Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Full length article

Deep neural network in QSAR studies using deep belief network

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

[a] Department of Bioinformatic and Systems Biology, School of Advanced Technologies in Medicine, Isfahan University of Medical Sciences, Hezar-Jerib Ave., Isfahan 81746 73461, Islamic Republic of Iran
[b] Department of Medicinal Chemistry, School of Pharmacy and Pharmaceutical Sciences, Isfahan University of Medical Sciences, Hezar-Jerib Ave., Isfahan, Islamic Republic of Iran
[c] Bioinformatics and High Performance Computing Reserch Group (BIO-HPC), Computer Engineering Department, Universidad Católica de Murcia (UCAM), E30107 Murcia, Spain

ARTICLE INFO

Article history:
Received 17 October 2016
Received in revised form 8 August 2017
Accepted 26 September 2017
Available online 18 October 2017

Keywords:
Deep belief network
Biological activity prediction
QSAR
Deep neural network
Drug design

ABSTRACT

There are two major challenges in the current high throughput screening drug design: the large number of descriptors which may also have autocorrelations and, proper parameter initialization in model prediction to avoid over-fitting problem. Deep architecture structures have been recommended to predict the compounds biological activity. Performance of deep neural network is not always acceptable in QSAR studies. This study tries to find a solution to this problem focusing on primary parameter computation. Deep belief network has been getting popular as a deep neural network model generation method in other fields such as image processing. In the current study, deep belief network is exploited to initialize deep neural networks. All fifteen targets of Kaggle data sets containing more than 70 k molecules have been utilized to investigate the model performance. The results revealed that an optimization in parameter initialization will improve the ability of deep neural networks to provide high quality model predictions. The mean and variance of squared correlation for the proposed model and deep neural network are $0.618 \pm 0.407e-4$ and $0.485 \pm 4.82e-4$, respectively. The outputs of this model seem to outperform those of the models obtained from deep neural network.

1. Introduction

Machine learning is a computer programming technique applicable in statistical and mathematical research. It is evolved from the study of pattern recognition and computational learning theory of artificial intelligence. This technique constructs algorithms that they can learn to perform biological activity predictions or data classifications. These objectives are also important in all drug discovery protocols [1].

Over the previous decades, lots of machine learning algorithms have been applied in drug design. Some conventional techniques include: ANN[1] [2,3], KNN[2] [4–6], RF[3] [7], SVM[4] [8–10], MLR[5] [11] one against one [12], Bayes classifier [13] and kernel based methods such as Gaussian process [14,15]. These methods mostly suffer from the same drawbacks, *i.e.* relying on a small number of ligands and a limited selection of descriptors. Therefore, they are called shallow learning techniques. The training of these methods is simple, and they are applicable for few molecules and descriptors. Thanks to several recently developed descriptor-generating softwares, thousands of descriptors are available for a large number of compounds being reported nowadays in literature. The shallow learning techniques are inefficient to model complex relationships between the molecular descriptors. Therefore, deep architecture becomes essential when high amount of data are under process. DL[6] configuration is based on the hierarchical construction in which higher level features are founded on lower level ones. In fact, this approach comprises of multiple levels of linear and nonlinear operations. The number of these operations (depth model) refers to the longest path from an input node to an output one.

In this study, the impact of DNN architecture initialization with DBN technique was considered to predict the biological activities of Kaggle compounds. Huge numbers of element exist in each

* Corresponding author.
1 ANN: Artificial Neural Networks.
2 KNN: K-Nearest Neighbors.
3 RF: Random Forest.
4 SVM: Support Vector Machine.
5 MLR: Multiple Linear Regression
6 DL: Deep Learning.

https://doi.org/10.1016/j.asoc.2017.09.040
1568-4946/© 2017 Elsevier B.V. All rights reserved.

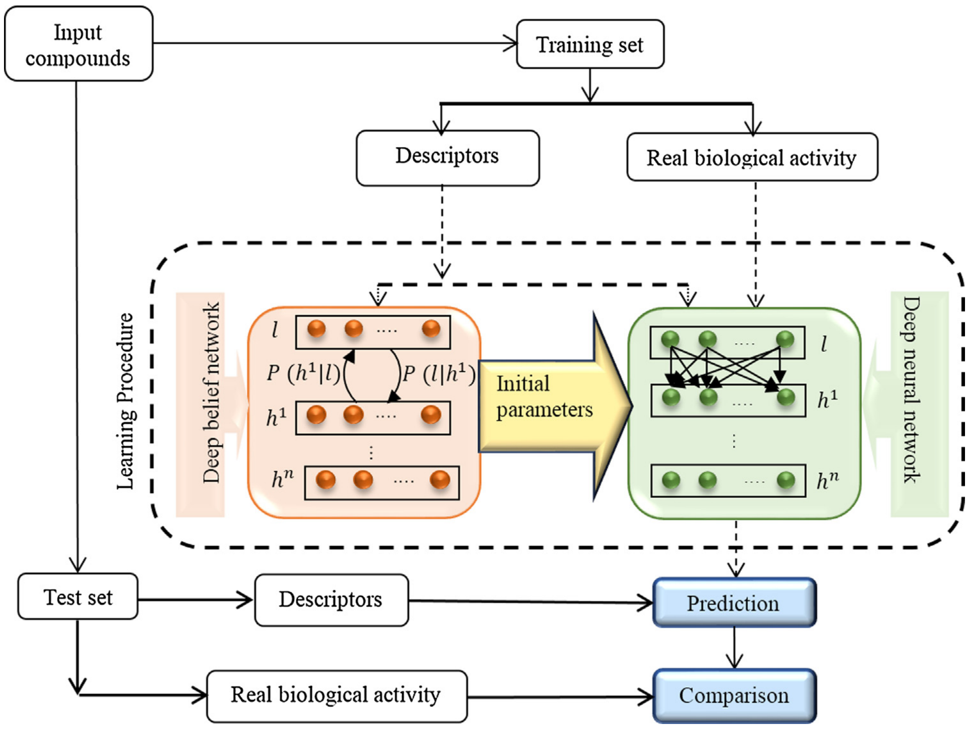

Fig 1. Main stages of proposed Model.

Kaggel dataset. In the smallest data base, there are more than 1500 molecules and 4500 descriptors for each compound. Therefore, neural network with back propagation algorithm is certainly prone to over-fitting. Based on the Hinton suggestion in 2006 [16], DBN was applied to avoid this problem. Thus, first of all, DBN was applied in order to initialize the learning procedure that fine-tunes weights of deep neural networks. These networks are called deep belief network-deep neural network (Fig. 1). In each layer of DBN, restricted Boltzmann machine was utilized. Since the gradient computation of log probability is difficult, contrastive divergence algorithm was performed on the data [26]. In this algorithm, input data should be divided into some batches. The batch size selection can change the results. Furthermore, the number of network layers and nodes directly affects the model construction time and output accuracy. It was expected that more accurate predictions would be made in terms of better correlations in shorter computation time. In addition, the proper selection of the initial parameters may decrease the probability of being stuck in local minima. This means that a more generalized model will be obtained. These advantages are added to the general advantage of DNN which overcomes the over-fitting problem in predictions. This was the reason to decide for a study on the influence of the differences in the number of nodes and layers in the network. The importance of selecting the best batch size in achieving a single set of adjustable parameters that perform well for all data sets was also regarded in the present study.

2. Related works

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 (MLP) 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]. Hinton et al. [16] introduced a fast and greedy algorithm to improve each layer using RBM.[7] This method was used to initialize a slower learning procedure that fine-tunes the weights using a contrastive version of the wake-sleep algorithm [16]. He showed that this invented algorithm could prevent over-fitting problem. Later, Benjio et al. in 2009 proposed deep architecture, in which single-layer models such as RBM were exploited as unsupervised learning building blocks to construct deeper models such as DBN[8] [17]. After that, Hinton (2012) introduced a practical guide useful to construct RBM algorithm step by step [18]. In 2014, new algorithm was introduced to prevent over-fitting problem by Srivastava named drop-out [19]. Nowadays, DL has been successfully applied in different processing fields such as computer vision, speech processing, image processing and chemo-informatics [20].

In chemo-informatics research, deep learning is applied to capture complex statistical patterns between thousands of descriptors extracted from numerous compounds. Various approaches have been used based on deep architecture. Alessandro Lusci et al. (2013) showed how recursive neural network approaches can be applied to the problem of predicting molecular properties [21]. Restricted Boltzmann machine was used for predicting drug-target interactions by Yuhao Wang and Jianyang Zeng in 2013 [22]. Thomas Unterthiner et al. [23] compared the performance of deep learning approach in seven target prediction methods on ChEMBL database. They found out that deep learning outperformed all other methods with respect to the AUC[9] [23]. Junshui Ma et al.proved that DNN[10] based on the procedure of drop-out can routinely make better prospective predictions than RF on a set of large diverse QSAR[11] data sets [24]. Hughes [25] utilized a database of 702 epoxidation reactions to build a deep machine learning network. Finally, it was

[7] RBM: Restricted Boltzmann Machine.
[8] DBN: Deep Belief Networks.
[9] AUC: Area Under the Curve.
[10] DNN: Deep Neural Networks.
[11] QSAR: Quantitative Structure Activity Relationship.

concluded that the SOEs[12] had 94.9% AUC, and separation of epoxidized and non-epoxidized molecules was done with 79.3% AUC [25]. The impact of different sampling approaches on deep network were considered by Ghasemi et al. in 2016 [26].

3. Material and methods

3.1. Data sets

The personal computer utilized for all models was an Intel-based core i7 (CPU at 3.20 GHz) and Geforce GRX760 graphic card. The proposed model was executed in Matlab (2016). All fifteen targets of the Kaggle database competition held by Merck sponsor (2012) were chosen as an input network to consider the proposed network efficacy (www.kaggle.com) (Table 1).

3.2. Deep belief network

Deep belief network is one of the classes of deep generative models composed of *l* stacks of restricted Boltzmann machine. The main aim of DBN is the weight initialization of a deep neural network to produce optimum model in comparison to the model by random weights. This approach makes the predictions extremely effective. Alternatively, DBN can be effectively used to perform layer by layer pre-training intended to initialize training of a back propagation algorithm.

The energy-based probabilistic model is a common method used to make a joint distribution between observed data, *x*. and hidden variables,*h*, as follows:

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

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

In each layer, calculated output was used as an input for the next layer [26].

3.3. Restricted Boltzmann machine

Restricted Boltzmann machine is a kind of Boltzmann machine with no internal layer connection within both visible and hidden layers. In this model, the probability of joint configuration *(l,h)* is defined as follows:

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

where $Z = \sum_{i,j} \exp\left(-Energy\left(l_i, h_j\right)\right)$ is called normalization factor.

The probability of visible unit is achieved by summation of all hidden units.

$$P(v) = \frac{1}{Z}\sum_h exp(-Energy(l, h)) \quad (3)$$

The derivative of the logarithm of probability equation mentioned above is defined as:

$$\frac{\partial \log(P(v))}{\partial\theta} = \frac{\partial\sum_h exp(-Energy(l, h))}{\partial\theta} - \frac{\partial \log(Z)}{\partial\theta} = \varphi^+ - \varphi^-, \quad (4)$$

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

Estimating the positive phase is simple because of the lack of internal connection between visible or hidden units. The conditional probability for any pair of hidden units is achieved by:

$$P\left(h_j = 1|l\right) = \frac{e^{c_j+W_jl}}{1+e^{c_j+W_jl}} = sigm\left(c_j + \sum W_jl\right), \quad (5)$$

where W_j is j^{th} row of *W* and *sigm (x)* is the sigmoid function. Visible variables can be recreated in the same way as hidden units.

The second part, negative phase, should be calculated for all visible and hidden units. One of the proposed algorithms to approximate log-likelihood gradient is contrastive divergence (CD). CD algorithm has been applied to update training parameters, biases and weights in RBM. The advantage of this procedure is observable when parallel processing is applied by Matlab [17,18].

Assuming hidden units as binary, all visible variables are separated into some categories based on the batch size defined at the first steps. Then, the hidden units are calculated with Eq. (6).

$$P\left(h_j\right) = sigm\left(c_j + \sum_i w_{i,j}l_i\right) \quad (6)$$

Finally, the hidden unit would turn on if the probability is greater than the threshold. For updating visible units, it is common to use the probability, p_i, that is computed with the following formula:

$$P(v_i) = b_i + \sum_j w_{i,j}h_j \quad (7)$$

After estimating the gradient, it is possible to update parameters, biases and weights. Two main parameters, learning rate and momentum, can improve the updated parameters in terms of the previous ones. Learning rate is multiplied by ΔW. If this parameter is too large, the reconstruction error would increase, and if it is too low, the processing time will be large. The best point for learning rate is associated with weight averaging through several updates. Momentum is useful for increasing learning speed. It is used after computing batch data and updating parameters, thus multiplied by W_{old} [18]. These steps are shown in Fig. 1.

All steps of training RBM method can be briefly put as:

1- Determining the required parameters:

a) ε :Learning rate for the stochastic gradient descent
b) b) <*b*>** 1 ** > μ :Momentum for updating parameters
c) Visible and hidden biases initialized by zero number
d) Initial weights set by random variable with Gaussian distribution by zero mean and 0.1 variance
e) The number of hidden units.
f) The number of layers.
g) Batch size for CD sampling

2- Computing $\varphi^+ = h\hat{l}$
3- Computing $\tilde{h} = P\left(h_j = 1\right)$ and $\tilde{l} = P\left(l_j = 1\right)$ for all *i,j* using CD sampling
4- Computing $\varphi^- = \tilde{h}.\tilde{l}$
5- $w_{ij}{}^{new} = \mu.w_{ij}{}^{old} + \Delta w_{i,j}$, $\Delta w_{i,j} = \varepsilon(\varphi^+ - \varphi^-)$
6- $b_{ij}{}^{new} = \mu . b_{ij}{}^{old} + \Delta b_{i,j}$, $\Delta b_{i,j} = \varepsilon\left(l - \tilde{l}\right)$
7- $c_{ij}{}^{new} = \mu . c_{ij}{}^{old} + \Delta c_{i,j}$, $\Delta c_{i,j} = \varepsilon\left(h - \tilde{h}\right)$

3.4. Deep neural network

The concept of DNN is closely associated with artificial neural network with many hidden layers and nodes in each layer utilized to predict biological activity of compounds. The input data are molecular descriptors,$v \in \mathbb{R}^{N\times M}$, *N* and *M* are the number of the

[12] SOEs: Site Of Epoxidations.

Table 1
Fifteen different Kaggle data sets utilized in this study [14].

Data set index	Data set	Description	Number of molecules	Number of descriptors
ACT1	3A4	CYP P450 3A4 inhibition $-\log(IC_{50})$ M	50,000	9491
ACT2	CB1	binding to cannabinoid receptor 1 $-\log(IC_{50})$ M	11640	5877
ACT3	DPP4	inhibition of dipeptidyl peptidase 4 $-\log(IC_{50})$ M	8327	5203
ACT4	HIVINT	target inhibition of HIV integrase in a cell based assay $-\log(IC_{50})$ M	2421	4306
ACT5	HIVPROT	target inhibition of HIV protease $-\log(IC_{50})$ M	4311	6274
ACT6	LOGD	logD measured by HPLC method	50,000	8921
ACT7	METAB	percent remaining after 30 min microsomal incubation	2092	4505
ACT8	NK1	target inhibition of neurokinin1 (substance P) receptor binding $-\log(IC_{50})$[a] M[b]	13482	5803
ACT9	OX1	target inhibition of orexin 1 receptor $-\log(Ki)$ M	7135	4730
ACT10	OX2	target inhibition of orexin 2 receptor $-\log(Ki)$ M	14875	5790
ACT11	PGP	ADME transport by *p*-glycoprotein log(BA/AB)	8603	5135
ACT12	PPB	ADME human plasma protein binding log(bound/unbound)	11622	5470
ACT13	RAT_F	ADME log(rat bioavailability) at 2 mg/kg	7821	5698
ACT14	TDI	ADME time dependent 3A4 inhibitions $\log(IC_{50}$ without NADPH/IC_{50} withNADPH)	5559	5945
ACT15	THROMBIN	target human thrombin inhibition $-\log(IC_{50})$ M	6924	5552

[a] $\log(IC_{50})$: −log of the inhibitory concentration inhibiting 50 percent of the enzymes/receptors.

molecules and descriptors, respectively. Hyperbolic tangent function was used to convert a neuron's weighted input to its output activation (Eq. (8)).

$$f_j(v_i) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad \text{and} \quad z = \left(v_i * w_{ij} + b_j\right), \tag{8}$$

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 predict the biological activity.

3.5. Random forest

Random forest has been one of the popular learning algorithms utilized for classification or regression in QSAR studies [27]. Boosting and bagging are two main methods utilized to classify trees. In the boosting step, extra weights are exclusive to points predicted incorrectly by earlier predictors while in the bagging step, prospers tree does not depend on earlier trees [28]. Number of trees is one of the major parameters in RF. In this study, biological activity prediction with RF algorithm was done by *R* package.

3.6. Multiple linear regression

Multiple linear regressions is one of the statistical modeling approaches. MLR is applied to estimate a linear model between input observations that are independent, *i.e.* descriptors in QSAR, and response, *i.e.* biological activities, that must be predicted [29].

4. Results and discussion

4.1. Data sets

A Kaggle competition database was used in model prediction. Table 1 presents these datasets in full details. For this study, only training sets were downloaded for the lack of information about the molecular activity of the test sets. To achieve the best evaluation, all molecules were arranged based on their biological activities as input data. After that, input data were divided into ten groups with the equal sizes. For each group, 4-fold cross validation was used to splitting data randomly into four complementary subsamples and cross validation was repeated for four times. In each period, one of the subsamples was saved as test set compound and the others were used for training set. On the other side, seventy five percent of each group was extracted randomly as training set including descriptors and activity values. They were used as to realize optimal initial parameters of DNN by DBN. The rest of the compounds were stored as the test set. They were un-touched data and 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. Finally, four different results were averaged to achieve a single prediction. This procedure was repeated for thirty times. All of the thirty results were averaged to predict the molecular activities.

4.2. Metrics

In order to assess the performance of the obtained model, four different and most common standard metrics were used with regard to the predicted and experimental activities in test sets, correlation coefficient (R), root mean square error ($RMSE$) and goodness of fit. These criteria are determined as follows:

$$R = \frac{\sum\left(Y_i - \bar{Y}\right)\left(\hat{Y}_i - \hat{Y}\right)}{\sqrt{\sum\left(Y_i - \bar{Y}\right)^2 \sum\left(\hat{Y}_i - \hat{Y}\right)^2}} \tag{9}$$

$$RMSE = \sqrt{\frac{\sum\left(Y_i - \hat{Y}_i\right)^2}{n}} \tag{10}$$

$$Goodness\ \ of\ \ fit = 1 - \frac{\sum\left(Y_i - \hat{Y}_i\right)^2}{\sum\left(Y_i - \bar{Y}\right)^2}, \tag{11}$$

where $Y$and $\hat{Y}$ are experimental and predicted activities, respectively.

The final criteria utilized to investigate the performance of the model is analysis of variance (ANOVA). ANOVA consists of calculations that provide information about levels of variability within a regression model and form a basis for tests of significance. The basic regression line concept, DATA = FIT + RESIDUAL, is rewritten as follows:

$$(y_i - \bar{y}) = \left(\hat{Y}_i - \bar{y}\right) + \left(y_i - \hat{Y}_i\right)$$

F can be defined as follows:

$$\mathrm{ANOVA: F - test: F} = \frac{\sum\left(\hat{Y}_i - \bar{Y}\right)^2}{\sum\left(Y_i - \hat{Y}\right)^2} = \frac{SSM}{SSE} \tag{12}$$

SSM and *SSE* is sum of squares of model and sum of squares of error, respectively. SSE reduction caused to the increment of the F. Thus, if model performance is improved, F goes high.

In this study, the main research objective was to investigate the impact of network initialization with optimal parameters. Accordingly, two models were constructed by different initialization techniques. In the first one, all of the parameters were initialized by the random values and in the second model, DBN network was utilized to define all of the initial parameters.

4.3. *Parameters selection for system training*

The major problems were considered in this study are the robustness of the proposed network with respect to different parameters and the effect of weights and biases initialization on the output of the network. As mentioned in previous section, the selection of initial parameters for model prediction is a critical issue.

Thus, the outputs of the model based on various parameters are analyzed in this study.

All of the parameters of network were selected arbitrarily as follows:

1. The number of hidden layers was defined
2. The number of hidden units was defined
3. The sigmoid function was chosen for activation function.
4. The size of batch data was defined
5. epochs were used for each layer
6. Learning rate was defined
7. Momentums were decreased from 0.5
8. The initialization of the weights was chosen randomly by Gaussian distribution with zero-mean and standard deviation of about 0.01.
9. Visible and hidden biases were initialized by zero.

In order to analysis of the proposed network robustness with different parameters, three different parameters were to be examined: batch size, number of hidden layers and number of neurons in each hidden layer.

4.3.1. *Batch size*

As mentioned in previous section, the main problem in restricted Boltzmann machine method is the difficulty in calculating gradient of logarithm probability of the normalization factor. This calculation can be made by contrastive divergence algorithm which requires the updated parameters for which the training set should be divided into small mini-batches. In image processing, 10–100 samples are usually chosen for batch size. There is no report suggesting these parameters for chemo-informatics databases. After several investigations of the data sets, it was concluded that 0.01–0.05 percent of the data sets seemed to be suitable for batch size. Then, according to the size of all data sets, it was decided to choose ten different batch sizes: lower than 0.01%, from 0.01% to 0.05% and higher than 0.05%, and the algorithm was run for all Kaggle datasets. The network was adjusted with three layers and 50 neurons in each. To evaluate the impact of these differences, ten different runs based on the data selection method mentioned on the previous section; in sum 1500 runs, were performed.

Afterwards, the mean correlation in all data sets was calculated. As shown in Fig. 2, when the batch size is very small (less than 50) or large (more than 1500), the outputs are not satisfactory. On the other hand, batch size is related to the number of samples in each iteration, thus with decreasing this parameter, the run time increases. As it can be seen, the best result is achieved when the batch size is between 0.01- 0.05 percent of the training data.

4.3.2. *The number of hidden layers*

To investigate the behavior of the network versus changing layers, ten runs with ten different input for all data sets was done, totally 1500 runs. To avoid the complexity network, it is supposed that all layers have the same neurons. Mean squared correlations of all data sets for five different layers are plotted over all targets in Fig. 3. As illustrated, the output of network with two layers is only slightly fewer than network with three hidden layers. Therefore, the two hidden layers was optimum choice for constructing network.

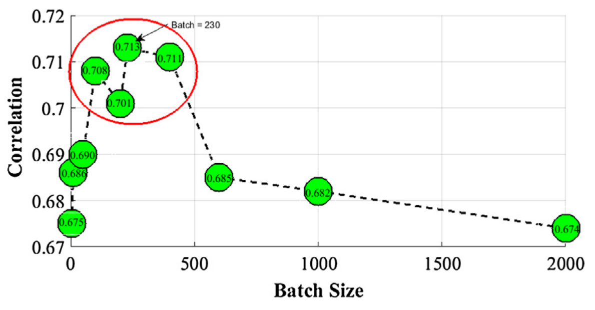

Fig. 2. Mean correlation of all data sets with ten different batch sizes. Each marker points to network with the same layers and nodes. The circle indicates the best region for choosing batch size and the selected point is related to the best obtained mean correlation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

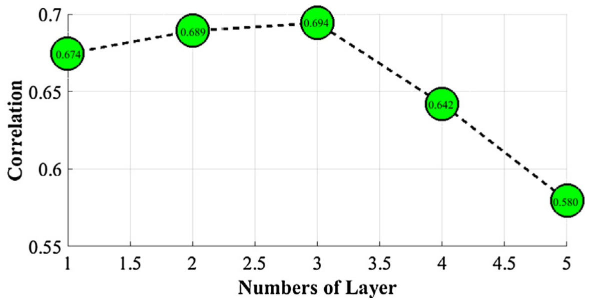

Fig. 3. Mean correlation of all data sets for five different layers. Each marker points to network with the same nodes and batch size.

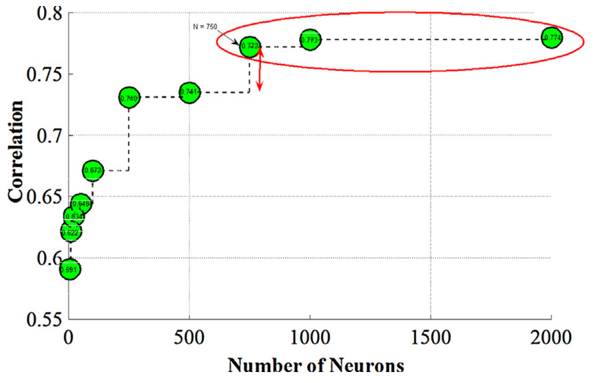

Fig. 4. Mean correlation of all data sets with ten different hidden units. Each marker points to network with the same layers and batch size. The circle indicates the best region for choosing neurons, and the selected point is related to the optimum node. The red double arrow points to the final increase in the value of the correlation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.3.3. *The number of neurons of the hidden layers.*

In order to consider the effect of changes in the number of nodes, the network with two layers and 250 batch data in each layer was performed. Then, as for the previous parameters, a network with ten runs using ten different inputs for each data set was run. The total number of the runs was 1500. The relationship between mean correlations of all data sets with ten different hidden units is shown in Fig. 4. By increasing the nodes, the optimum point for the number of hidden layer neurons will be obtained. But if the number of the nodes exceeded the optimum point, the output will not change

Table 2
Squared correlation of all targets with their standard deviations for MLR, RF, ANN, DNN and DBN-DNN.

Method	MLR		RF		ANN		DNN		DBN-DNN	
Index	$R^2 \pm std$	*F-test*	$R^2 \pm std$	*F-test*	$R^2 \pm std$	*F-test*	$R^2 \pm std$	*F-test*	$R^2 \pm std$	*F-test*
ACT1	0.026 ± 0.016	0.392	0.365 ± 4.1e − 4	0.861	0.184 ± 0.034	0.685	0.312 ± 5.3e − 4	0.785	0.391 ± 4.4e − 4	0.891
ACT2	0.186 ± 0.037	0.609	0.654 ± 2.2e − 4	1.68	0.408 ± 0.022	0.780	0.566 ± 2e − 4	1.401	0.734 ± 2.5e − 5	1.97
ACT3	0.008 ± 0.028	0.196	0.665 ± 3.6e − 4	1.63	0.336 ± 0.037	0.846	0.755 ± 8.1e − 5	1.521	0.812 ± 5.5e − 5	1.74
ACT4	0.006 ± 0.043	0.318	0.414 ± 3.2e − 3	1.302	0.260 ± 0.060	0.607	0.247 ± 1.1e − 3	1.095	0.432 ± 1.3e − 3	1.335
ACT5	0.044 ± 0.026	0.548	0.704 ± 3.0e − 4	1.849	0.376 ± 0.043	1.6388	0.637 ± 2e − 4	1.797	0.728 ± 1.2e − 4	1.854
ACT6	0.015 ± 0.021	0.21	0.50 ± 2.3e − 3	0.780	0.302 ± 0.042	0.560	0.493 ± 1.7e − 3	0.769	0.599 ± 1.6e − 3	0.892
ACT7	0.016 ± 0.089	0.169	0.40 ± 1.2e − 2	1.437	0.281 ± 0.035	0.902	0.355 ± 2.1e − 3	1.038	0.456 ± 1.6e − 3	1.567
ACT8	0.038 ± 0.036	0.619	0.68 ± 0.4e − 4	2.36	0.393 ± 0.041	1.072	0.570 ± 1e − 4	1.686	0.740 ± 0.8e − 4	2.879
ACT9	0.036 ± 0.023	0.104	0.70 ± 0.5e − 3	2.051	0.489 ± 0.045	0.717	0.613 ± 1e − 4	1.655	0.723 ± 1.2e − 4	2.213
ACT10	0.055 ± 0.024	0.531	0.706 ± 1.1e − 4	1.931	0.429 ± 0.035	1.284	0.646 ± 3.6e − 5	1.843	0.753 ± 1.6e − 5	2. 066
ACT11	0.022 ± 0.042	0.068	0.508 ± 3.3e − 4	1.0950	0.271 ± 0.040	0.636	0.354 ± 3.6e − 4	1.140	0.507 ± 2.9e − 4	1.0994
ACT12	0.065 ± 0.029	0.424	0.496 ± 4.3e − 4	1.021	0.291 ± 0.043	0.307	0.408 ± 2.2e − 4	0.667	0.579 ± 2e − 4	1.120
ACT13	0.005 ± 0.041	0.268	0.431 ± 1.6e − 3	1.392	0.191 ± 0.037	0.725	0.378 ± 2e − 4	1.116	0.635 ± 1.4e − 4	1.861
ACT14	0.003 ± 0.037	0.700	0.385 ± 3.6e − 3	0.935	0.109 ± 0.040	0.898	0.297 ± 2e − 4	0.966	0.425 ± 2.2e − 5	1.032
ACT15	0.013 ± 0.023	0.038	0.754 ± 1.0e − 4	3.021	0.507 ± 0.024	1.696	0.643 ± 1e − 4	2.422	0.752 ± 1e − 4	3.146
Mean	0.039 ±0.0343	0.346	0.558 ± 1.7e − 3	1.56	0.322 ±0.038	0.890	0.485 ±4.8e − 4	1.33	**0.618** # ***XPS*** # 9617 ;±**4.1e-4**	**1.69**

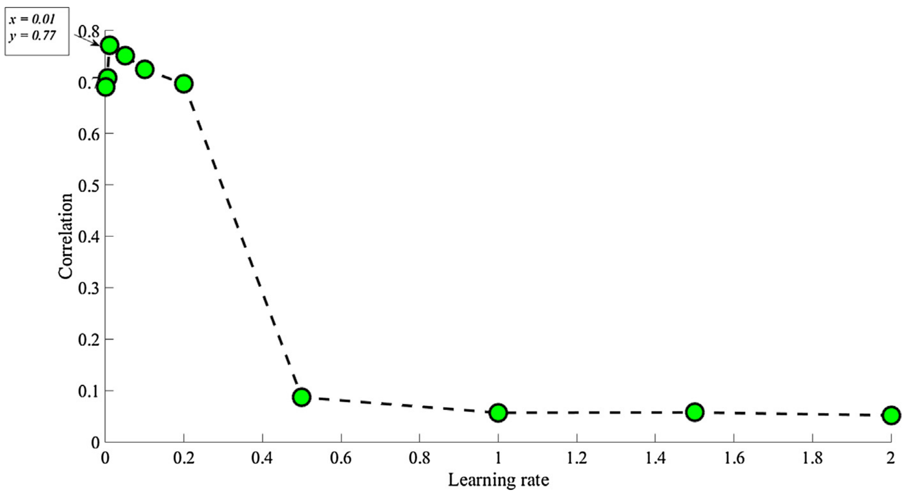

Fig. 5. Mean correlation of the network with ten different learning rate. The arrow indicates the best point for choosing learning rate.

significantly, whereas, by increasing the hidden units, the run time increases. Thus, it is better to choose the optimum point as the number of nodes.

4.3.4. Choosing value of the learning rate

In order to consider the effect of changes of the value of the learning rate on the network output, the network with two layers and 250 batch data in each layer was performed. The relationship between mean correlations of all data sets with ten different learning rate is shown in Fig. 4. By increasing the learning rate, the optimum point will be obtained. But if the value exceeded the optimum point, the output will decreased.

4.3.5. Comparing the proposed model by some of the other regression methods

To evaluate the effect of initialization on learning network parameters, the results of DNN and some of the shallow learning algorithms, MLR, RF and ANN are reported. In DNN and ANN approaches, random parameters were used for network initialization. Since deep belief network had very good consequences in signal or image processing for both generative and discriminative models, we decided to apply it for pre-training and initializing network parameters.

First, in order to gain the effects of DBN used for initializing DNN, square of correlations (R^2) were calculated with their standard deviations for all test sets. Thirty different runs were performed with different input data for all Kaggel data sets (totally 450 runs). The results of these runs are summarized in Table 2. At the end of the table the mean squared correlation is presented. The network with one hidden later and ten nodes in the hidden layer was applied for ANN and RFs were constructed with hundred trees. DNNs and DBNs-DNNs were created based on the recommended parameters in the previous section *i.e.* two hidden layers, 750 hidden units in each hidden layer and batch size equal to 0.03% of training data for each datasets. It approves the theory in drug discovery that the proper selection of network parameters could prevent serious problems such as prone to over fitting. MLR is a linear model. Hence, as it was expected, the outputs are not satisfactory. The outputs of ANNs are some deal better than MLR, but their variances are hundred times more than three other approaches.

Computation of correlation is really useful to evaluate model performance. But, as the only criteria it can be misleading. The goodness of fit is another useful criteria used for model validation. goodness of fit is normally lower than R but is a true guide to the predictive potential of the equation [30]. Fig. 5 compares the correlation (R) and goodness of fit between the two different methods, DNN and DBN-DNN, for each data set. As the previous table, all targets were shown based on the index of the target. For all data base, the proposed model has R and goodness of fit values more than the

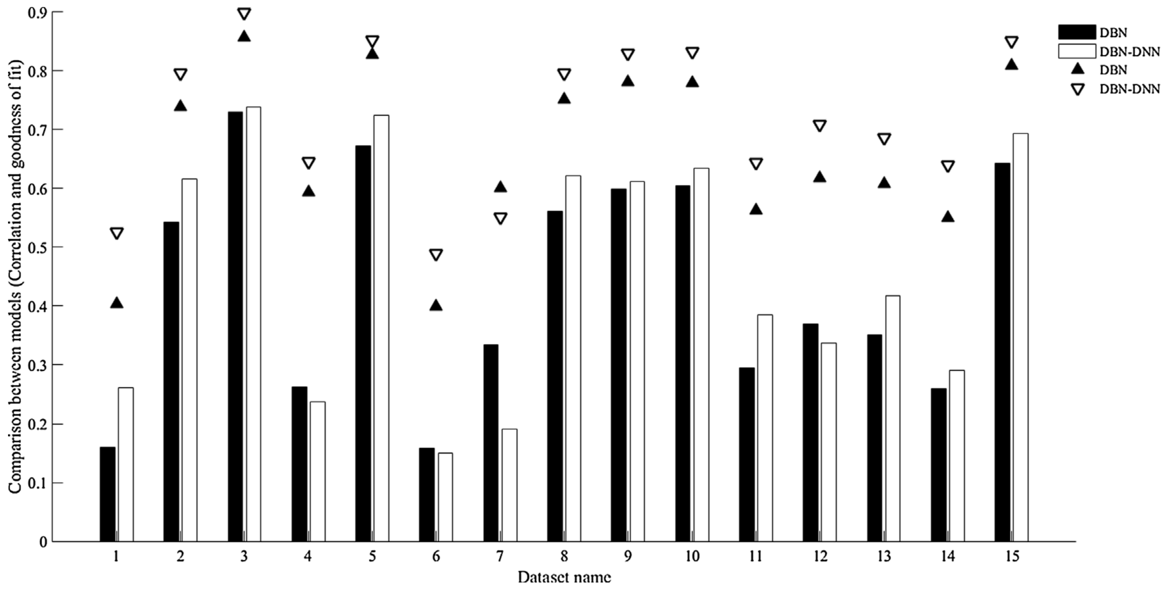

Fig. 6. The comparison of the correlation and goodness of fit calculated for the proposed model with DNN for each target. The upward and downward pointing triangle markers indicate *R*, and the bar graph shows the goodness of fit for DNN and DBN-DNN, respectively.

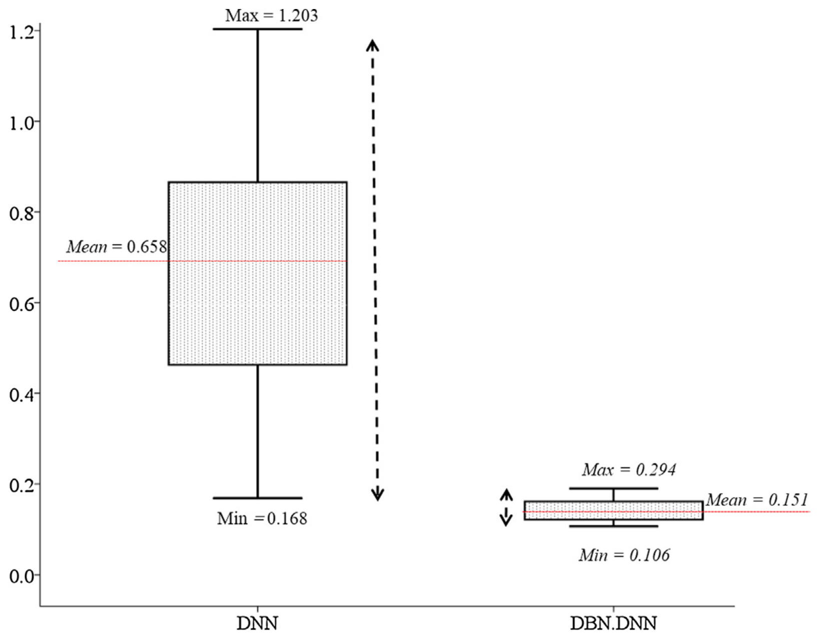

Fig. 7. The comparison of *RMSE* of the proposed model with deep neural network for all targets. The horizontal lines are related to the mean *RMSE* for DNN and DBN-DNN, and the vertical black double arrow indicates the range of two methods. It shows how network initialization affects the prediction of the model. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

one obtained in DNN, though this generally indicates the fact that the proposed model works efficiently.

To comparison outputs of two different models, the differences of *RMSE* between them for all of the targets were calculated. Fig. 6 shows that the *RMSE* averaged over all DBN-DNN and all 15 data sets are 0.658 greater in value than that of DNN. The red lines indicate the average of *RMSE* obtained by DNN and DBN-DNN methods, respectively. It shows that the average of *RMSE* for a network with fine-tuned weights and biases is less than the one for other approach. On the other hand, when the initial parameters are chosen appropriately, the model will be more efficient than the model obtained by random initialization (Fig. 7).

5. Conclusion

The main purpose of this research was to examine the proper parameter initialization in deep neural networks using deep belief networks. Experimental results show that DBN approach can be used to determine the initial DNN parameters, biases and weights, which in most cases outperform DNN method. One important feature of DBN is the ability to fine-tune the network parameters with an unsupervised learning algorithm. RBM is the best suited method used in each layer of network. In fact, this combination was assumed to be an efficient novel answer to the problems pointed out earlier. This solution is an important advantage to avoid output variation from one input to the other. On the other hand, it helps us to control the problems such as, over-fitting and being stuck in local minimum.

In order to come across with the best decision in choosing some key parameters, we changed batch sizes and the number of hidden units and layers of the networks in the suitable range. Finally, a set of values were recommended for the proposed network, appropriate for high throughput screening. It is worth mentioning that the recommended parameters produced, significantly outperformed for most datasets while the modeling run times were less than one minute for all datasets and a personal computer applied for the modeling jobs.

To investigate the efficiency of the proposed model, the results were calculated through three different metrics and finally compared with some of the shallow learning algorithm, MLR, RF, ANN and DNN with random initial parameters. The results indicated that DBN-DNN model performed well, and the mean correlation of all data sets were greater than mentioned algorithms while the range of the changes was smaller than others.

The results revealed that an optimization in initialization will improve DNN potential to provide high quality predicting models. In fact, time saving and possibility of replacing a high performance computer or cluster by a mediocre personal computer proved that there would be no need to use a very complex network after changing the weights and biases, whereas the output of such model demonstrates significant superiority over the output of the deep neural network.

Associated content

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

Funding

This work was supported from the vice chancellery of research in Isfahan University of Medical Sciences.

References

[1] J.P. Ceron-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, Application of Computational Drug Discovery Techniques for Designing New Drugs Against Zika Virus, Open Access, Drug Designing, 2016, pp. 1–2.
[2] M. Shahlaei, A. Fassihi, L. Saghaie, Application of PC-ANN and PC-LS-SVM in QSAR of CCR1 antagonist compounds: a comparative study, Eur. J. Med. Chem. 45 (2010) 1572–1582.
[3] H. Pérez-Sánchez, G. Cano, J. García-Rodríguez, Improving drug discovery using hybrid softcomputing methods, Appl. Soft Comput. 20 (2014) 119–126.
[4] S. Ajmani, K. Jadhav, S.A. Kulkarni, Three-dimensional QSAR using the k-nearest neighbor method and its interpretation, J. Chem. Inf. Model. 46 (2006) 24–31.
[5] P. Itskowitz, A. Tropsha, k nearest neighbors QSAR modeling as a variational problem: theory and applications, J. Chem. Inf. Model. 45 (2005) 777–785.
[6] F. Nigsch, A. Bender, B. van Buuren, J. Tissen, E. Nigsch, J.B. Mitchell, Melting point prediction employing k-nearest neighbor algorithms and genetic parameter optimization, J. Chem. Inf. Model. 46 (2006) 2412–2422.
[7] V.E. Kuz'min, P.G. Polishchuk, A.G. Artemenko, S.A. Andronati, Interpretation of QSAR models based on random forest methods, Mol. Inf. 30 (2011) 593–603.
[8] M. Shahlaei, A. Fassihi, QSAR analysis of some 1-(3, 3-diphenylpropyl)-piperidinyl amides and ureas as CCR5 inhibitors using genetic algorithm-least square support vector machine, Med. Chem. Res. 22 (2013) 4384–4400.
[9] M. Shahlaei, R. Sabet, M.B. Ziari, B. Moeinifard, A. Fassihi, R. Karbakhsh, QSAR study of anthranilic acid sulfonamides as inhibitors of methionine aminopeptidase-2 using LS-SVM and GRNN based on principal components, Eur. J. Med. Chem. 45 (2010) 4499–4508.
[10] G. Cano, J. García-Rodríguez, H. Pérez-Sánchez, Improvement of virtual screening predictions using computational intelligence methods, Lett. Drug Des. Discov. 11 (2014) 33–39.
[11] M. Shahlaei, A. Madadkar-Sobhani, A. Fassihi, L. Saghaie, D. Shamshirian, H. Sakhi, Comparative quantitative structure–activity relationship study of some 1-aminocyclopentyl-3-carboxyamides as CCR2 inhibitors using stepwise MLR, FA-MLR, and GA-PLS, Med. Chem. Res. 21 (2012) 100–115.
[12] F, A. Ghasemi, Mehri, J. Peña-García, H. den-Haan, A. Pérez-Garrido, H. Fassihi, Improving activity prediction of adenosine A2B receptor antagonists by nonlinear models, in: International Conference on Bioinformatics and Biomedical Engineering, Springer, 2015, pp. 635–644.
[13] A. Koutsoukas, R. Lowe, Y. KalantarMotamedi, H.Y. Mussa, W. Klaffke, J.B. Mitchell, R.C. Glen, A. Bender, In silico target predictions: defining a benchmarking data set and comparison of performance of the multiclass naïve bayes and parzen-rosenblatt window, J. Chem. Inf. Model. 53 (2013) 1957–1966.
[14] F.R. Burden, Quantitative structure-activity relationship studies using gaussian processes, J. Chem. Inf. Comput. Sci. 41 (2001) 830–835.
[15] R. Lowe, H.Y. Mussa, J.B. Mitchell, R.C. Glen, Classifying molecules using a sparse probabilistic kernel binary classifier, J. Chem. Inf. Model. 51 (2011) 1539–1544.
[16] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (2006) 1527–1554.
[17] Y. Bengio, Learning Deep Architectures for AI, vol. 2, Foundations and trends® in Machine Learning, 2009, pp. 1–127.
[18] G. Hinton, A practical guide to training restricted Boltzmann machines, Momentum 9 (2010) 926.
[19] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
[20] L. Hinton, D. Deng, G.E. Yu, N. Mohamed, A. Jaitly, V. Senior, P. Vanhoucke, T.N. Sainath, Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups, Signal Process. Mag. IEEE 29 (2012) 82–97.
[21] A. Lusci, G. Pollastri, P. Baldi, Deep architectures and deep learning in chemoinformatics: the prediction of aqueous solubility for drug-like molecules, J. Chem. Inf. Model. 53 (2013) 1563–1575.
[22] Y. Wang, J. Zeng, Predicting drug-target interactions using restricted Boltzmann machines, Bioinformatics 29 (2013) i126–i134.
[23] T. Unterthiner, A. Mayr, G. Klambauer, M. Steijaert, J.K. Wegner, H. Ceulemans, S. Hochreiter, Deep Learning for Drug Target Prediction, 2014.
[24] J. Ma, R.P. Sheridan, A. Liaw, G.E. Dahl, V. Svetnik, Deep neural nets as a method for quantitative structure–activity relationships, J. Chem. Inf. Model. 55 (2015) 263–274.
[25] T.B. Hughes, G.P. Miller, S.J. Swamidass, Modeling epoxidation of drug-like molecules with a deep machine learning network, ACS Cent. Sci. 1 (2015) 168–180.
[26] F. Ghasemi, A. Fassihi, H. Pérez-Sánchez, A. Mehri Dehnavi, The role of different sampling methods in improving biological activity prediction using deep belief network, J. Comput. Chem. (2016).
[27] V. Svetnik, A. Liaw, C. Tong, J.C. Culberson, R.P. Sheridan, B.P. Feuston, Random forest: a classification and regression tool for compound classification and QSAR modeling, J. Chem. Inf. Comput. Sci. 43 (2003) 1947–1958.
[28] A. Liaw, M. Wiener, Classification and regression by randomForest, R news 2 (2002) 18–22.
[29] L.S. Aiken, S.G. West, S.C. Pitts, Multiple Linear Regression, Handbook of Psychology, 2003.
[30] A.R. Leach, V.J. Gillet, An Introduction to Chemoinformatics, Springer Science & Business Media, 2007.