Article

Unsupervised Feature Learning in Time Series Prediction Using Continuous Deep Belief Network

Qili Chen [1], Guangyuan PAN [2*], Ming Yu [3], Jiuhe Wang [1]*

[1] Automation college at Beijing information science and technology University, Beijing, China; qilichen@hotmail.com
[2] Civil & Environmental Engineering at University of Waterloo , Waterloo, Canada; g5pan@uwaterloo.ca
[3] Electronic Engineering at The Chinese University of Hong Kong, Hong Kong SAR, China
* Correspondence: g5pan@uwaterloo.ca; Tel.: +1-226-978-9816

Featured Application: We have specific interests on using DBN, a deep learning model, for parameters predicting in environmental protection industries, not only because of the previous promising results, but also the beauty in DBN's unsupervised learning algorithm. In this paper, we also aim to analyze the model characteristics and extend its potential usages. By improving the unsupervised learning process algorithm, the proposed model now can process continuous data in a better way. In the experiments, we find that the model outperforms many other existing ones including the traditional DBN.

Abstract: A continuous Deep Belief Network (cDBN) with two hidden layers is proposed in this paper, focusing on the problem of weak feature learning ability when dealing with continuous data. In cDBN, the input data is trained in an unsupervised way by using continuous version of transfer functions, the contrastive divergence is designed in hidden layer training process to raise convergence speed, an improved dropout strategy is then implemented in unsupervised training to realize features learning by de-cooperating between the units, and then the network is fine-tuned using back propagation algorithm. Besides, hyper-parameters are analysed through stability analysis to assure the network can find the optimal. Finally, the experiments on Lorenz chaos series, CATS benchmark and other real world like CO2 and waste water parameters forecasting show that cDBN has the advantage of higher accuracy, simpler structure and faster convergence speed than other methods.

Keywords: Unsupervised training; Features learning; Deep learning; Time series forecasting

1. Introduction

Deep Belief Network (DBN) has been popularly focused on around the world since its good abilities in image and acoustic recognition tasks [1]. By simulating the cognitive and knowledge reference processes of human brain, DBN is famous for its special training method which is called greedy unsupervised training [2]. By stacking several Restricted Boltzmann Machines (RBM) one upon another in this process, DBN learns the features of input signals supervisor needlessly to obtain a better distributed representation, without requiring extra data with labels on it like Back Propagation [3]. Instead, DBN combines the processes of feature learning and supervised training together, avoiding features been selected manually and incorrectly [4]. The essence that the process of unsupervised training is in fact the weights initialization for supervised training is put forward [5]. Weights can be selected and trained into a very good area and ensure the step of supervised training more effective and faster than weights random initialization [6]. Nowadays, based on this philosophy, more and more Deep Neural Network models have been proposed and new records

 doi:10.20944/preprints201810.0494.v1

have been achieved in AI, especially in pattern recognition, such as feature extraction, voice recognizing, document categorization, semantic understanding, and so on [7-12].

Although it has achieved lots of improvements, DBN works in a way of data discretization in extracting features, thus all the visible and hidden units are Bernoulli values. Hence, little achievement and application of using DBN in industry area is published [13]. For example, a satisfied accuracy cannot be realized in time series forecasting tasks in data collection and detection systems [14]. Although many other deep learning models, such as convolutional neural networks (especially pretrained models), LSTM and deep reinforcement learning are able to process continuous data in time series prediction, DBN does not performs well in this application [33-34]. The driving force behind this research is that DBN is sounded for its feature learning ability and unsupervised learning, the real-world applications especially time series prediction will benefit from DBN's data processing ability.

The contributions of this paper are follows.

- First, to increase DBN's ability of processing continuous data, a continuous Deep Belief Network (cDBN) with two hidden layers is proposed in this paper, to realize modelling time series data in industries applications.
- Second, some training technics are improved, and their effectiveness are analysed, for example, Contrastive Divergence (CD) is implemented to train one hidden layer at a time; the output of the first RBM is set to be the input of the second RBM. In the unsupervised training process, an improved strategy of dropout in unsupervised learning is implemented to reduce over fitting by preventing co-adaptation of feature detectors, and this strategy helps to learn features intelligently and wisely because it reduces a neuron's dependency on others.
- Third, through stability analysis, the stability theorem is put forward to provide hyper-parameters-selecting.
- Finally, this cDBN is tested in two kinds of experiments; simulation experiments, for example, Lorenz chaos series and Competition on Artificial Time Series (CATS) benchmark, and real-world applications such as atmospheric oxygen carbon (CO2) forecasting and wastewater parameter prediction. The results show that it has higher accuracy, simpler structure, and better stability of using cDBN.

The following parts are designed as: Section 2 is the related works done by other researchers and us before. Section 3 describes the structure and algorithm of proposed cDBN, Section 4 introduces an improved dropout strategy for unsupervised learning, Section 5 is stability analysis, Section 6 shows the experiments using cDBN, and the final section gives the conclusions of this study.

2. Related Works

Researchers have done many significant works towards realising continuous deep belief network in time series prediction. Hinton proposes a method to receive and process continuous data for visible layer, however those visible and hidden units can still only deal with Bernoulli values in unsupervised training [15]. The author of [16] analyses the theory of training a Restricted Boltzmann Machine, and shows it has the ability of processing continuous data and can be used in network modelling. More approaches have been achieved recently by [17], in which an improved RBM, whose transfer function is changed to be able to process continuous data, is used to predict stock changing. By trying different numbers of units and layers, a satisfied prediction is realized. Besides, in order to solve the problem of co-adaptation in training, the strategy of dropout is proposed [18], this strategy drops units randomly in supervised training, and this prevents complex co-adaptations of feature detectors in hidden units, insuring that a unit can be helpful and independent in the context with several other specific feature detectors. But the problem of low effectiveness of feature learning in unsupervised training is still severe, especially in time series predicting tasks [19]. Kuremoto uses Particle Swarm Optimization (PSO) to calculate the number of hidden layer units before designing the structure of a DBN, however, this method still needs to pre-set a certain size for hidden layers, also it is still time consuming [20].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

We have also done some previous researches towards this direction. In [35], a special regularization-reinforced term is developed to make the weights in the unsupervised training process to attain a minimum magnitude. Then, the non-contributing weights are reduced, and the resultant network can represent the inter-relations of the input-output characteristics. Using this method, the optimization process is able to obtain the minimum-magnitude weights of DBN. In [36-37], we attempt to demonstrate the potential of this new model for crash prediction using DBN, the continuous version transfer function and the technic of regularization are carefully studied. According to these previous studies, DBN has been proved to have the potential for processing continuous data by benefiting its feature learning ability. So, in this paper, we will move on to demonstrate the developing method.

3. Methodology

3.1 The structure of cDBN

An illustration of the proposed cDBN is shown in Fig 1. This cDBN consists of 4 layers. The first layer is visible layer (V) which receives the original time series data at time t, (t-1) … (t-n), so it's the input of cDBN. V is followed by two hidden layers, L1 and L2. V and L1 formed the first RBM, and L1 and L2 make up the second one. The structure of each RBM is that of a two-way full connectivity between V and L (as shown by double sided arrows). No connections exist between units of the same layer. The absence of intra-layer connections results to a "restricted" Boltzmann Machine. In the training process, each hidden layer extracts the last layer's data information and feature to form a better although more abstract distributed representation of input data. The last layer is output which is only one unit. This layer also means another feature that has been learnt from input, and its mathematical meaning is the predicted value at time (t+1). The reason why two hidden layers are recommended is on traditional DBN, two layers are good enough in a pattern reorganization task, and they are definitely more powerful in extracting features than only one layer used [21-22], besides, multi layers will be time consuming for industries and real-world applications.

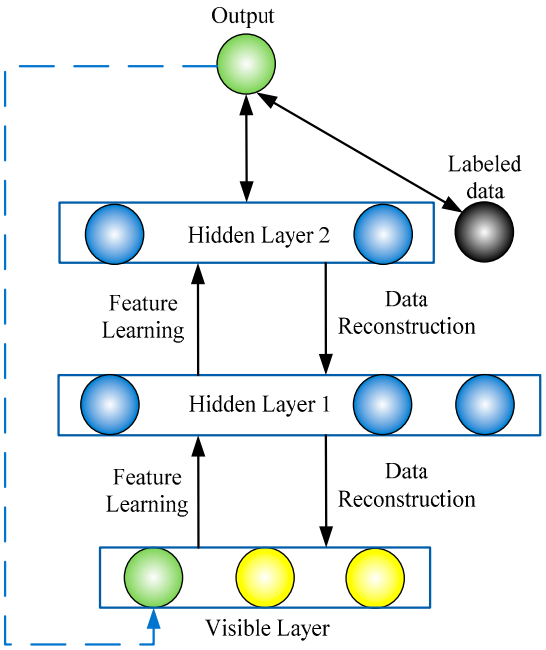


Fig 1 Continuous DBN's Structure

As DBN must transit the original signal to be Bernoulli values when processing, its applications are limited. Hinton improved its visible layer to receive continuous values, while the signal in the unsupervised training process will still be transited to Bernoulli value [15]. According to previous work that has been done by Bengio, RBM is capable of processing continuous data and modelling in theory [16]. Furthermore, some recent works have shown that continuous RBM can be used in stock

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

prediction [23]. A continuous RBM is put forward to predict water parameters in Huai River in China and gets good results in [24]. In the study of [20], particle swarm optimization (PSO) is used to select the best structure for DBN, and then the unit-amount-selected DBN is used in the task of CATS benchmark, good results are obtained. Based on the works that have been done, the transfer functions in DBN are improved to process continuous data in this paper, and then an improved dropout in the training process is studied.

3.2 The training processes of cDBN

DBN is made of several RBMs, which are trained separately before unfolding the whole network to use back propagation in weights' fine tuning. So DBN is a kind of recurrent neural network that combines unsupervised and supervised learning together. The knowledge learning and reasoning processes in a RBM, which can be called knowledge generation, are shown in formulas (1) and (2).

$$p(h_j = 1) = \frac{1}{1+e^{-b_j-\sum_i v_i w_{ij}}} \quad (1)$$

$$p(v_i = 1) = \frac{1}{1+e^{-c_i-\sum_j h_j w_{ji}}} \quad (2)$$

where vi is the value of unit i of visible (input) layer, hj is the value of j in hidden (output of RBM) layer. b and c stand for the biases of visible and hidden layers respectively. wij is the weight between visible unit i and hidden unit j. Define θ=(W,b,c), then according to Contrastive Divergence [25] the formula of renewing weights and biases is shown as follows.

$$\theta^{(\tau+1)} = < h_j^0 v_i^0 > - < h_j^1 v_i^1 > \quad (3)$$

where <> denotes the average over the sampled states, hj0 vi0 means the state of visible layer multiples hidden layer at the beginning, and hj1 vi1 means the same except both layers have ran Markov chain once.

While in cDBN, the transfer functions will be improved to process continuous data. Let sj and si be the values of hidden unit j and visible unit i, and the sigmoid function in (1) and (2) will be kept, while the step of discretization is omitted. After that, an item of noise function will be added to realize the transformation.

$$s_j = \varphi_j(\sum_i w_{ij} s_i + \sigma \cdot N_j(0,1)) \quad (4)$$

$$s_i = \varphi_i(\sum_j w_{ij} s_j + \sigma \cdot N_i(0,1)) \quad (5)$$

At the same time,

$$\varphi_j(x_j) = \theta_L + (\theta_H - \theta_L) \cdot \frac{1}{1+e^{-a_j x_j}} \quad (6)$$

$$\varphi_i(x_i) = \theta_L + (\theta_H - \theta_L) \cdot \frac{1}{1+e^{-a_i x_i}} \quad (7)$$

(4) and (5) are the processes of learning and generating, in which, N(0,1) is a Gaussian random variable with mean 0 and variance 1. σ is a constant, and φ(·) denotes the sigmoid function with asymptotes θ_H and θ_L. a is a variable that controls noise, which means it controls the gradient of transfer function.

4. Improved unsupervised learning

When a large size of cDBN is trained on a small training set, it typically performs poorly on held-out test data. This is because of over fitting, which means the network is trained to accurately recognize the examples presented in the training as separate cases ignoring possible associations between them, and the feature detectors (hidden units) have been tuned to work well together on the training data but not on the test data. Units in the same layer are supposed to be independent from each other in cDBN, but sometimes they won't, since the information is calculated recurrently in the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

training, and a unit can affect another one by doing this procedure because of the phenomenon called explaining away [2]. As a fact of this, some neurons are effective only when the one it relies on is learns the right information, so bad neurons with wrong knowledge will fail features-learning in others through over fitting in both unsupervised and supervised training.

The strategy of dropout is brought up in [18] and has shown promising results. It gives big improvements on many benchmark tasks and even sets new records for some pattern recognition tasks [26]. Dropout is used in the process of fine tuning, preventing complex co-adaptations on the training data. On each presentation of each training case, each hidden neuron is randomly omitted (the weights connect to which are kept, only the neuron is ignored) from the network according to a modified probability. The omitted neurons will not be considered as a part of the network, so the neurons' values are not updated, and the weights are kept the same as what they are in the last training case, and they will restart working in the next case. In the stage of fine tuning, an upper bound of L2 norm is pre-set for each hidden unit's weights instead of using any L2 norm penalty factor. And in the training process, if weights are over range of the bound, normalization on the weighs will be acted. By doing this, the weights will have a better learning rate to find more potential solution spaces. And in test stage, the "mean work" method is performed to calculate the output of the feed forward network. Mean work used in the test set by activating all of the hidden units and attenuate their weights with a certain proportion. This method acts well in some cases. However, since it is used in fine tuning, it needs to be modified before being applied in cDBN.

Since values in cDBN's units are all continuous and over fitting is still a serious problem in unsupervised training, so an improved unsupervised training process with dropout is proposed in this paper. The formula of dropout can be described as follows,

$$s_j = s_j \cdot dropF \tag{8}$$

$$dropF = \begin{cases} 1, & n \geq r_{dr} \\ 0, & n < r_{dr} \end{cases} \tag{9}$$

where s_j is the state of unit j in hidden layer, it will be decided to be constant or be turned to 0 in one case by dropF, which is dropout fraction as be calculated in (9). n is a random value between [0, 1], and $\boldsymbol{r_{dr}}$ means how many percentages of units in a layer will be turned to 0. In the procedure of CD, dropout is supposed to be applied in the processes of calculating s_{j0} and s_{i0}, because these two states affect the next two states in CD, an all of the weights will be recalculated according to s_{j0} and s_{j0}. And it should be noticed that in some time series forecasting tasks, there are not too many units in the first visible layer, and one of them is supposed to take the value of output in the last case, in this case only s_{j0} will perform dropout, otherwise some important information will be lost in the knowledge reconstruction of s_{j0}. When using Contrastive Divergence, the update forms of weights and biases are as follows,

$$\Delta w_{ij} = \eta_w(< s_i^0 s_j^0 > - < s_i^1 s_j^1 >) \tag{10}$$

$$\Delta b = \frac{\eta_b}{b^2}(< {s_j^0}^2 > - < {s_j^1}^2 >) \tag{11}$$

$$\Delta c = \frac{\eta_c}{c^2}(< {s_i^0}^2 > - < {s_i^1}^2 >) \tag{12}$$

where ηw, ηb, ηc are the learning rate of weight and biases, and <·> still means what has been addressed above.

So,

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t) \tag{13}$$

$$b(t) = b(t-1) + \Delta b(t) \tag{14}$$

$$c(t) = c(t-1) + \Delta c(t) \tag{15}$$

where t is the t^{th} epoch been run.

Fig.2 shows the schematic diagram of improved dropout in unsupervised learning process, which is the procedure of unsupervised training in a RBM in cDBN. Dashed lines mean that the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

neurons are omitted temporally other than permanently, since they will be reactivated in the next epoch. By using improved dropout in unsupervised training, hidden units' values are turned to be 0 randomly according to weights updating rules (8) and (9). This strategy prevents the affection of co-adaptation between some feature detectors since this makes two units in a same layer won't appear in the same epoch all the time. So the update of weights won't rely on the interaction of two units, and dropout forces them find more solution spaces and learn knowledge independently and wisely. Dropout can be seen as a kind of mean work, because cDBN with different structures will be activated for each training case, and finally, all of the units will be activated for test set. As regularization is often being used to reduce over fitting, and in fact native Bayes regularization method is a special case of dropout, because native Bayes assumes that the feature detectors are already independent. Native Bayes can learn each feature at a time if training set is small, and still has good results. But improved dropout can learn more than one feature at a time and combine them together at last when all of the detectors are activated, realizing features learning better.

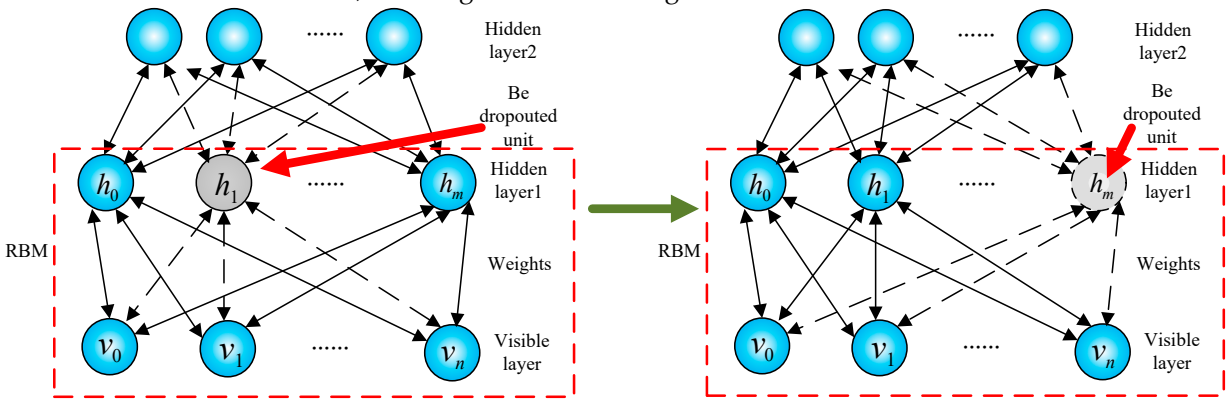


Fig 2 Improved Dropout in Unsupervised Learning

A summarized cDBN training procedure with improved dropout to reduce over fitting is shown as follows.

Step 1. Decide how many layers will be included and the size of each layer, and limitation of iteration number.

Step 2. Initialize cDBN.

Step 3. Put the original data into visible layer which contains m units, and start calculating using formula (4) and (6). By running (8) and (9), the dropout fraction, to get the left units in the epoch, and then perform (5) and (7) which is to calculate the generated data of input, in which procedure, dropout will also be used to omit the units in the first hidden layer.

Step 4. Update weights and biases using CD.

Step 5. If the stopping condition which has been set in the experiment below is reached, go to step 6. Otherwise go back to step 3.

Step 6. If the stopping condition in the last layer is reached, go to step 7. Otherwise, set the output of this RBM be the input of the next one.

Step 6. Turn to step 3 and run the next RBM.

Step 7. Compare the expected output with the real output to calculate the error. Use back propagation in the process of fine tuning.

Step 8. The output of the final output in the last layer is transited to be a unit of the visible layer in the next training case. If this is not the last case, turn to step 3, otherwise, go to step 9.

Step 9. Stop running.

5. Convergence and stability analysis

cDBN can be seen as a probability generation model that contains two hidden layers, as well as a complex model that is made of several simple models. In fact, each pair of neighboring layers of cDBN is a continuous RBM, so the method of unsupervised training can be used in the training process. As a probability generation model which is undirected and based on network energy theory,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

continuous RBM uses Contrastive Divergence algorithm's one step iteration in data pre-train stage to update network weights. Traditional training way transfers the values of hidden and visible units to be Bernoulli value (0 or 1), other than using the probabilities of units themselves. However, in cDBN in this paper, the contradiction theory is avoided by taking the probability to be output by using formulas (6) and (7), so Contrastive Divergence can still be used as a rapid training strategy. Then it will be,

$$s_i^0 \epsilon [0,1] \tag{16}$$

$$s_j^0 = \varphi_j\left(\sum_i w_{ij} s_i^0 + \sigma \cdot N_j(0,1)\right) \tag{17}$$

$$s_i^1 = \varphi_i\left(\sum_j w_{ij} s_j^0 + \sigma \cdot N_i(0,1)\right) \tag{18}$$

$$s_j^1 = \varphi_j\left(\sum_i w_{ij} s_i^1 + \sigma \cdot N_j(0,1)\right) \tag{19}$$

The algorithm of CD is to approximate maximum likelihood method, and it is used to learn each weight of continuous RBM in a cDBN. After learning the first RBM, the second one will be stacked on top of the first one, and the output of the first RBM becomes the input layer of the second RBM. By doing this, a multi-layer model is formed. In the procedure of stacking, the input layer of the first continuous RBM is initialled by the original data. When the weights are trained, the output will be copied directly to the input of the second continuous RBM, and then repeat the same training procedure until the termination conditions are reached. So sj1 in equations (16) to (19) is the output of the first RBM (RBM1) as well as the input of the second RBM (RBM2).

In the unsupervised learning step, by training one RBM, the algorithm of contrastive divergence (CD) can be seen as a simple version of Markov Chain Monte Carlo (MCMC), so the error after training one RBM may not be 0, but should be close to it. In this part we use ϱ to be the error. Then there is,

$$0 < \rho < 1 \tag{20}$$

And there could be several (n) RBMs in a traditional DBN, depending on the tasks. So theoretically, the final error of DBN could be close enough to 0, which is

$$\rho^n \approx 0 \tag{21}$$

In the process of supervised learning, back propagation will be used, and the training error of a BP network has been researched, which can be 0 according to Weierstrass theory [3]. So the convergence of cDBN will be assured.

As a fast training algorithm, CD is in fact not in any approximate function's gradient decent directions, and as a result, its approximation for maximum likelihood is very rough, so in order to assure this proposed network is capably and stably converging to the solution space and to find the optimal solution, it is necessary to use theory to select hyper parameters through stability analysis for cDBN. And only by analyzing this can the proposed strategy of CD and dropout have the theoretical basis to be used in cDBN. So, by what have been stated above, we've got s_j^1 (RBM1) = s_i^0 (RBM2), and according to (16), the output of RBM should be [0,1], so the convergence condition of cDBN is to make sure $s_j^1 \epsilon [0,1]$. And the theorem of stability is put forward as follows.

Theorem. When cDBN is stable, $s_j^1 \in [0,1]$, if and only if $s_j^0, s_i^1 \in [0,1]$.

Proof. Let $s_j^0 \in [t_L^1, t_H^1]$, $s_i^1 \in [t_L^2, t_H^2]$, where t_L^1, t_H^1 are lower and upper limit of sj^0, t_L^2, t_H^2 are lower and upper limit of s_i^1.

1) Sufficiency:

If the middle states of s_j^0, $s_i^1 \in [0,1]$, then when training a RBM using dropout, according to (17)-(19), the output is $s_i^1 \in [0,1]$, which is within the prescribed scope. So, cDBN is stable.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

2) Necessity:

① If cDBN is stable, and if $t_L^1>0$. As the transfer function is monotonous, the middle states of s_j^0, s_i^1, which meet $s_i^1> s_j^0$, can be calculated by CD. Hence, $t_L^2> t_L^1$. And according to (19), $s_j^1> s_i^1$, which means the output of cDBN is bigger than 0. This contradicts to assumption.

② Hence $t_L^1\leq 0$. Similarly, $t_L^1\geq 0$.

③ So, $t_L^1=0$. By using the same proof, we have t_H^1.

④ Hence $s_j^0\in[0,1]$.

⑤ Similarly, when doing Gibbs sampling using CD, the output is also between [0,1], which means $s_i^1\in[0,1]$.

Q.E.D.

From the proof we know that the process of dropout doesn't affect the stability of cDBN, and according to the theorem, there is

$$s_j^0 = \theta_L + (\theta_H - \theta_L)\cdot\frac{1}{1+e^{-a_j x_j}}\ \ s_j^0 = \theta_L + (\theta_H - \theta_L)\cdot\frac{1}{1+e^{-a_j x_j}} \tag{22}$$

will be assured, and this equation is the selecting basis of hyper parameters. If Gaussian noise is set to be 0, for every x>0, we have $0<e^{-x}<1$, so $s_j^0\in[(\theta_H+\theta_L)/2, \theta_H]$. Then the selection of $\theta_L=-1$, $\theta_H=1$, $a_j=1$ can make formula (15) workable. In this cDBN, there are only two hidden layers. According to the convergence and stability analysis, no matter how many RBMs are stacked in a cDBN, if the input and output of cDBN are [0, 1], the intermediate state of learning must be continuous values in [0, 1]. It makes sure that the network is stable and reliable in training process.

6. Experiments and Analyses

Time series forecasting is one of the most common and important tasks in real world. It uses statistics and analysing methods on historical data, learns the inherent law and trend, then predicts the possible value of the future data before providing decisions for industries. Because the sensitive features of inner and outer environment, modelling and predicting for time series data is a complex nonlinear problem, which is important and challenging. Traditional methods, such as linear processing and least square, process the input data in a special way before using, and usually resulting in bad predictions, and thus they are not suitable to solve these tasks. cDBN trains data in an unsupervised way and has very powerful feature learning and extracting function. Besides, according to Weierstrass's first theorem, Neural Networks (NNs), as a data-driven model, can approximate any nonlinear functions in any accuracy. In additionally, cDBN can be seen as a black-box model in the training process. So in order to test the short term and long term predicting abilities, cDBN is put forward in some experiments below. The input of cDBN is the real data of time series, and in the experiment of long term prediction, the last predicted data will be set as one of the input unit in the next step of prediction. Although there is still lack of good algorithms for deciding the size of each hidden layer and empirical has to be implied here, improved dropout has the ability of reducing over fitting. Different layer sizes can be tried, and the larger one can be chosen if multi-tasks are presented. Last but not least, only one output neuron is used in cDBN which equals to the predicted number.

Four experiment belonging to computational simulation and real-world application are explored in this section. First, the simulation experiments of Lorenz chaos series prediction will be conducted. Second, a long-term prediction called CATS Benchmark missed data prediction will be

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

utilized to testify the model. Once the model has been proved effective, two real-world applications will be introduced in experiments three and four. In the first real-world application, the technic of unsupervised dropout will also be tested, and the performance variance with right to different dropout rates will be analysed.

6.1 Simulation Experiment 1: Lorenz chaos series prediction

The experimental target is to check whether the proposed cDBN is capable of predicting short term data accurately with a simplified structure, whether the improved dropout can further improve the accuracy. Chaotic time series forecasting is a typical benchmark problem on testing new methods of neural networks, and since Lorenz effect indicates the sensitive dependence on initial conditions of chaos, is barely feasible to forecast long term data in this task. However, short-term prediction such as one-ahead prediction can be realized if a predictor approximates the nonlinear system enough. The Lorenz chaos equations are as follows,

$$\begin{cases} \frac{dx}{dt} = \sigma(y_t - x_t) \\ \frac{dy}{dt} = -x_t z_t + \gamma(x_t - y_t) \\ \frac{dz}{dt} = x_t y_t - \beta z_t \end{cases} \tag{23}$$

where x is convection intensity, y indicates horizontal temperature difference between rising airflow and sinking airflow, while z is vertical temperature difference between them. The parameters are σ=10, γ=28, β=8/3.

In the experiment, time series data on dimension x is processed. In the generated series, the first 1,000 sets are omitted to reduce affection of initial state on the series. The training set has 1,500 values, and test set is the following 1,000 numbers. Three nodes in input layer are introduced to do one step prediction, that is to say, values at time (t-2), (t-1), t are set to be input, and the output is value at the next time t+1. RMSE is introduced as the evaluation index,

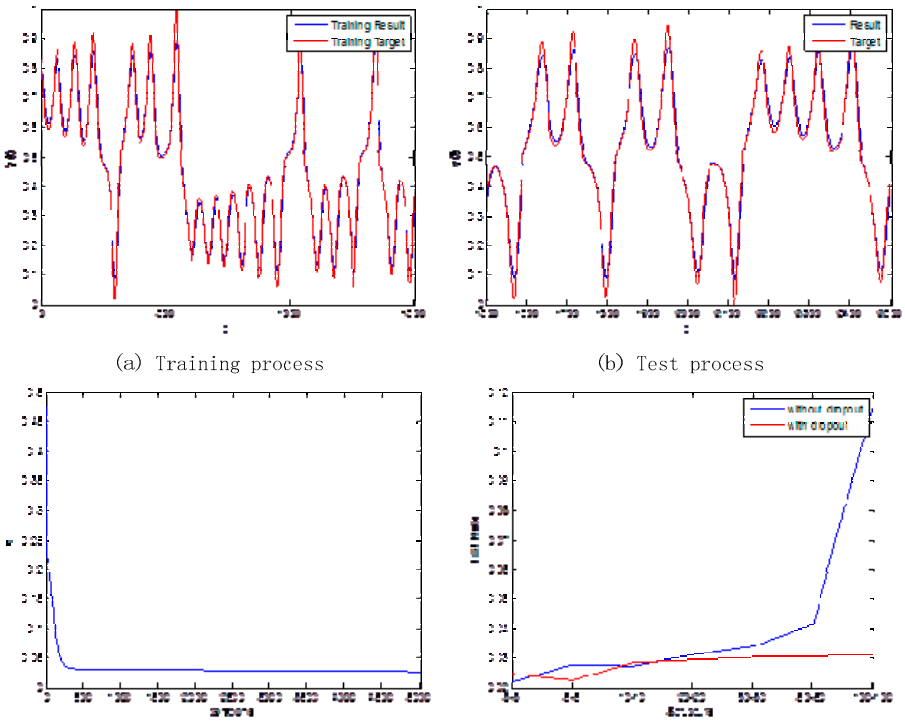
$$RMSE = \sqrt{\frac{\sum_N (y(k) - \hat{y}(k))^2}{N}} \tag{24}$$

where y(k) and $\tilde{y}$(k) indicate the ideal output and observed output at time k respectively. In order to verify improved dropout's effectiveness, different kinds of structures will be tried from 10-10 to 100-100, and the improvement will be recorded. Epochs in the stagy of automatic knowledge learning (unsupervised learning) is 100 and fine tuning (supervised learning) learning rate is 2. Dropout fraction in unsupervised learning is 0.2, which means 20% of the hidden units will be set to 0 randomly in training process. Fine tuning iterations are set as 5,000. The results are shown in Table 1 and Fig 3.

Table 1 Results of Lorenz experiment

Methods	Hidden units	Training RMSE	Test RMSE	Computing time/s
cDBN	**5-5**	**0.0213**	**0.0228**	**3.245**
DBN[2]	5-5	0.0263	0.0280	3.750
SRNN[27]	6	0.0232	0.0302	6.750
BPNN[28]	6	0.0700	0.0835	>10

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1



(a) Training process (b) Test process

(c) Training error (d) Comparison

Fig 3 Results of Lorenz experiment

From Fig.3 (a) and (b), it can be seen that even though it is hard to learn the knowledge at some of the sharp places, cDBN is still capable of realizing rapidly learning and tracking the function line well. With the structure of 3-5-5-1, cDBN with dropout is trained with the training error 0.0213. And in testing stagy, the network responds somehow at the same quality, it presents very good approximate ability and generalization ability. Also, the test RMSE is reduced to 0.0228, better than DBN without dropout and some other methods. BPNN is the most classical model in solving the problem, but both the training and test error are big. And SRNN is one outstanding method that is proposed in [27], by using which the test error is a little bigger than using DBN, the training error is smaller. But when applying the proposed network in this paper, we've had both very small training and test error in the experiment. In fact, the time consuming is a little less because the computing speed is more depend on contrastive divergence than other factors, and sometimes it's a little faster due to the time saving by not computing a part of the neurons.

Fig.3 (c) shows the error decreasing line in fine tuning, from which we can see that the convergence speed is quite rapid thus it saves computing time. That is because the weights have been initialized by unsupervised training are already in a good solution area. Fig 3(d) presents the comparison of cDBN that has the same structure both with and without dropout. The red line shows test RMSE of cDBN with dropout while the blue one is without. From Fig 3(d) we can see when the hidden units are 3-3, cDBN without dropout acts better, that is because when the size of the layer is small, features in each of them are too important to be ignored. Then, when more units are added, over fitting becomes a problem so the test RMSE increases. However, cDBN with dropout improves the learning effectiveness by preventing co-adaptation of feature detectors, and test RMSE can be kept at a low level.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

Besides, when the hidden units are the same, cDBN with dropout strategy basically responds better than some other methods. This is because cDBN uses the second hidden layer to extract more features from the first hidden layer, and is capable of learning a better distributed presentation of input.

6.2 Simulation Experiment 2: CATS Benchmark missed data prediction

The second experiment is to verify whether cDBN is capable of predicting long term data accurately, and neurons respond better than those without dropout. The used database is Competition on Artificial Time Series (CATS benchmark), which was brought up in [29] and later became a common benchmark to test new models in AI. This benchmark consists of 5 data blocks, each of which has 1,000 sets of data, the first 980 are given and the rest 20 are missed, see in Fig 4.

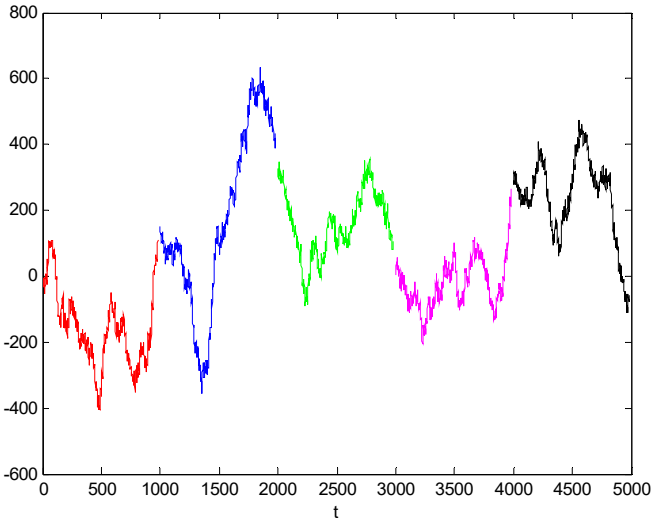


Fig 4 CATS benchmark Data

So in all the 5 blocks, 100 sets in total are needed to be predicted, and then the results can be compared by E1 and E2.

$$E1 = \frac{\sum_{981}^{1000}(x(t)-\hat{x}(t))^2}{100} + \frac{\sum_{1981}^{2000}(x(t)-\hat{x}(t))^2}{100} + \frac{\sum_{2981}^{3000}(x(t)-\hat{x}(t))^2}{100} + \frac{\sum_{3981}^{4000}(x(t)-\hat{x}(t))^2}{100} + \frac{\sum_{4981}^{5000}(x(t)-\hat{x}(t))^2}{100} \quad (25)$$

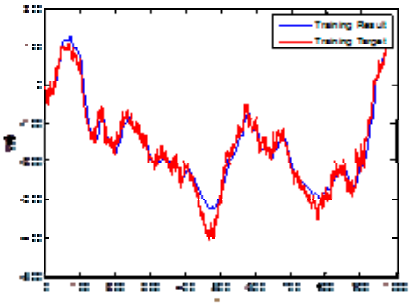
$$E2 = \frac{\sum_{981}^{1000}(x(t)-\hat{x}(t))^2}{80} + \frac{\sum_{1981}^{2000}(x(t)-\hat{x}(t))^2}{80} + \frac{\sum_{2981}^{3000}(x(t)-\hat{x}(t))^2}{80} + \frac{\sum_{3981}^{4000}(x(t)-\hat{x}(t))^2}{80} \quad (26)$$

Although the line is quite sharp, the data in CATS still has certain internal laws, and finding them is the task for DBN before predicting missed data. For each of the first 4 missed blocks, both the information before and after the missed block can be used in the training process to learn as much knowledge as it can. But for the last missed block, no later information is given, so only the data before can be used. As a result, mission E1 is harder than E2, because E1 has to concern all of the five missed blocks while E2 only has to concern the first four.

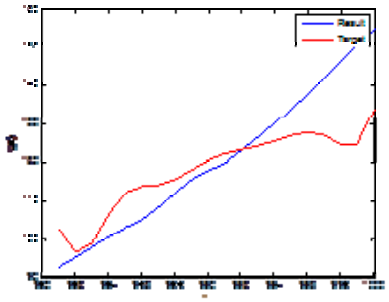
As there are 20 numbers in each missed block, it's a long term prediction task. The input nodes are set to be 10, and in the testing process, the last case's prediction, x(t), will become one of the input node in the next prediction for x(t+1). In order to train the training set and to verify dropout's effectiveness, and to make the comparison easier, a fixed structure of 10-20-40-1 will be chosen.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

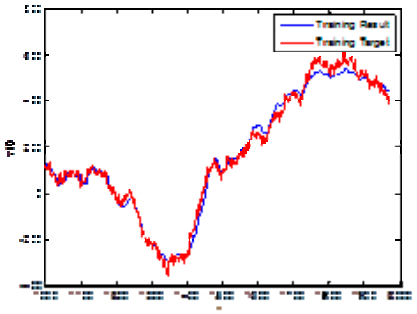
For each missed block, the sets before and after it are trained at once. The epochs in knowledge learning process are 200, dropout fraction is 0.5, which means half of the hidden units are dismissed in each epoch, but they will return in the next epoch. Learning rate in fine tuning is between 0.5 and 1, fine tuning iterations are set as 5,000. Results can be found in Fig 5. The results indicate that for each block, cDBN using improved dropout attains a very good learning result and reaches to a satisfied prediction. The original line of CATS is extremely sharp especially in short term, since there are many obvious ups and downs in it. The learning line of cDBN is relatively smooth, which means it has good generalization ability for the missed numbers.



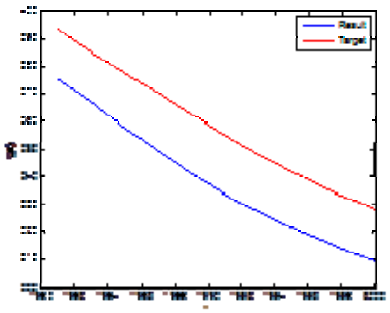
(a) Training block 1



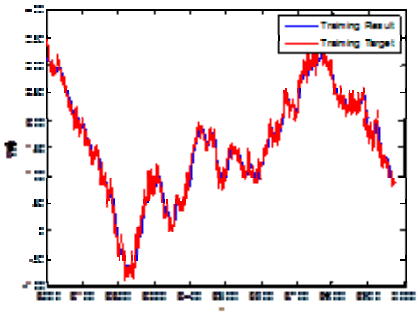
(b) Testing block 1



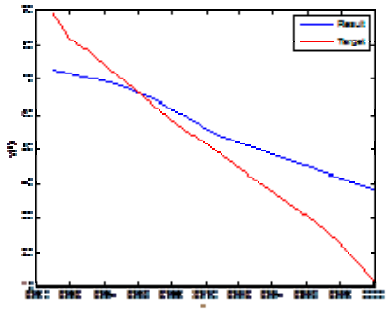
(c) Training block 2



(d) Testing block 2

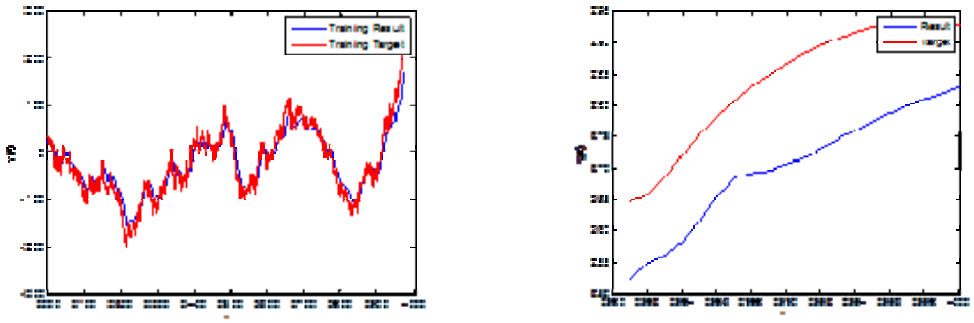


(e) Training block 3

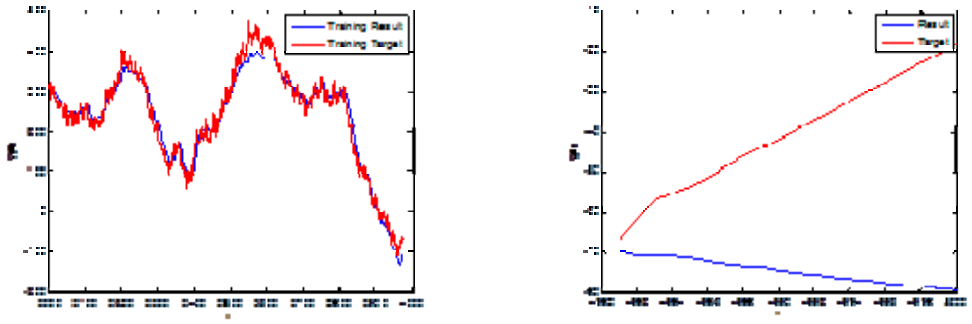


(f) Testing block 3

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1



(g) Training block 4 (h) Testing block 4



(i) Training block 5 (j) Testing block 5

Fig 5 CATS benchmark experimental results of cDBN using improved dropout

Fig 5(b) and (f) indicate the prediction for the first and the third block, from which we can see the testing lines are very close to the original ones. Fig 5(d) and (h) are the testing lines of missed block 2 and 4, and they are somehow parallel lines which show the network has learnt enough knowledge in predicting and the result line has the same trend with an error of around 20. Fig 5(j) indicates the prediction on block 5, and it can be seen the network is hard to make a good prediction and the trend is wrong. This result meets what have been discussed above.

Table 2 Comparison between different models in CATS benchmark

E1	Model	E2	Model
459.151	**cDBN**	**215.611**	**cDBN**
408	Kalman Smoother[30]	222	Ensemble Models[31]
771.455	DBN[18]	618.912	DBN[18]
1215	PSO-DBN[20]	979	PSO-DBN[20]

Table 2 compares the results with some different other models. Kalman Smoother and Ensemble Models in the second row got the first place in IJCNN04's competition. When using cDBN without dropout, we've had E1 = 771.455, E2 = 618.912 worse than the both two models but better than the others such as Hierarchical Bayesian Learning and MLP. And if the dropout fraction is set as 0.5, the test errors are reduced to E1 = 459.151, E2=459.151 which is a big step forward than the one without dropout in row 4.

The CATS benchmark experiment shows that cDBN is capable of predicting long term data accurately and the strategy of dropout makes the prediction better by overcoming over fitting

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

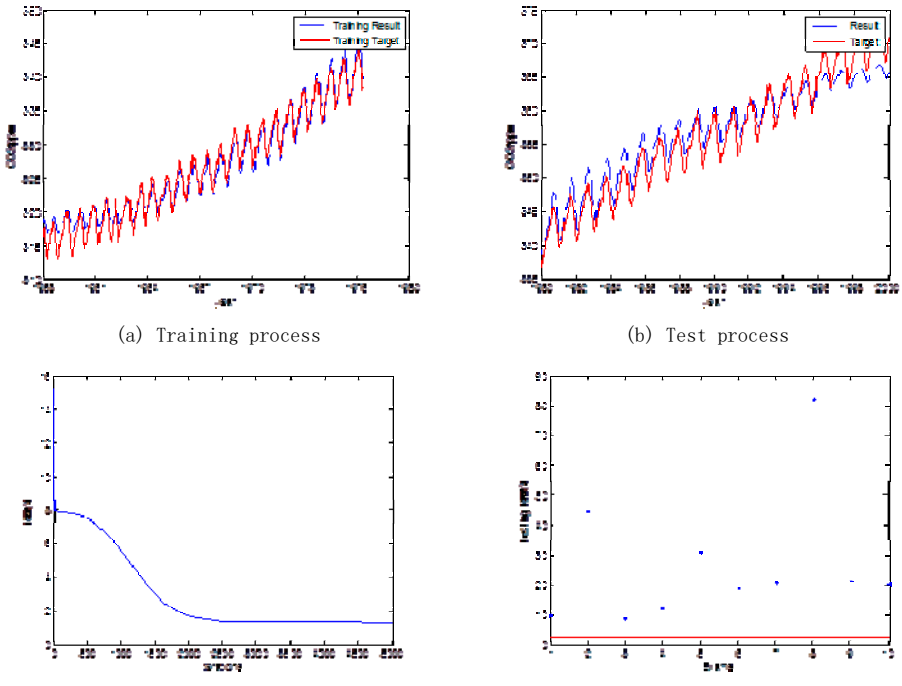
problem. Although this model cannot select network size such as how many layers would be the best, the strategy of improved dropout helps to improve the learning effectiveness by changing network structure randomly in the unsupervised training process, thus it can also be seen as a structure self-organizing model. And if more structures are implied, more accuracy results may be realized.

6.3 Real-world application 1: CO2 forecasting

The experiment of this part is to verify whether cDBN with dropout can be used in real world applications that contain noise, and whether the convergence is stable. cDBN transfers the results of pre-training in deep belief network to the initial weights in the next neural network, and then uses back propagation algorithm to do fine tuning. This training method is quite valuable especially in those tasks whose training set is small, this is because, as a matter of fact, initial weights can have a significant affection on the final models that are trained, and the weights that have been pre-trained are nearer to the optimal weights in solution space than those that have not. However, over fitting will be another serious problem in this condition since the weights are easy to be over trained.

Environmental issue has been one of the most concerned themes of the world. CO^2 concentration change is caused by many factors, such as geographical factors, human social activities, marine monsoon and so on. Human society is affected significantly due to CO^2 change. For instance, when density of CO^2 is too high, greenhouse effect can be a furious problem in metabolism. As a result of this, forecasting CO^2's density is not only meaningful for scientific investigation, but also provides reference for local government and factories to control CO^2 emission. 2.529,

Because of the errors caused by external factors, sampling CO^2's density in short term can barely reflect local CO^2's change, as a result, studying on long term CO^2's density after being de-noised is necessary. In order to assure the accuracy of the sampled data, CO^2 data that is used to forecast the global climate change should be sampled in those reigns that haven't been polluted by urban discharges.



(a) Training process (b) Test process

(c) Training error (d) Comparison

Fig 6 Experimental result of CO2 forecasting

 doi:10.20944/preprints201810.0494.v1

In this experiment, the data of Mauna Loa, Hawaii 2,220 sets in total from 1965 to 2000 is used [32]. Training set is 1,220 sets and the following 1,000 are left for testing. Also, in order to test improved dropout, different kinds of structures will be attempted from 10-10 to 50-50, and the improvement will also be recorded. Epochs in the step of knowledge automatic learning are 100 and dropout fraction is 0.5. Learning rate in fine tuning (supervised learning) is set to be 3 and the iterations are 5,000. The result with 40-40 neurons is shown in Fig 6.

Fig 6(a) is the training process while (b) shows the test process, and (c) represents the training error in fine tuning. It can be seen from (a) that the density of CO^2 changes regularly according to seasons, and has a trend of rising year by year. cDBN is capable of de-noising in this experiment, and the training and testing results turn out to be good.

In the features-learning stagy, the network acts bad at first but it manages to find the inter rules rapidly. And in the generalization stagy, the network responds accurately in the beginning and finally fails to forecasting, this may be because the features of human activities at the end of last century have not been learnt well enough.

Table 3 Result data of CO2 forecasting

Training RMSE	Test RMSE	Mean	Variance	Methods
1.322	2.483	2.510	0.0012	cDBN
0.001-0.01	8.76-82.49	27.07	492.1	BPNN[3]

Table 3 is associated with Fig6(d), which represents the results of using cDBN and BPNN, and it is easy to be seen that cDBN is better in most places. By running each program 10 times, training errors in BPNN are quite small, which are around 0.001-0.01, which shows a good chance to be over fitting. The tests RMSE are 9.7852, 44.6642, 8.7591, 12.3830, 31.0149, 18.8894, 20.8709, 82.4998, 21.5635, and 20.2909. The mean value is 27.0721, and the variance is 492.1079. While tests RMSE of improved cDBN the test RMSE are 2.483, 2.495, 2.519, 2.529, 2.491, 2.500, 2.601, 2.489, 2.492, 2.500. The mean value is 2.510 and variance is 0.0012, which means it can almost always convergent to the optimal.

Table 4 and Figure 7 records the improvements of cDBN with different dropout rates. When the hidden units are set to be 10-10, cDBN without dropout acts better, that is probably because when the units are few and no need to concern about over fitting, and features in each of them are too important to be omitted. When putting more nodes in hidden layers, the base test RMSE will raise, but dropout improves the learning effectiveness by preventing co-adaptation of feature detectors, keeping test RMSE at a low level. And in different conditions, different degrees of improvement have been achieved. For example, with the structures of 20-20, 30-30 and 50-50, the decrease of errors from 4.43% to 65% can be achieved, and when the hidden units are 40-40, the error can be decreased by as much as 69.55%. It can also be observed that the final error obtained by the model is sensitive to the choice of dropout fraction. The best performance will be achieved when structure is 40-40 with dropout rate 0.5, which is very close to using a structure of 20-20 directly, as a result, this method can also provide a way of find the best structure with high stability in real-world application. This experiment also finds that a lager structure with dropout rate 0.5 is easier to achieve higher prediction accuracy.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1

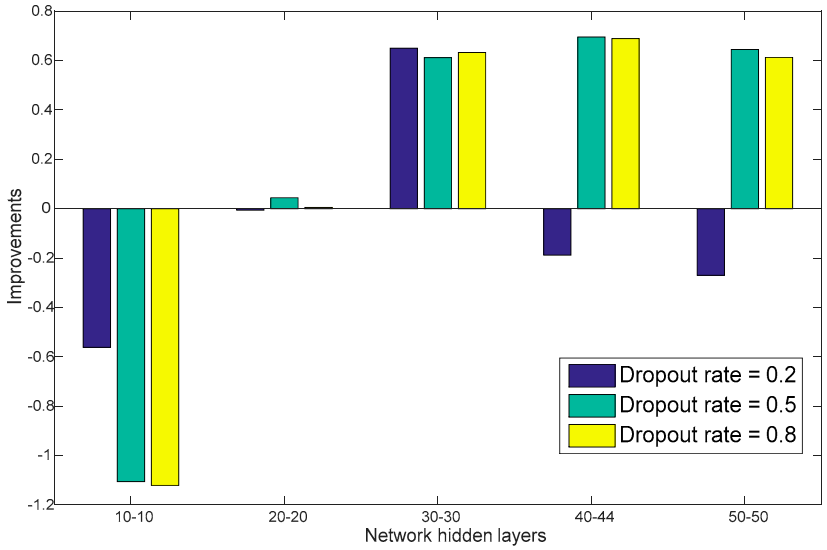


Fig 7 Improvements of different dropout rates

Table 4 Comparison of different dropout rates

Structures	Dropout	Training RMSE	Test RMSE	Improvement
10-10	0	1.383	2.887	
	0.2	1.166	4.509	-56.18%
	0.5	1.101	6.079	-110.56%
	0.8	1.084	6.123	-112.09%
20-20	0	1.355	2.733	
	0.2	1.292	2.750	-0.62%
	0.5	1.315	2.612	4.43%
	0.8	1.312	2.719	0.51%
30-30	0	0.933	7.781	
	0.2	1.261	2.720	65.04%
	0.5	1.244	3.018	61.21%
	0.8	1.256	2.857	63.28%
40-40	0	0.935	8.155	
	0.2	0.884	9.689	-18.81%
	0.5	**1.322**	**2.483**	**69.55%**
	0.8	1.370	2.533	68.94%
50-50	0	2.163	7.124	
	0.2	0.950	9.045	-26.97%
	0.5	1.375	2.530	64.49%
	0.8	1.418	2.756	61.31%

6.4 Real-world application 2: Wastewater parameter prediction

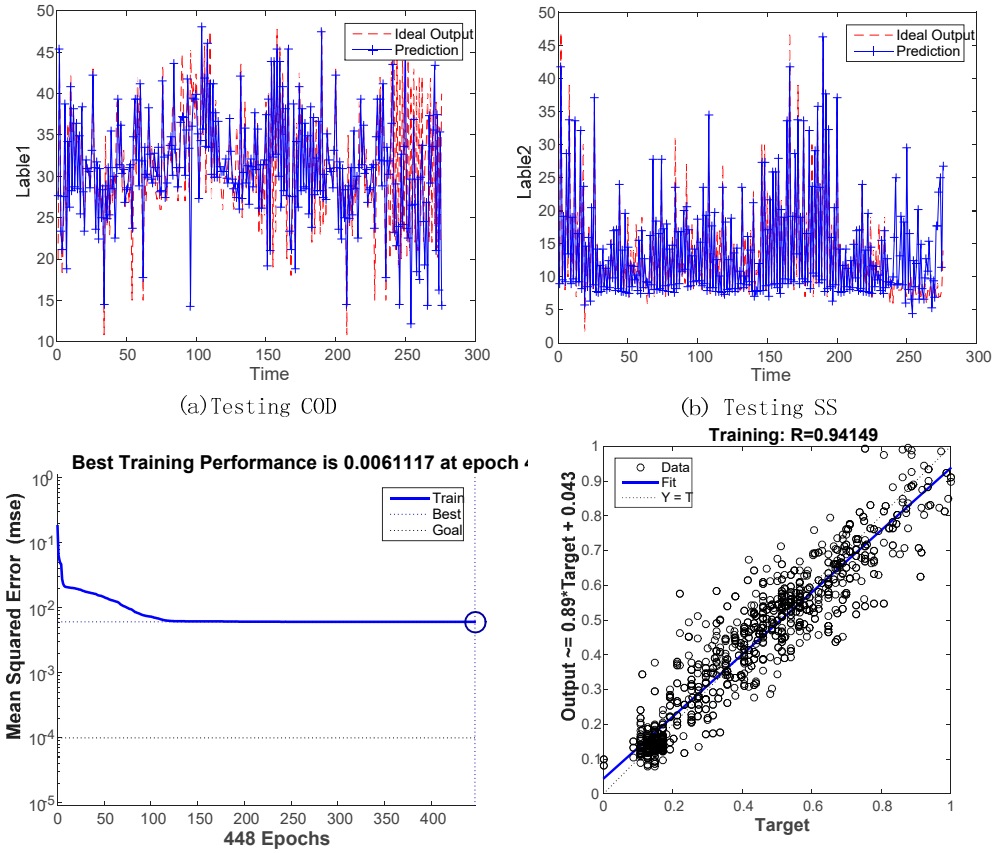
In this experiment, cDBN will be applied to another real-world application, wastewater parameter prediction. In recent decades, wastewater problem has become one of the major

environmental concerns. Wastewater treatment process (WWTP) is a highly nonlinear dynamic process. In order to minimize microbial risk and optimize the treatment operation, many variables must be obtained real-timely, in which, biochemical oxygen demand (BOD), chemical oxygen demand (COD), suspended solid (SS), PH level and nutrient levels, total nitrogen (TN)-containing, total phosphorus (TP)-containing are the most important ones. Subject to large disturbances, where different physical (such as settling) and biological phenomena are taking place. It is especially difficult to measure many parameters of WWTP online. Although wastewater quality parameters can be measured by laboratory analysis, a significant time delay, which may range from a matter of minutes to a few days, is usually unavoidable. This limits the effectiveness of operation of effluent quality. Thus, a water quality prediction model is highly desirable for wastewater treatment.

The data used here is from the water quality testing daily sheet of a small-sized wastewater treatment plant in Beijing, China. The data has 867 sets and it includes information on influent BOD, influent SS, influent NH4-N, influent TP, PH. Only the five mentioned here were used to predict the effluent COD and SS. The proposed network is implemented to model wastewater treatment process with the inputs being the value of the above specified five variables, and the outputs being the effluent COD and SS. The challenges of this experiment are twofold. Frist, instead of predicting one parameter, this is a multi-parameter problem, and each parameter may have intern affection on each other. Second, the data collected is from a small-sized plant, it is not many, and may contain large amount noise. Results are shown in Figure 8 and Table 5.

Figure 8 (a) and (b) are testing results on predicting COD and SS, (c) is the trend of mean squared error in training, and (d) shows the regression. Table 5 records testing RMSE on COD and SS using different dropout rates. This experiment proves that cDBN is capable of predicting correctly on a multi-parameter problem. Normally it can convergent in 200 supervised learning steps with high regression rate. This experiment also proves that network structure and dropout rate still have impressive affects on the test RMSE. Low testing RMSE on COD can be observed where structure is large and dropout rate is small, while low SS testing RMSE can be find where structure is not too large and dropout rate not too big. This is probably because the dataset used is small, and predicting COD is much more complicated than SS, so COD needs a larger structure and more hidden neurons to learn the input features.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 October 2018 doi:10.20944/preprints201810.0494.v1



(a) Testing COD (b) Testing SS

(c) Training DBN (d) Regreesion

Fig 8 Experimental result of Waste Water Parameter Prediction

Table 5 Testing RMSE using different dropout rates

Structures	Dropout	Test RMSE/COD	Test RMSE/SS
6-6	0	6.917	6.353
	0.2	7.645	5.561
	0.5	7.911	4.995
	0.8	7.274	6.507
10-10	0	7.818	4.438
	0.2	**6.402**	**3.960**
	0.5	6.605	6.695
	0.8	8.329	3.719
20-20	0	5.403	5.100
	0.2	6.372	6.614
	0.5	8.213	8.341
	0.8	8.159	5.153

7 Conclusion

cDBN is a kind of deep learning neural network that has transferred values in hidden and visible layers from discrete to continuous. The main work of this study is: 1) A Continuous DBN with two hidden layers is proposed, in which an improved dropout strategy for unsupervised learning is

 doi:10.20944/preprints201810.0494.v1

implemented. Besides, Contrastive Divergence is introduced to raise the convergence speed. 2) A stability theorem is brought up and proved. By analysing network's convergence and stability, the hyper-parameters are given. 3) The experiments of Lorenz time series forecasting, CATS benchmark prediction, CO^2 forecasting and wastewater parameter prediction are implemented and have verified that the proposed network is capable of learning information features wisely and has good generalization ability. Besides, results show simplified structure, high accuracy and effective convergence. The experiments have also implied that cDBN can not only be used in benchmark tasks, but also can solve real world applications in industries.

Future works will focus on the two points. 1) In the aspect of algorithm, more useful structure self-organizing methods could be studied and improved in cDBN to attain better approximate ability. By increasing and omitting nodes more intelligently, cDBN could reach an improved state for one task. 2) In the aspect of real-world applications, the effectiveness of cDBN in other fields could be studied, for instance in intelligent control and regression problems.

Acknowledgements and Funding

This work is supported by the National Science Foundation of China under Grants 51711012 and 51477011, and Beijing Municipal Education Commission under Grants KM201811232016.

Author Contributions: methodology, Q.C, G.P.; software, G.P.; validation, Q.C; writing—original draft preparation, G.P; writing—review and editing, M.Y.; supervision, J.W.; project administration, J.W; funding acquisition,Q.C.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] Hinton, G., Salakhutdinov, R.: 'Reducing the dimensionality of data with neural networks', Science, 2006, 313, (5786), pp. 504-507

[2] Hinton, G., Osindero, S., The, Y.: 'A fast learning algorithm for deep belief nets', Neural Computation, 2006, 18, pp. 1527-1554

[3] Rumelhart, D., Hinton, G., Williams, R.: 'Learning representation by back-propagating errors', Nature, 1986, 232, pp. 533-536

[4] Bengio, Y., Olivier, D.: 'On the expressive power of deep architectures', Algorithmic Learning Theory, Springer Berlin Heidelberg, 2011, 18-36

[5] Bengio, Y., Pascal, L., Dan, P., Hugo, L.: 'Greedy Layer-Wise Training of Deep Networks', Advances in Neural Information Processing Systems 19 (NIPS 2006), MIT Press, 2007, pp. 153-160

[6] Bengio, Y.: 'Learning Deep Architectures for AI', Foundations & Trends in Machine Learning, 2009, 2, (1), pp. 1-127

[7] Deselaers, T., Hasan, S., Bender, O.: 'A deep learning approach to machine transliteration', Proc of the 4th Workshop on Statistical Machine Translation, 2009, pp. 233-241

[8] Dahl, G., Dong, Y., Deng, L.: 'Large vocabulary continuous speech recognition with context-dependent DBN-HMMS', Proc of IEEE International Conference on Acoustics, Speech and Signal Processing, 2011, pp. 4688-4691

[9] Itamar, A., Derek, C., Rose, T.: 'Deep Machine Learning-A New Frontier in Artificial Intelligence Research', IEEE Computational Intelligence Magazine, 2010, 11: 13-18

[10] LeRoux, N., Bengio, Y.: 'Representational power of restricted boltzmann machines and deep belief networks', Neural Computation, 2008, 20, (6), pp. 1631-1649

[11] Taylor, G., Hinton, G., Roweis, S.: 'Modeling Human Motion Using Binary Latent Variables', Advances in Neural Information Processing Systems 19, Proceedings of the 2006 Conference, 2006, pp. 1345-1352

[12] Mohamed, A., Dahl, G., Hinton, G.: 'Acoustic modeling using deep belief networks', IEEE Transactions on Audio, Speech and Language Processing, 2012, 20, (1), pp. 14-22

[13] Jurgen, S.: 'Deep Learning in Neural Networks: An Overview', Technical Report IDSIA-03-14, 2014, pp. 1-88

[14] Langkvist, M., Karlsso, L., Loutfi, A.: 'A review of unsupervised feature learning and deep learning for time-series modeling', Pattern Recognition Letters, 2014, 42, (6), pp. 11-24

[15] Hinton, G., Osindero, S., Welling, M., et al.: 'Unsupervised Discovery of Nonlinear Structure Using Contrastive Backpropagation', Cognitive Science, 2006, 30, (4), pp. 725-731

[16] Chen, H., Murray, A.: 'A continuous restricted boltzmann machine with an implementable training algorithm', IEEE Proc Vision Image Signal Process, 2003, 3, (150), pp. 153-158

[17] Ren, Z., Shen, F., Zhao, J.: 'A Model with Fuzzy Granulation and Deep Belief Networks for Exchange Rate Forecasting', IJCNN2014, Beijing, 2014, pp. 366-373

[18] Hinton, G., Srivastava, N., Krizhevsky, A.: 'Improving neural networks by preventing co-adaptation of feature detectors', arXiv preprint arXiv:1207.0580, 2012

[19] Längkvist, M., Karlsson, L., Loutfi, A.: 'A review of unsupervised feature learning and deep learning for time-series modeling', Pattern Recognition Letters, 2014. 42, (6), pp. 11-24

[20] Kuremoto, T., Kimura, S., Kobayashi, K.: 'Time series forecasting using a deep belief network with restricted Boltzmann machines', Neurocomputing, 2014, 137, pp. 47-56

[21] Bergstra, J., Bardenet, R., Bengio, Y., et al.: 'Algorithms for hyper-parameter optimization', Nips, 2011, 24, pp. 2546-2554

[22] Bergstra, J., Bengio, Y., Bottou, L.: 'Random search for hyper-parameter optimization', Journal of Machine Learning Research, 2012, 13, (1), pp. 281-305

[23] Chao, J., Shen, F., Zhao, J.: 'Forecasting Exchange Rate with Deep Belief Networks', IJCNN11, 2011, PP. 1259-1266

[24] Chen, J., Jin, Q., Chao, J.: 'Design of Deep Belief Networks for Short-Term Prediction of Drought Index Using Data in the Huaihe River Basin', Mathematical Problems in Engineering, 2012, PP. 1-16

[25] Hinton, G.: 'Training Products of Experts by Minimizing Contrastive Divergence', Neural Computation, 2002, 14, pp. 1771-1800

[26] Zhang, S., Bao, Y., Zhou, P., et al.: 'Improving deep neural networks for LVCSR using dropout and shrinking structure', 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, PP. 6849-6853

[27] Chen, Q., Chai, W., Qiao, J.: 'A Stable Online Self-Constructing Recurrent Neural Network', Lecture Notes in Computer Science, 2011, 6677, pp. 122-131

[28] Li, C., Pin, C., Chang, F.: 'Reinforced Two-Step-Ahead Weight Adjustment Technique for Online Training of Recurrent Neural Networks', IEEE Transactions on Neural Networks and Learning Systems, 2012, 23, pp. 1269-1278

[29] Amaury, L., Erkki, Q., Olli, S.: 'Time Series Prediction Competition: The CATS Benchmark', IJCNN04, Budapest, 2004, pp. 1615-1620

[30] Sarkka, S., Vehtari, A., Lampinen, J.: 'Time Series Prediction by Kalman Smoother with Cross Validated Noise Density', IJCNN04, Budapest, 2004, 2, pp. 1653-1657

643 [31] Wichard, J., Ogorzalek, M.: 'Time Series Prediction with Ensemble Models', IJCNN04, Budapest,
644 2004, pp. 1625-1629
645 [32] CO2Now.org. (2015). [Online]. Available: http://co2now.org/
646 [33] Chang X, Yu Y L, Yang Y. Semantic Pooling for Complex Event Analysis in Untrimmed Videos.
647 IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, 39(8):1617-1632.
648 [34] Mnih V, Badia A P, Mirza M. Asynchronous Methods for Deep Reinforcement Learning.
649 Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 2016.
650 [35] Qiao J, Pan G, Han H. A regularization-reinforced DBN for digital recognition. Natural
651 Computing, 2017, 2:1-13.
652 [36] Pan G, Fu L, Thakali L. Development of a global road safety performance function using deep
653 neural networks. International Journal of Transportation Science & Technology, 2017, 6(3), 159-173.
654 [37] G Pan, L Fu, L Thakali, M Muresan, M Yu. An Improved Deep Belief Network Model for Road
655 Safety Analyses. 2017, TRB 2018 (Accepted).