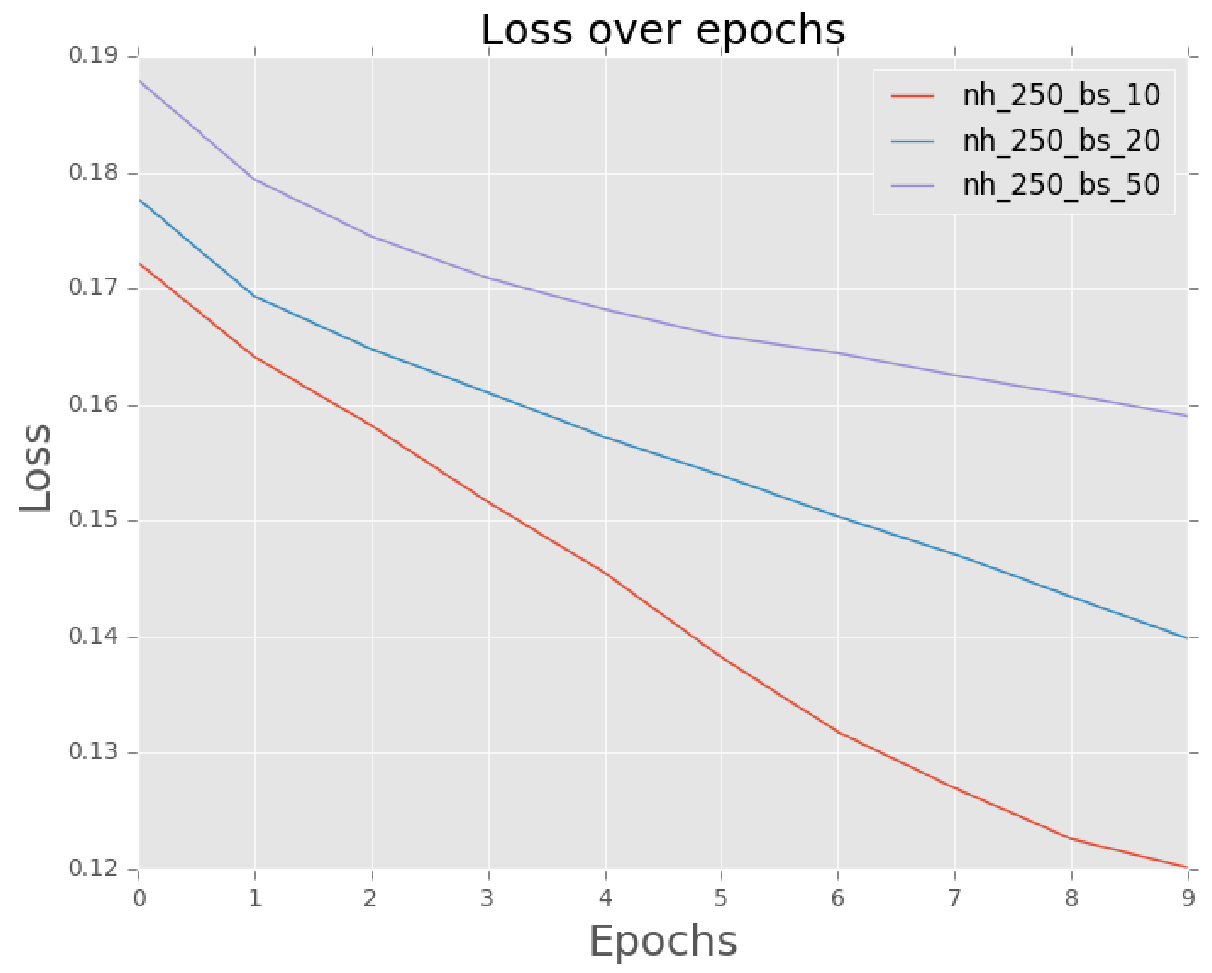


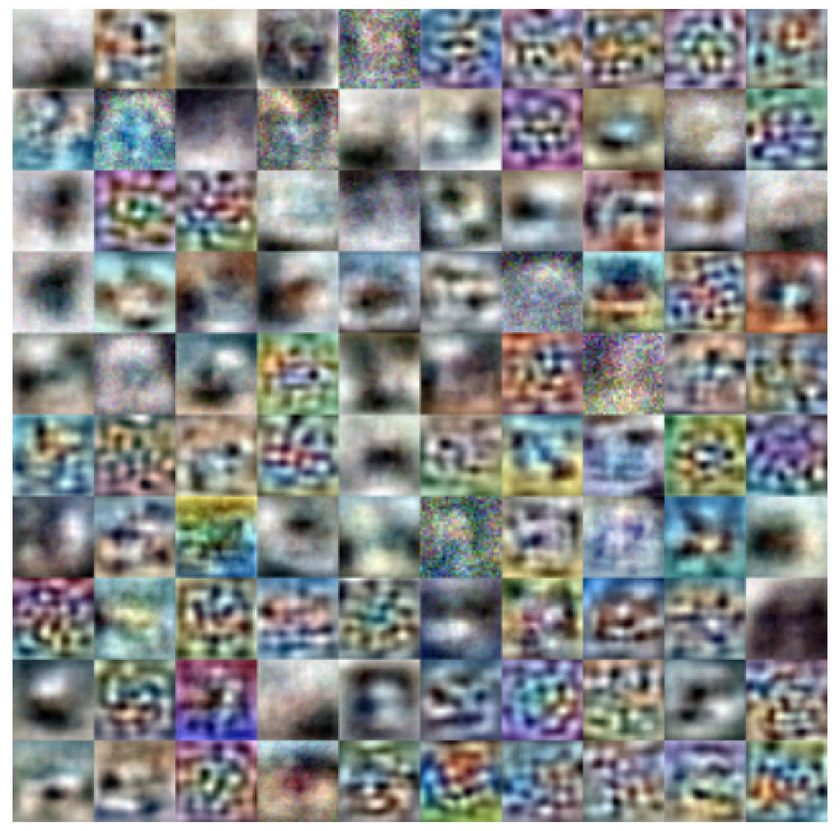


Results On CIFAR data

Using tensorflow, we applied the algorithm to the CIFAR dataset (a dataset of colour images in different classes).

The left figure is the evolution of the average stochastic reconstruction at each iteration for different settings of the algorithm. The right figure is the visualization of the weights between one hidden unit and each element of the input vector.





Conclusion

- Increasing the number of hidden layers improves the performances but the risk of overfitting increases. We found that about 250 hidden layers is a good trade-off for the MNIST and CIFAR datasets.
- The more steps we use for Gibbs sampling, the lower is the bias of our estimate. However it takes more time to compute. An alternative of the Contrastive Divergence algorithm is called Persistent Contrastive Divergence: at each iteration of the algorithm, we initialize the Gibbs sampling with the sample from the previous iteration.
- It is possible to adapt the RBM model for unbounded reals by adding a quadratic term ($\frac{1}{2}\mathbf{x}^\top\mathbf{x}$) to the energy. Then, $p(\mathbf{x}|h)$ becomes a Gaussian distribution. This is called Gaussian-Bernoulli RBM. However, this model is less stable.