The Role of Different Sampling Methods in Improving Biological Activity Prediction Using Deep Belief Network

Fahimeh Ghasemi,[a] Afshin Fassihi,[b] Horacio Pérez-Sánchez,[c] and Alireza Mehri Dehnavi*[a]

Thousands of molecules and descriptors are available for a medicinal chemist thanks to the technological advancements in different branches of chemistry. This fact as well as the correlation between them has raised new problems in quantitative structure activity relationship studies. Proper parameter initialization in statistical modeling has merged as another challenge in recent years. Random selection of parameters leads to poor performance of deep neural network (DNN). In this research, deep belief network (DBN) was applied to initialize DNNs. DBN is composed of some stacks of restricted Boltzmann machine, an energy-based method that requires computing log likelihood gradient for all samples. Three different sampling approaches were suggested to solve this gradient. In this respect, the impact of DBN was applied based on the different sampling approaches mentioned above to initialize the DNN architecture in predicting biological activity of all fifteen Kaggle targets that contain more than 70k molecules. The same as other fields of processing research, the outputs of these models demonstrated significant superiority to that of DNN with random parameters. © 2016 Wiley Periodicals, Inc.

DOI: 10.1002/jcc.24671

Introduction

The years from the late 1990s to the early 2000s can be regarded as the golden period of using artificial neural network (ANN) for the prediction or classification of molecular bioactivity in computer-aided drug design. The attention ANN received was mostly as to its convenience for studying a few molecules and descriptors as well as its simplicity in training. Recently, the number of biologically active molecules and molecular descriptors has raised exponentially. Parallel to this increment, deep neural network (DNN) which is a multilayer perceptron network with many hidden layers and plenty of nodes in each layer could not overcome prone to over-fitting and getting stuck in local minima problems in drug discovery the same as other research area such as image processing and speech processing.[1,2]

A variety of chemo-informatics research studies were proposed to overcome mentioned problems. Alessandro Lusci et al. (2013) showed how recursive neural network approaches can be applied to the problem of predicting molecular properties.[3] In 2014, Thomas Unterthiner et al. compared the performance of deep learning approach used in seven target prediction methods on the ChEMBL database. They indicated that deep learning outperformed all other methods with respect to the area under the curve (AUC) and was significantly better than all commercial products.[4] Junshui Ma et al. (2015) maintained that DNN can routinely make better prospective predictions than random forest on a set of large diverse quantitative structure activity relationship (QSAR) datasets.[5] In 2015, Hughes utilized a database of 702 epoxidation reactions to build a deep machine learning network and identified the site of epoxidation with 94.9% AUC performance and separated epoxidized and non-epoxidized molecules with 79.3% AUC.[6]

When applied in drug design, DNN approach also raises a problem: the output models are sometimes not satisfactory when the initial parameters utilized in constructing network, weights and biases, are chosen randomly.[7] In this study, to solve this and other previous mentioned problems, the new algorithm was proposed that invented by Hinton in 2006 named deep belief network (DBN). DBN is a kind of novel generative unsupervised learning algorithm that caused revival DNN in image processing.[8] DBN attempts to begin DNN learning procedure by the optimum rectified initial parameters, weights and biases, instead of random parameters. Hierarchical learning, in which higher level features are constructed by lower level ones, is the main purpose of DBN. In fact, this approach is composed of multiple levels of linear and nonlinear operations.[9] The number of these operations, called the model depth, refers to the longest path from an input node to an output one.[10] The main application of this network is to recognize the best values for initializing the learning procedure to fine-tune the weights and biases of ANN. The commonly used method in constructing each layer of DBN is restricted Boltzmann machine (RBM).[11]

[a] F. Ghasemi, A. Mehri Dehnavi
Department of Bioelectric and Biomedical engineering, School of Advanced Technologies in Medicine, Isfahan University of Medical Sciences, Hezar-Jerib Ave, Isfahan 81746 73461, IRIran
E-mail: mehri@med.mui.ac.ir

[b] A. Fassihi
Department of Medicinal Chemistry, School of Pharmacy and Pharmaceutical Sciences, Isfahan University of Medical Sciences, Hezar-Jerib Ave, Isfahan 81746 73461, IRIran

[c] H. Pérez-Sánchez
Computer Science Department, Universidad Católica San Antonio de Murcia (UCAM), Murcia E30107, Spain

Contract grant sponsor: Isfahan University of Medical Sciences; Contract grant number: 394883

RBM is an energy-based method used as a discriminative or generative model for labeled or unlabeled data. It has a single layer of hidden units with no internal layer of visible and hidden neurons. To generate data with this method, it is necessary to compute log likelihood gradient for all visible and hidden samples.[12] To solve this problem, Gibbs sampling method was suggested. Gibbs sampling method is a Markov chain Monte Carlo introduced by Geman (1984) to obtain a sequence of observations approximated from the joint probability distribution of two or more random variables.[13] This iterative approach is a randomizing algorithm and guarantees obtaining the best results from the model. It is obvious that running this algorithm for many steps is too time consuming to be practical.[11,14] Hinton et al. (2006) introduced a fast greedy algorithm that quickly produces a fairly good set of parameters, even in deep networks with millions of parameters and many hidden layers using contrastive divergence (CD) method.[8] The CD training algorithm is a special Gibbs sampling method that starts the Gibbs chain from real data points rather than random values.[14,15] The CD algorithm based on the number of steps is demonstrated as CD_n. CD_1 is fast but significantly different from the likelihood gradient. When there is enough time for computation, CD_10 is generally shown to be better.[16] Teileman (2008) presented another gradient approximation algorithm named persistent contrastive divergence (PCD) or Stochastic Maximum Likelihood (SML) which is the same as Gibbs sampling algorithm. Only on the first step, PCD is run with random variables while for other iterations of Gibbs chain, it is initialized by the previous step rather than by random values. It was shown that this algorithm produces more meaningful feature detectors, and outperforms the other algorithms.[16] In 2009, PCD was improved by Teileman and Hinton to provide fast persistent contrastive divergence (FPCD) algorithm obtained by decoupling the parameters used in positive and negative phases.[17] In this model, adding the new parameter, "fast," to the update step caused the weights and biases to learn rapidly and caused the model to be optimized to reach the global minimum.[11]

In this respect, it was decided to study the impact of DBN based on different sampling approaches, namely CD, PCD, and FPCD, to initialize the DNN architecture in biological activity prediction of compounds. First of all, three different DBNs were constructed. Then, they were applied to pre-train input data and fine-tune the weights and biases of DNN, separately. In our proposed algorithm, to achieve the best evaluation, all compounds of each dataset were clustered into ten groups based on their biological activity values. Then, 75% of each group was selected randomly as training sets, and the rest were retained for test sets. Deep learning algorithms are very beneficial when parallel programming with Matlab is used. All fifteen different targets of Kaggle competition were utilized in this study. The best results of this competition are related to George Dahl et al. with mean square of correlation (R^2) of 0.49 over all datasets using DNN with random initialization.

Table 1. Fifteen different Kaggle datasets.

Dataset index	Dataset	Number of molecules	Number of descriptors
ACT1	3A4	50,000	9491
ACT2	CB1	11640	5877
ACT3	DPP4	8327	5203
ACT4	HIVINT	2421	4306
ACT5	HIVPROT	4311	6274
ACT6	LOGD	50,000	8921
ACT7	METAB	2092	4595
ACT8	NK1	13482	5803
ACT9	OX1	7135	4730
ACT10	OX2	14875	5790
ACT11	PGP	8603	5135
ACT12	PPB	11622	5470
ACT13	RAT_F	7821	5698
ACT14	TDI	5559	5945
ACT15	THROMBIN	6924	5552

Methods

Datasets

The personal computer used for all calculations and modeling was an Intel-based core i7 (CPU at 3.20 GHz) with Windows operating system (win 7, 64 bit) and Geforce GRX760 graphic card. The proposed model was implemented in Matlab (2010). The database that was utilized in this article was downloaded from a Kaggle competition database (www.kaggle.com). Kaggle competition was held by Merck sponsor (2012) to compare all machine learning methods used in QSAR approaches. In this study, all 15 targets of the Kaggle database were chosen, all containing molecular activity and molecular descriptors listed in Table 1. Descriptions of each target can be found at Ref. [5].

Deep belief network

DBN is a class of deep generative models composed of l stacks of RBM. The main goal of DBN is the weight initialization of a DNN to produce optimum model in comparison to the model by random weights. This approach makes the predictions extremely effective. Conversely, DBN can be effectively used to perform layer by layer pre-training intended to initialize training of a back propagation algorithm.[10]

The energy-based probabilistic model is a common method among different approaches used in generative deep networks. It can be used to make a joint distribution between observed data, l, and hidden variables, h, for each layer as follows[10]:

$$P\left(l, h^1, \ldots, h^m\right)= \left(\prod_{i=1}^{m-2} P\left(h^i|h^{i+1}\right)\right).P\left(h^{m-1}|h^m\right), \qquad (1)$$

where $l = h^0$, $P\left(h^i|h^{i+1}\right)$ is a conditional distribution for hidden-hidden units in an RBM related to the k^{th} level of the DBN, and $P(h^{m-1}|h^m)$ is the hidden by hidden joint distribution in top-level RBM.[10]

In each layer, weights and biases were calculated and the output was used as an input for the next layer.

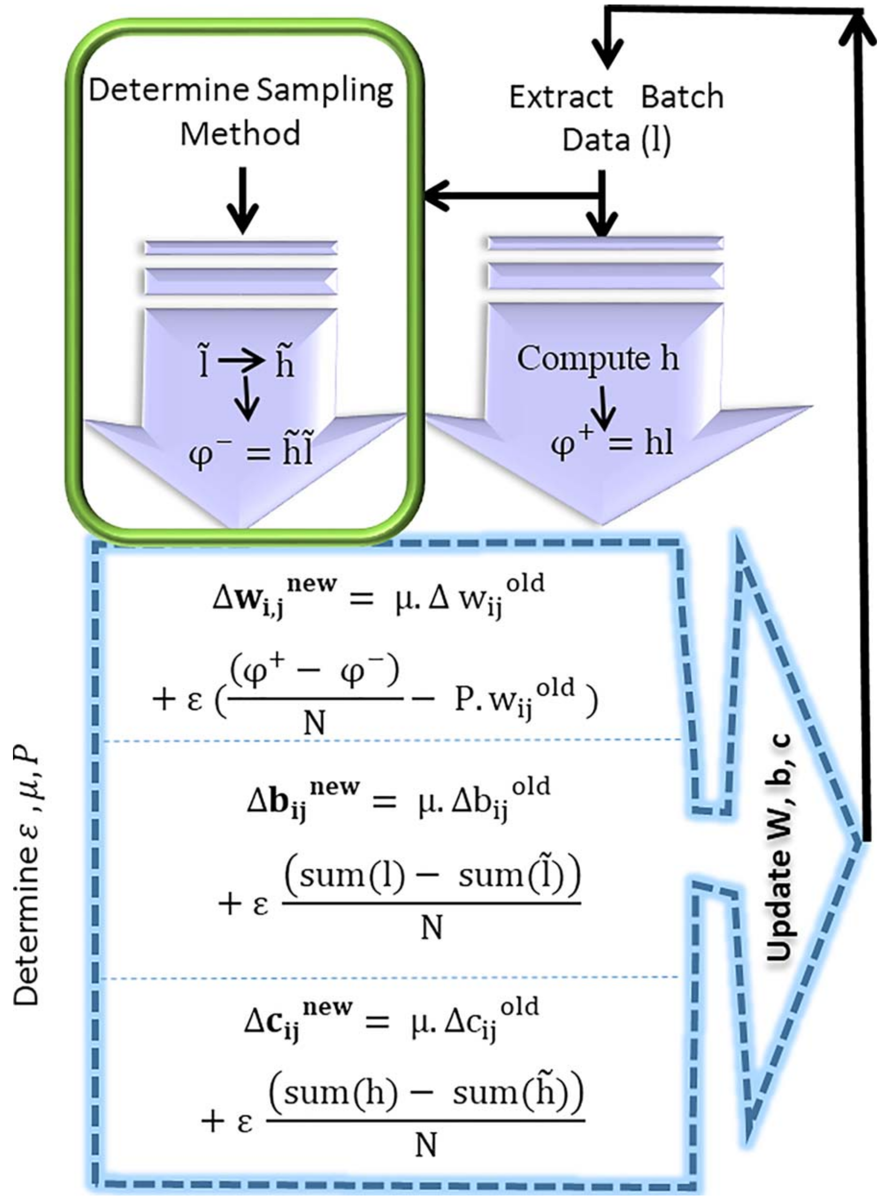
Determine Sampling Method
Extract Batch Data (l)
$\tilde{l} \rightarrow \tilde{h}$
$\varphi^{-} = \tilde{h}\tilde{l}$
Compute h
$\varphi^{+} = hl$
$\Delta w_{i,j}^{new} = \mu.\Delta w_{ij}^{old} + \varepsilon\left(\frac{(\varphi^{+} - \varphi^{-})}{N} - P.w_{ij}^{old}\right)$
$\Delta b_{ij}^{new} = \mu.\Delta b_{ij}^{old} + \varepsilon\frac{(\text{sum}(l) - \text{sum}(\tilde{l}))}{N}$
$\Delta c_{ij}^{new} = \mu.\Delta c_{ij}^{old} + \varepsilon\frac{(\text{sum}(h) - \text{sum}(\tilde{h}))}{N}$
Determine ε, μ, P
Update W, b, c

Figure 1. The scheme of all steps of RBM. They were repeated for batch number stages. [Color figure can be viewed at wileyonlinelibrary.com]

Restricted Boltzmann machine

RBM is a generative statistical model utilized to estimate the output by probability distribution of inputs. RBM has established applications in speech recognition, image processing, drug design, etc. It is a kind of Boltzmann machine with no internal layer connection within both visible and hidden layers. RBM is a particular type of energy-based model, mapping each configuration of input variable to a scalar energy. In this model, the probability of joint configuration (*l*,*h*) is defined as follows[10,18]:

$$\Pr(l,h)=\frac{\exp(-\text{Energy}(l,h))}{Z}, \tag{2}$$

where $Z=\sum_{i,j}\exp\left(-\text{Energy}\ (l_i,h_j)\right)$ is called normalization factor. The energy function of joint configuration (*l*,*h*) can be defined as:

$$\text{Energy}(l,h)=-\beta(l)+\sum_{j}\gamma_j(l,h_j), \tag{3}$$

$$\beta(l)=-bl$$

$$\gamma_j(l,h_j)=-h_j(c_j-W_jl),$$

where *b* and *c* vectors are biases of visible and hidden variables, respectively, and *W* is the corresponding weight matrix for the connection between the input and hidden variables. The probability of visible unit is achieved by summation of all hidden units.

$$P(v)=\frac{1}{Z}\sum_{h}\exp(-\text{Energy}(l,h)) \tag{4}$$

The derivative of the logarithm of probability equation mentioned above acts in a bootstrap manner.

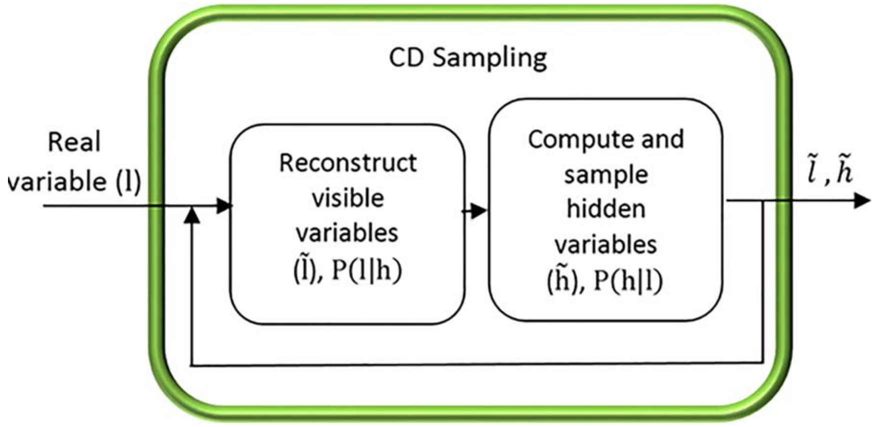
CD Sampling
Real variable (l)
Reconstruct visible variables (l̃), P(l|h)
Compute and sample hidden variables (h̃), P(h|l)
$\tilde{l}, \tilde{h}$

Figure 2. All steps of CD sampling method. The output of the positive phase is applied for initializing the negative phase. The green rectangle is related to the Figure 1. [Color figure can be viewed at wileyonlinelibrary.com]

$$\frac{\partial \log\left(P(v)\right)}{\partial\theta} = \frac{\partial \sum_h \exp\left(-\mathrm{Energy}(l,h)\right)}{\partial\theta} - \frac{\partial \log(Z)}{\partial\theta} = \varphi^{+} - \varphi^{-}, \tag{5}$$

where φ^{+} and φ^{-} are named as positive and negative phases, respectively.

As to the lack of internal connection between visible or hidden units, estimating the positive phase is simple. Input variables are initialized by random variables, and hidden units are set as binary values, $h_i \in [0,1]$, with 0.5 assigned to threshold.[10] Therefore, the conditional probability for any pair of hidden units is defined as:

$$P\left(h_j = 1|l\right) = \frac{e^{c_j + W_j l}}{1 + e^{c_j + W_j l}} = sigm\left(c_j + \sum W_j l\right), \tag{6}$$

where W_j is j^{th} row of W and $sigm(x)$ is the sigmoid function.

As in hidden units, visible variables can be reconstructed by the equation below. The visible unit can be calculated in the same way as hidden units.

$$P(l_i|h) = \frac{e^{b_i + W_i h}}{1 + e^{b_i + W_i h}} = sigm(b_i + W_i h) \tag{7}$$

The second part to be computed is the negative phase, but it is difficult as this part should be calculated for all visible and hidden units. To solve this problem, various approached were proposed as will be discussed in the next section. Briefly, all steps of training RBM method can be written at Figure 1.

Contrastive divergence

After Gibbs sampling recursive algorithm by Geman (1984),[13] Hinton in 2002 presented a practical method for estimating log-likelihood gradient of RBM called negative phase.[14] In this method, Gibbs chain is started from a real data point instead of a random state and running fewer number of steps is required comparing to Gibbs method. CD_n is related to the n-steps of recursive algorithm started from a batch of visible data selected randomly (l_1).[22] In fact, $\tilde{l}$ and $\tilde{h}$ are the reconstructed visible and hidden values from Markov chain after n-steps.[19] All steps of CD algorithm are briefly shown in Figure 2.

Persistent contrastive divergence

PCD, also known as SML, is an approximated method of Gibbs sampling approach and similar to CD technique. In this method, random numbers are used for the first chain of Gibbs Markov, instead of batch data of visible input in CD technique. But, other steps are initialized by the previous steps (Fig. 3). It is demonstrated that this method estimates the log-likelihood gradient better than CD.[16]

Fast persistent contrastive divergence

In FPCD, Tieleman and Hinton (2009) have improved PCD method by adding a new factor, called *fast,* to update network parameters. This factor leads to learn rapidly and it is important in investigating the energy space for increasing the PCD performance (see Fig. 4). Thus, in this algorithm, the two main parts for updating parameters ($\theta = \{w, b, c\}$) are θ_{regular} and θ_{fast}. In fact, FPCD will be the same as PCD if θ_{fast} is equal to zero. Two elements are used to assemble θ_{fast}, α, and φ [eq. (8)].

$$\theta_{\text{fast}} = \alpha * \theta_{\text{fast}} + \varphi, \tag{8}$$

where $\alpha = 19/20,\ \varphi = \varphi^{+} - \varphi^{-}$

Deep neural network

The concept of DNN is closely associated with ANN with many hidden layers and nodes in each layer. ANN is a mathematical model used for biological activity prediction or compound classification in drug discovery. The input data of the model, used to train DNN, are vectors of molecular descriptors,$v \in \mathbb{R}^{N \times M}$, and N and M are the number of the molecules and descriptors, respectively. Hyperbolic tangent function was used to convert a neuron's weighted input to its output activation [eq. (9)].

$$f_j(v_i) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}, \tag{9}$$

$$\text{and } z = \left(v_i * w_{ij} + b_j\right),$$

where v_i, w_{ij}, and b_j are visible input data, weight, and bias, respectively. In the output layer, linear function was used to

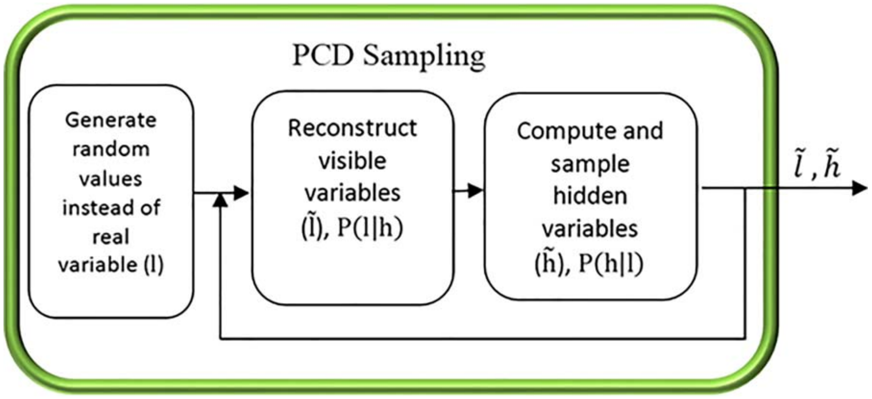
PCD Sampling
Generate random values instead of real variable (l)
Reconstruct visible variables
$(\tilde{l})$, P(l|h)
Compute and sample hidden variables
$(\tilde{h})$, P(h|l)
$\tilde{l}, \tilde{h}$

Figure 3. All steps of PCD sampling method. The interaction between positive and negative phase is related to updating parameters. The green rectangle is related to the Figure 1. [Color figure can be viewed at wileyonlinelibrary.com]

predict the biological activity. In the simple form of DNN, that is, ANN, there are three layers needed to construct the architecture of network, input, hidden, and output layers. Figure 5 shows the scheme of DNN with two hidden layer.

In DNN, network parameters should be fixed by input data, called training procedure. The popular criteria for regression purpose are mean square error (MSE) [eq. (10)].

$$\text{MSE}\ \left(Y, \tilde{Y}\right) = \frac{1}{m} \sum_{i=1}^{m} \left(Y_i - \tilde{Y}_i\right) \tag{10}$$

The common technique to learn network parameters, weights and biases, is error back-propagation algorithm in which parameters are improved based on the gradient order.[20]

Results and Discussion

Datasets

A Kaggle competition database was used for model prediction. It contained all different targets including descriptors and activity values for the training set and just the descriptors for the test set. The details of these datasets are provided in Table 1. Only training sets were downloaded for this research because information about the activity is essential for model validation. To achieve the best evaluation, training sets and test sets were selected in the following steps:

1. All molecules were arranged based on their biological activities as input data.
2. Input data were divided into ten groups with equal sizes.

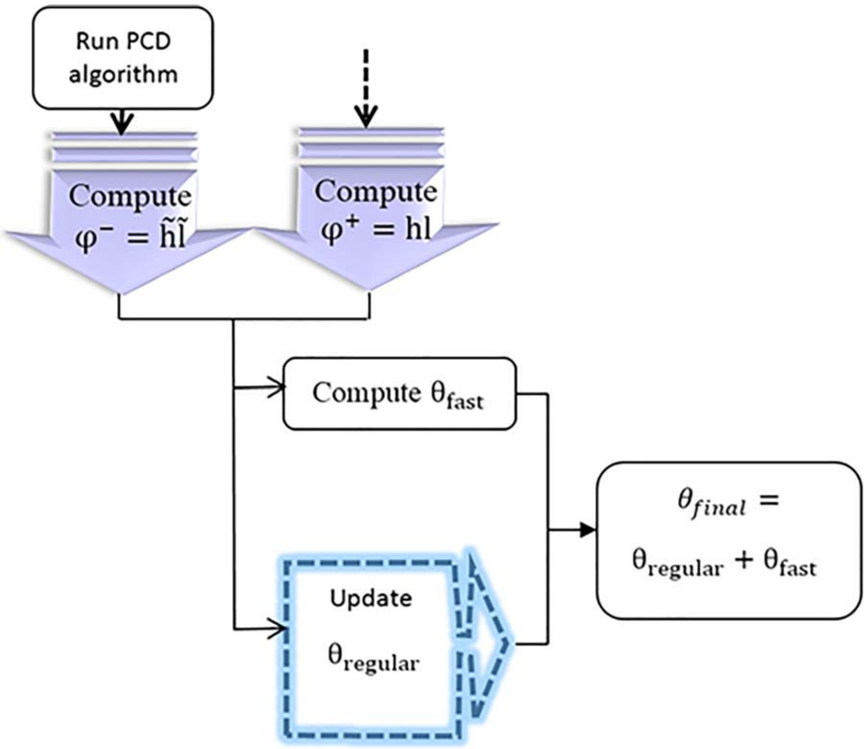

Figure 4. All steps of FPCD sampling method. The colors are referred to the Figure 1. [Color figure can be viewed at wileyonlinelibrary.com]

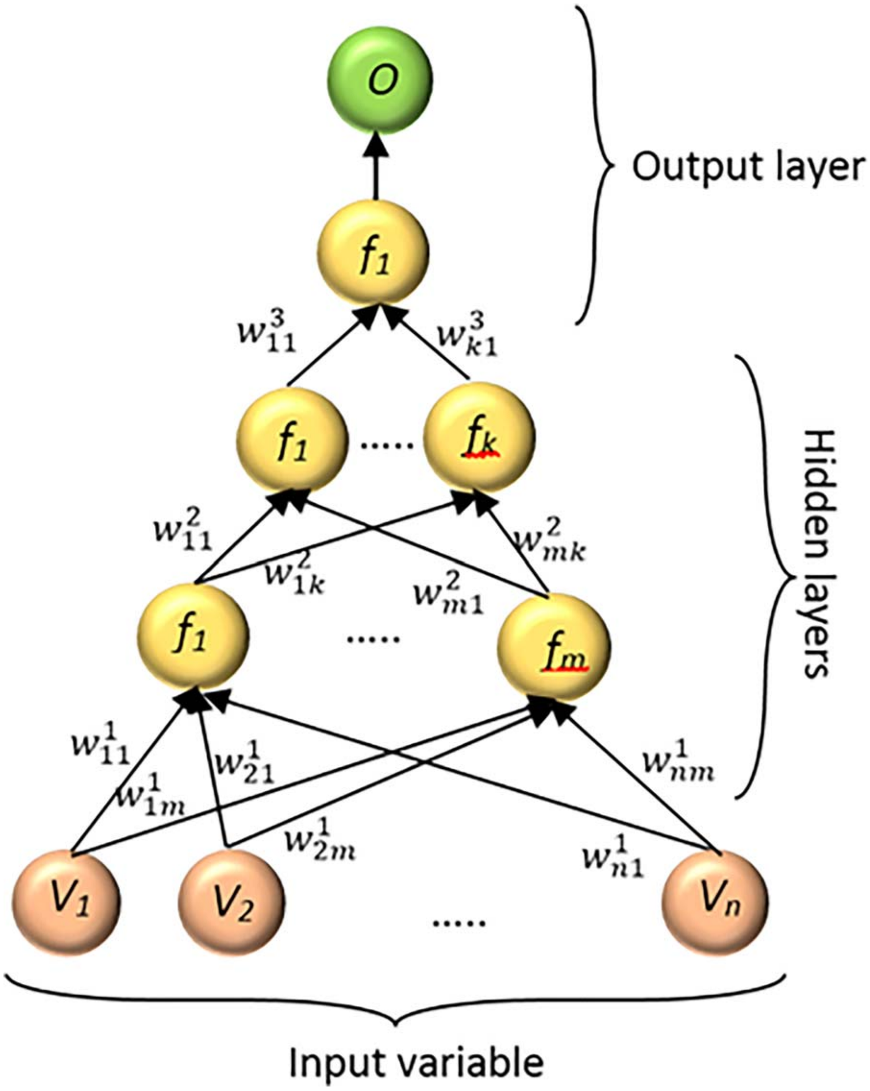

Figure 5. The structure of DNN with two hidden layers. Input and output layers are related to the ligand's descriptors and predicted biological activity, respectively and fi indicates the sigmoid function. [Color figure can be viewed at wileyonlinelibrary.com]

3. For each group, a fourfold cross validation was used for partitioning data randomly into four complementary subsamples and cross validation was repeated for four times.
4. In each iteration, one of the subsamples, almost 25% of each group, was saved as test set and the rest, 75% including descriptors and activity values, were saved as training set.

Training set was used to realize optimal initial parameters of DNN by DBN and the test set was un-touched data left out for evaluating the performance of the model. In the test set, the aim is to predict the biological activities of molecules just based on their descriptors. All algorithms were examined for all targets and thirty different input variables.

Metrics

Two standard metrics used to assess the performance of the obtained model was mean square error, MSE, for DBN output, and Pearson correlation coefficient, *R*, for DNN output. Error, regards to the observed, descriptors, and reconstructed values.

$$\text{Error}=\frac{1}{N\times M}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(d_{ij}-\hat{d}_{ij}\right)^2, \tag{11}$$

where d_{ij} and $\hat{d}_{ij}$ are related to the j^{th} descriptors from i^{th} molecule of visible and predicted value, respectively. *R* is calculated based on the predicted and observed activities in the test sets.

$$R=\frac{\sum(Y_i-\bar{Y})\left(\hat{Y}_i-\bar{\hat{Y}}\right)}{\sqrt{\sum(Y_i-\bar{Y})^2\sum\left(\hat{Y}_i-\bar{\hat{Y}}\right)^2}}, \tag{12}$$

where Y and $\hat{Y}$ are the observed and predicted molecular activity values of the i^{th} molecule, respectively.

CD, PCD, and FPCD algorithms

As mentioned in the previous sections, two main purposes of the present research were considering the effects of (1) different sampling methodologies on the results of DBN, and (2)

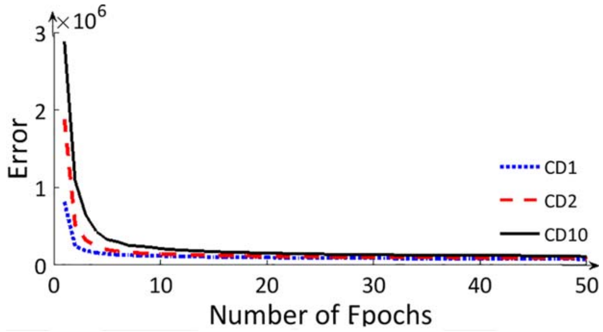

Figure 6. The comparison of the output of DBN with CD algorithm based on three different iterations. The number after CD in legend implies the number of iterations. [Color figure can be viewed at wileyonlinelibrary.com]

parameter initialization on the output of DNN. All of the parameters of DBN were selected arbitrary as follows:

1. The number of hidden layers was three.
2. The number of hidden units was 1000, 700, and 500, respectively.
3. The sigmoid function was chosen for activation function.
4. Three hundred variables were randomly selected for batch data.
5. Fifty epochs were used for each layer.
6. Learning rate was fixed at 0.01.
7. Momentums decreased from 0.5 to zero.
8. The same as other research areas such as image processing, the weights initialization was selected randomly by Gaussian distribution with zero-mean and standard deviation of about 0.01. Selection based on this distribution is beneficial when the researcher wants to create independent and uncorrelated random variables.[21]
9. Visible and hidden biases were initialized by zero.
10. The number of sampling iteration was related to the sampling method.

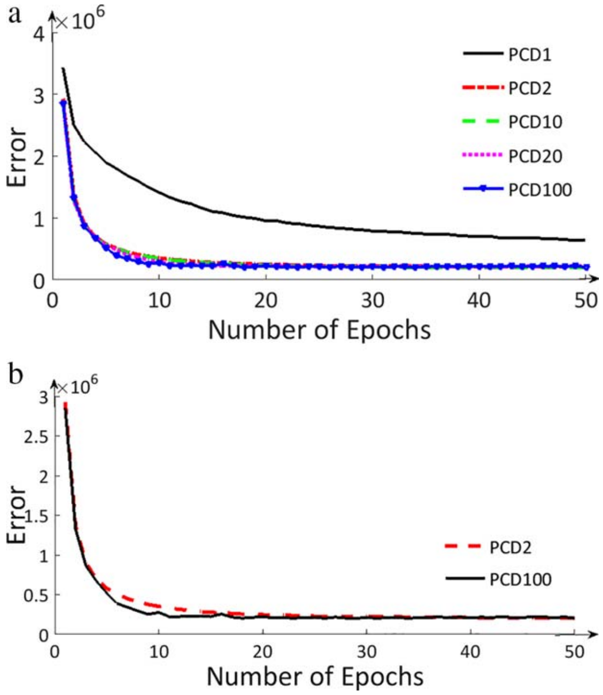

Figure 7. a) The comparison between different iterations in PCD algorithm. The number after PCD in legend implies the number of iterations. As it is shown, the worst case is related to PCD1 and the best one is for PCD 100. b) The comparison between PCD2 and PCD100. As it is shown, there is no obvious difference between iteration 2 to iteration 100 in the results of DBN, as seen in the blue circle. [Color figure can be viewed at wileyonlinelibrary.com]

Investigating the effect of three different sampling methods on DBN results

Selecting the best iteration in sampling method could be useful to reach the best results from DBN. Thus, each method was examined based on different chains. Figures 6–8 show the output of DBN based on the CD, PCD, and FPCD, respectively. It is worth mentioning that the results were averaged from all targets.

As it is shown in Figure 6, the best results were achieved with the first iteration of CD algorithm. On the other side, there is a direct relationship between increasing the time and the number of iteration.

According to Figure 7, unlike the CD technique, there is a tradeoff between time and iteration. For example, if the number of iterations is equal to 100, the stable condition is achieved in the primary epochs, green circle, but the run time is quite more than other ones. After fifty epochs, all of the items have the same results, as seen in the dotted circle.

The same as PCD algorithm, different numbers of iterations were examined for FPCD. The outputs are provided in Figure 8. Again, there is a significant difference between one iteration and others, but for more than one iteration, there is not much difference in outputs of algorithm. Time and output accuracy are two crucial parameters to select the iteration.

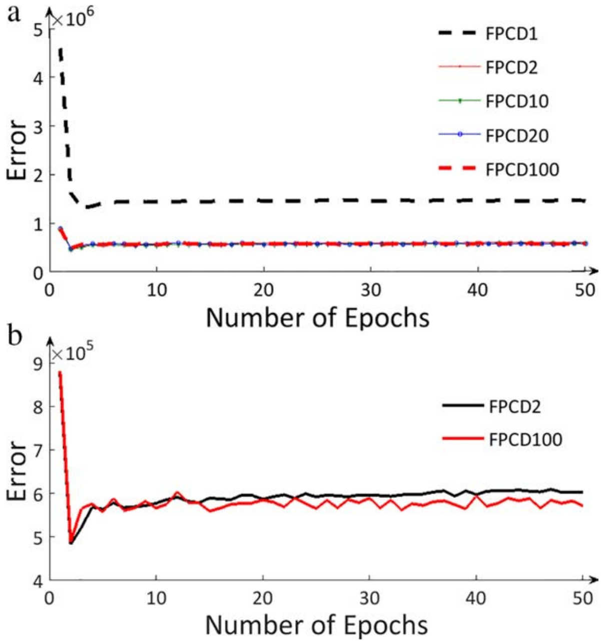

Figure 8. Investigating the effects of changing the FPCD sampling iterations on the DBN output. a) It is related to the five different number of chains. As it is shown, the differences between one iteration and others are obvious. b) It shows the comparison between 2 and 100 iterations. Time and output accuracy are two crucial parameters in selecting the iteration. [Color figure can be viewed at wileyonlinelibrary.com]

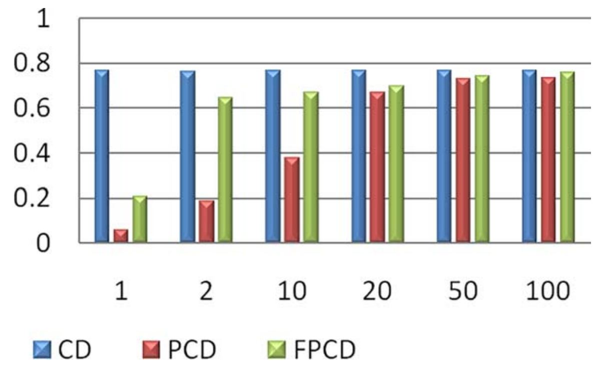

Figure 9. The mean correlation of the proposed model based on the sampling method and the different chains. [Color figure can be viewed at wileyonlinelibrary.com]

To achieve the best evaluation, the results of three sampling method based on the different chain were shown in Figure 9.

As it can be seen, for chain 100, the outputs of all methods are close together, but CD algorithm is better than the others. Conversely, changing the iteration (number) in this method does not affect the results. Thus, CD algorithm is suggested based on the time and accuracy.

The effects of initialization with DBN on the results of DNN

For the second step, the effects of DNN initialization were considered with random values and DBN, separately. For the first situation, all of the parameters, weights and biases, were selection by Gaussian distribution with zero-mean and standard deviation of about 0.01. For the second one, the outputs of three different sampling methods were tested for each target. Outputs of the proposed model, displayed in the Table 2, are based on the different sampling algorithms and DNN method with random parameters for all targets. These results are calculated from thirty different inputs.

Table 2. Averaged mean correlation of 30 times run with different inputs for all targets.

	DBN-DNN			
Dataset index	CD	PCD	FPCD	DNN
ACT1	**0.625**	0.610	0.619	0.559
ACT2	**0.857**	0.690	0.811	0.752
ACT3	**0.901**	0.844	0.896	0.869
ACT4	0.657	0.613	**0.663**	0.497
ACT5	0.853	0.840	**0.864**	0.798
ACT6	**0.774**	0.726	0.763	0.702
ACT7	0.675	0.634	**0.738**	0.596
ACT8	**0.860**	0.736	0.787	0.755
ACT9	**0.850**	0.794	0.827	0.783
ACT10	**0.868**	0.810	0.837	0.804
ACT11	**0.712**	0.689	0.709	0.595
ACT12	**0.761**	0.670	0.713	0.637
ACT13	**0.797**	0.646	0.676	0.615
ACT14	0.652	0.590	**0.654**	0.545
ACT15	0.867	0.820	**0.871**	0.802
Correlation (R)	**0.780**	0.714	0.762	0.687
Squared Correlation (R^2)	**0.609**	0.510	0.580	0.472

The bold characters are the highest values compared to the other methods.

The data show that the correct choice of initial parameters could be affected on the biological activity prediction. Beside, CD algorithm could capture the biological activity better than the two other algorithms because in CD method, the outputs of positive phase are utilized as the input for the negative phase. Conversely, FPCD algorithm offers better results than PCD because the *fast* parameter causes error to rapidly come down to zero.

Conclusion

In this study, the combination of DBN and DNN was assumed to be an efficient novel answer to the fundamental problems of local minima and over-fitting in dealing with a large number of molecules and descriptors. Also, since the best results depend on the best gradient estimation in RBM, three different sampling algorithms including CD, persistent CD, and FPCD, were utilized. As far as we know, there is no report on using DBN with different sampling methods in high throughput screening to overcome the above-mentioned problems. This study involved two main parts:

1. Investigating the effects of sampling methods on the results of DBN.
2. Investigating the effects of DBN on the results of DNN.

One of the main effective items in gradient estimation in sampling algorithms is the number of chains or iterations. Thus, to make the best decision, we changed it within the suitable range of 1 to 100. To consider the efficiency of all methods, the same input training and test groups were applied on the network. Based on the error of the model and run time, decision for the best values as the optimum numbers were made after 30 times run for each target.

In our experience, in contrast to the results presented in the image and speech processing, in the CD algorithm, the iteration sampling number has a negative effect on the results of DBN-DNN, although the results of PCD and FPCD were the same, as in other research areas.

The mean and mean squared correlations (shown in Table 2) demonstrate that an optimization in initialization will improve the ability of DNN to provide high quality predicting models. Thus, it can be said that DBN is a good alternative to the pretraining step for determining the initial parameters. Conversely, the results of the sampling algorithm indicate the remarkable point that there is no need to use very complex networks. With one chain, the CD algorithm could lead the proposed model to the desired results. It is useful for saving time and exploiting a mediocre personal computer instead of high performance computers or clusters for such a modeling job.

Associated Content

The Kaggle datasets were utilized. This datasets is available on the Internet at: www.kaggle.com.

Keywords: statistical modeling · biological activity prediction · deep neural network · initialization · deep belief network

How to cite this article: F. Ghasemi, A. Fassihi, H. Pérez-Sánchez, A. Mehridehnavi. *J. Comput. Chem.* **2017**, *38*, 195–203. DOI: 10.1002/jcc.24671

[1] J. P. Cerón-Carrasco, T. Coronado-Parra, B. Imbernón-Tudela, A. J. Banegas-Luna, F. Ghasemi, J. M. Vegara-Meseguer, I. Luque, S. Sik, S. Trædal-Henden, H. Pérez-Sánchez, *Drug Des. Open Access* **2016**, 1.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *IEEE Signal Process. Mag.* **2012**, *29*, 82.

[3] A. Lusci, G. Pollastri, P. Baldi, *J. Chem. Inf. Model.* **2013**, *53*, 1563.

[4] T. Unterthiner, A. Mayr, G. Klambauer, M. Steijaert, J. K. Wegner, H. Ceulemans, S. Hochreiter, *Adv. Neural Inf. Process. Sys.*, **2014**, *27*.

[5] J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, V. Svetnik, *J. Chem. Inf. Model.* **2015**, *55*, 263.

[6] T. B. Hughes, G. P. Miller, S. J. Swamidass, *ACS Cent. Sci.* **2015**, *1*, 168.

[7] F. Ghasemi, A. Mehri, J. Peña-García, H. den-Haan, A. Pérez-Garrido, A. Fassihi, H. Péréz-Sánchez, International Conference on Bioinformatics and Biomedical Engineering, **2015**; pp. 635–644.

[8] G. E. Hinton, S. Osindero, Y. W. Teh, *Neural Comput.* **2006**, *18*, 1527.

[9] D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, S. Bengio, *J. Mach. Learn. Res.* **2010**, *11*, 625.

[10] Y. Bengio, *Found. Trends Mach. Learn.* **2009**, *2*, 1.

[11] O. Breuleux, Y. Bengio, P. Vincent, *Neural Comput.* **2011**, *23*, 2058.

[12] G. Hinton, *Momentum* **2010**, *9*, 926.

[13] S. Geman, D. Geman, Pattern Analysis and Machine Intelligence, IEEE Transactions on 1984, PAMI-6(6), 721–741.

[14] G. E. Hinton, *Neural Comput.* **2002**, *14*, 1771.

[15] G. E. Hinton, Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470), IET, **1999**; pp. 1–6.

[16] T. Tieleman, Proceedings of the 25th international conference on Machine learning, ACM, **2008**; pp. 1064–1071.

[17] T. Tieleman, G. Hinton, Proceedings of the 26th Annual International Conference on Machine Learning, ACM, **2009**; pp. 1033–1040.

[18] Y. Wang, J. Zeng, *Bioinformatics* **2013**, *29*, i126.

[19] D. Yu, L. Deng, Deep Neural Network-Hidden Markov Model Hybrid Systems; Springer: Signals and Communication Technology, **2015**; pp 99–116.

[20] D. P. Cattin, MIAC; University of Basel, **2013**; p. 11.

[21] J. Papoulis, Probability & statistics, Prentice-Hall Englewood Cliffs, **1990**; Chapter 4, pp. 84–131.

[22] V. Mnih, H. Larochelle, G. E. Hinton, arXiv preprint arXiv*:1202.3748*, **2012**.

Received: 11 August 2016
Revised: 8 October 2016
Accepted: 19 October 2016
Published online on 14 November 2016