Deep Belief Net-Based Fault Diagnosis of Flight Control System Sensors

Yingnan Guo*, Wenhui Ning and Fangyi Wan
Department of Flight Vehicle Design Engineering, Northwestern Polytechnical University, School of Aeronautics, room C310, Aeronautical Building, Youyi West Road 127#, Xi’an, China
Email: gyn@nwpu.edu.cn

Abstract. The present study implements a model based on the deep belief net (DBN) into the sensor fault diagnosis of the flight control system. The principle of DBN system identification was adopted to simulate and establish a nonlinear observer of the unmanned aerial vehicle (UAV) for the online estimation of sensors’ output, which identified the fault type by analyzing the residuals of estimated data and the actual output. After the fault detection is completed, the measured value of the faulty sensor is isolated and replaced by the observer-generated value, in order to ensure the UAV normal flight. The proposed DBN model uses the data of normal flight control system sensors as training samples for offline training. A flight control digital simulation system was established to select the optimal DBN model via comparative test runs. Eventually, the common faults of sensors were analyzed and diagnosed online. The results obtained strongly indicate that the proposed method ensures a rapid and accurate diagnostics and isolation of faults, as well as provides a proper signal reconstruction.

1. Introduction
With the rapid development of computer technologies, deep learning has become a popular research direction of machine learning and gained a wide range of applications, including the handwritten numeral recognition, data dimensionality reduction, speech recognition, image understanding, machine translation, prediction of protein structures, and emotion recognition [1-5]. Another promising field of the deep learning application is flight control, in particular, that of unmanned aerial vehicles (UAVs). Their increasingly challenging missions require more and more sophisticated flight control systems, whose failure probability increases accordingly, and their fault detection becomes very topical. Insofar as sensor faults account for about 70% of all faults in UAV flight control systems, the primary task is in the fault analysis of a flight control system becomes its sensor fault diagnosis. Unfortunately, most conventional fault diagnosis methods either necessitate the construction of an accurate mathematical model of the system [6] or place specific stringent requirements for the collected data [7]. The online diagnosis function was successfully introduced in the sensor fault diagnosis method based on the neural network observer [8]. However, it assigned separate observers for each output parameter, which made the problem solution too sophisticated. Furthermore, each observer form implied multiple inputs and single outputs with low data utilization rate.

In this respect, deep learning application seems to be more lucrative due to its excellent feature-learning ability. The principle of deep belief net identification is applied to the designed multi-input and multi-output observer, in order to estimate several outputs. This peculiarity reduces model complexity and improves the data utilization rate. The fault can be detected and classified by the
residual analysis of observed results and actual outputs, with the further signal reconstruction. This study uses the UAV flight control system sensor as a research object, applies to it the proposed innovative DBN-based fault diagnosis system, as well as conducts numerical simulations to verify the feasibility of the developed fault diagnosis system performance.

2. The Principle of Deep Belief Net Observer

Deep belief net (DBN) is a generic model of deep learning introduced in 2006 by Hinton et al. [1, 2, 9], which is aimed to train a hierarchy of features one level at a time in iteration process called “unsupervised pre-training”. At each iteration, one layer of weights and biases is added, while a set of layers is constructed after multiple iterations. The pre-trained system can be used to build a classifier. To form the deep architecture, multiple layers should be appropriately stacked, e.g., by stacking pre-trained restricted Boltzmann machines (RBMs) into a deep belief network (DBN), stacking autoencoders (or RBMs) into a deep autoencoder, or constructing a free energy function iteratively. The first option, which is known to have strong universality and wide application range, is used in this study.

RBM is a typical neural network, which allows one to connect nodes between visible and hidden layers, whereas Hidden units can obtain the high-level correlation of the visible units. The RBM structure is depicted in figure 1. Assume that there are \( m \) nodes in the visible layer, where the input of the \( j \)th node is \( v_j \), and there are \( n \) nodes in the hidden layer, where the output of the \( i \)th node is \( h_i \). Consider the visible layer state vector \( v = (v_1, v_2, \ldots, v_m)^T \) and the hidden layer state vector \( h = (h_1, h_2, \ldots, h_n)^T \).

The energy function of the standard RBM is defined as:

\[
e(v, h | \theta) = -\sum_{j=1}^{m} \sum_{i=1}^{n} w_{ij} v_i h_j - \sum_{j=1}^{n} a_j v_j - \sum_{i=1}^{n} b_i h_i
\]

where \( \theta = \{w, a, b : 1 \leq i \leq n, 1 \leq j \leq m\} \); \( w_{ij} \) is the weight between visible and hidden nodes \( j \) and \( i \), respectively; \( a_i \) and \( b_i \) are the biases of the visible node \( j \) and hidden node \( i \), respectively. The smaller is the energy function, the more stable is the RBM system. When the network energy reaches its minimum through training, the parameters of the network are optimal.

The result of stacking a series of RBMS is a DBN. Figure 2 depicts a typical DBN with three hidden layers. The number of nodes in the visible and outer layers are determined by the vector dimension of the training data set and by the number of parameters to be estimated, respectively. However, insofar as there are no definite rules for selecting the number of nodes in each hidden layer, the trial-and-error approach is mainly used.

![Figure 1. The RBM structure.](image1)

![Figure 2. The structure of DBN with three hidden layers.](image2)

The DBN observer is a tool based on the deep belief net, which exploits the principle of system identification to estimate the pending output parameters of the current instant by observing the flight state and input instructions of the previous instant for the system [10].
Assume that the vector of deviation angle, throttle opening and other signals derived via the command received from the flight control system is \( U = [u_1, u_2, \ldots, u_n] \), the flight status data are expressed by \( X = [x_1, x_2, \ldots, x_n] \), while the system output is \( Y = [y_1, y_2, \ldots, y_n] \). Then, the discrete mathematical model of a dynamic nonlinear flight control system takes the following form:

\[
X_{k+1} = f(X_k, U_k) 
\]

\[
Y = CX \tag{2} 
\]

The flight status data \( x_1, x_2, \ldots, x_n \) can be measured by sensors and directly converted to \( y_1, y_2, \ldots, y_n \), respectively. Suppose that the parameter measured by the sensor \( y_i \) to be diagnosed is \( x_i \), while other flight status data \( x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n \) can be converted to \( y_1, y_2, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n \), which conversion can be represented by vector \( Y \). Therefore, according to the output parameters of the sensors to be detected, the obtained dependence can be summarized as:

\[
(x_{i-1}, x_{i+1}, \ldots) = f_i (x_i, x_{i-1}, x_{i+1}, \ldots, u_k) + f_i (u_{i-1}, u_{i+1}, \ldots, u_k) \tag{4} 
\]

The above dependence depicts the basic principle of DBN observer. The right and left parts of equation (4) correspond to the DBN observer input and output, respectively. The previous flight data, e.g., acquired via a quick access recorder (QAR), are processed as the training dataset of the observer model, and the above relationship is fitted with the feature identification ability of DBN. After the observer is trained, the single data of deviation angle and accelerator at the previous moment of UAV are input into the observer, and the output is the predicted output of the sensor to be diagnosed at the current instant. The residual error is assessed by comparing the predicted output with the actual one. The residual error analysis will reveal whether the sensor failed or not and, in the former case, identify the particular fault type [11]. The flowchart of the sensor fault diagnosis for a flight control system based on the DBN observer is depicted in figure 3.

### 3. The Structure and Training of Observer

For the DBN observer described above, the unsupervised pre-training [12] combined with the backpropagation algorithm [13] based on the conjugate gradient algorithm was selected to fine-tune the parameters in the training process, as well as identify the nonlinear control model and dynamic model of the particular UAV. After training, the observer is added to the whole simulation system to realize the on-line fault diagnosis and signal reconstruction. In the design of UAV control ratio, the symmetry postulate is usually met, so that the longitudinal and lateral motions of UAV can be decoupled. For brevity sake, only the longitudinal motion of UAV was investigated in the further test simulations. The overall frame diagram of UAV longitudinal control system is shown in figure 4.

**Figure 3.** Sensor online diagnosis flowchart.

**Figure 4.** Overall frame diagram of UAV longitudinal control system.
3.1. The Principle of Dbn Observer

The state output of a longitudinal system includes such data as speed, posture, altitude, and position, which are related to longitudinal control of the altitude \( h \), rate of change in altitude \( \dot{h} \), angle of pitch \( \theta \), and angular rate of pitching \( \dot{\theta} \). The input is the deviation angle \( \delta \) and throttle opening signal \( \delta_t \), while the output parameters of the sensors to be diagnosed are the angle of attack \( \alpha \), angle of pitch \( \theta \), and angular rate of pitching \( \dot{\theta} \). Thus, the observer principle can be described as:

\[
(a_{i\ldots j}, \theta_{i\ldots j}, \dot{\theta}_{i\ldots j}) = f_s(h_{i\ldots j}, \dot{h}_{i\ldots j}, \theta_{i\ldots j}) + f_x(\delta_{i\ldots j}, \dot{\delta}_{i\ldots j})
\]

According to the above principle, the observer was established and trained using the previous normal flight status data. After training, the observer could output the estimated values at the instant \( k \) by inputting the flight status values corresponding to \( k \). The estimated output of the observer was compared with the actual output of the sensor to be diagnosed. If the residual error exceeded the preset threshold value, this indicated that the fault occurred, which type could be identified by the linear regression analysis, and then the feedback signal of control ratio was reconstructed to maintain the normal flight of UAV. If the residual value was less than the set threshold value, this indicated that the sensor operation was normal.

3.2. The Training of DBN

Since DBN is a probabilistic model-based neural network, the data should be normalized before training. The training of DBN mainly consists of two parts: (i) the pre-training of RBM without layer-by-layer supervision and (ii) the backpropagation algorithm, which is used to adjust the network with supervision, in order to ensure that its parameters reach their optimal values. The specific steps are as follows:

- Step 1: Train the first RBM to achieve a stable state with \( v = h^0 \).
- Step 2: Take the output of the first RBM as the visible layer input of the second RBM and train the second RBM until it reaches a stable state.
- Step 3: Repeat step 2 until the last RBM.
- Step 4: Fine-tune the parameters of each layer with the backpropagation algorithm, so that the whole network can be optimized.

During the first three steps, the pre-training of RBM mainly adopts the \( k \)-step contrast divergence (CD-\( k \)) algorithm, with \( k = 1 \). The specific training process of the CD-\( k \) algorithm is as follows.

1. Assign the training sample data to the visible layer and map it to the hidden layer. Take a set of training samples \( x = (x_1, x_2, \ldots, x_m) \) and assign it to the visible layer as the initial state. Then, upon its substitution into the activation function, the activation probability of the hidden node can be derived as follows:

\[
p(h_i^{(0)} = 1|v^{(0)}) = \text{sigm}\left(\sum_n w_{ni}v_n^{(0)} + b_i\right)
\]

2. Consider the hidden layer samples. A group of samples is extracted from the probability distribution of the hidden layer derived from the above formula as the state of the hidden layer nodes. In the probability distribution of the hidden layer, the following relationship is assumed:

\[
h_i^{(0)} = \text{pr}(h_i^{(0)} | v^{(0)})
\]

3. Refactor visible layer. According to the hidden layer state data extracted from the above formula, the node probability of the reconstructed visible layer is calculated as follows:

\[
p(v_i^{(0)} = 1|h^{(0)}) = \text{sigm}\left(\sum_n w_{ni}h_n^{(0)} + a_i\right)
\]

4. Extract visible layer samples. A group of samples is extracted from the probability distribution of the visible layer reconstructed from the above formula as the state of the visible layer nodes.
(9)

\[ v^{(0)} - p(v^{(0)} | h^{(0)}) \]

(5) Refactor the new hidden layer. Take the reconstructed visible layer as the visible layer nodes and input them into the activation function. The probability of the new hidden layer node activated is derived as follows:

\[ p(h^{(0)} = 1|v^{(0)}) = \text{sigm}\left( \sum_{i} w_{ij} v^{(0)} + b_{j} \right) \]  \hspace{1cm} (10)

(6) Update the model parameter set \( \theta \). According to the residuals of the original and reconstructed visible layers, the first calculated hidden layer, and the second reconstructed hidden layer, the update of RBM internal parameters is reduced to the following set of equations:

\[
\begin{align*}
    w &= w + \alpha \left(p(h^{(0)} = 1|v^{(0)})(v^{(0)^T} - p(h^{(0)} = 1|v^{(0)})(v^{(0)^T}) \right) \\
    a &= a + \alpha \left(p(h^{(0)} = 1|v^{(0)})(v^{(0)} - v^{(0)}) \right) \\
    b &= b + \alpha \left(p(h^{(0)} = 1|v^{(0)})(v^{(0)} - p(h^{(0)} = 1|v^{(0)})) \right)
\end{align*}
\]

where \( \alpha \) is the learning rate, which value determines the step length of each adjustment. The variation range of \( \alpha \) being from 0.005 to 0.200, the value of \( \alpha = 0.1 \) was used in this study.

Repeat steps 1-6 until the reconstruction error converges or becomes less than the required value.

4. Simulation Verification
The standard historical data of UAV level flight at different altitudes were used as the training and test samples. The total time of the simulation was 65s, the sampling interval was 10ms, and the samples were normalized. The network structure included five layers: one input layer, multiple hidden layers, and a single output layer. Firstly, unsupervised pre-training was adopted to train the network. Then, the parameters were tuned up and down by the backpropagation algorithm based on the conjugate gradient algorithm.

The number of nodes in hidden layers is known to influence the performance of the DBN observer to some extent. This study compared the performance of DBN observers with different structures to find the optimal DBN structure based on the same training and test samples. The experimental results are listed in Table 1.

The number of required tuning iterations in Table 1 was determined by the convergence criterion of the structural errors of each network. The average variance and mean deviation were derived as \( \tau = \frac{1}{n} \sum_{i,j} e(i,j) \) \[ e(i,j) \] and \( \sigma = \frac{1}{n} \sum_{i,j} \| e(i,j) \| \), whereas \( e(i,j) \) was a training or test sample error sequence and \( n \) was the number of sample data in the training or test sets. According to the training and test error results in Table 1, the network with the maximal number of layers was not the optimal choice, insofar as an excessive number of layers made the training more complicated and hindered the optimal solution derivation. As the number of nodes in each layer increased, the accuracy was gradually improved. However, this process became saturated after reaching the critical number of nodes. Finally, the three-layer DBN network with a structure of 8-50-50-200-3 was selected to construct the observer model, according to the test error comparative analysis.

The common fault types of sensor in the simulation included the stuck fault (whereas the sensor output remained unchanged for a certain period), constant deviation fault, and constant gain fault. Such faults were simulated and analyzed, respectively, during the UAV climbing process. After fault incorporation, the values of angle of attack, angle of pitch, and angular rate of pitching would be erroneous; then, they would be taken as the observations of the sensor to be diagnosed. The fault was incorporated after 20s. When the fault occurred, the input values of the next moment were replaced by the output values of observers. During the simulation process, only one kind of parameter was
incorporated at a time. For the longitudinal motion case, the estimation and signal reconstruction for each parameter of the DBN observer diagnosis system are shown in figures 5-7, respectively.

The simulation results obtained strongly indicate that the DBN observer designed in this study can estimate the correct output of the sensor quickly and accurately. After the sensor malfunction, the proposed fault diagnosis system can rapidly detect the fault and replace the erroneous output of the malfunctioned sensor by the observer output. The performed signal reconstruction allows the UAV to continue its flight mission safely and reliably.

Table 1. Training and test results obtained via DBN models with different structure.

| Number of Hidden Layers | DBN Structure | Number of Tunings | Training Time (s) | Training Error | Test Error | Mean | Average Variance | Variance | Mean Deviation | Deviation |
|------------------------|---------------|-------------------|------------------|----------------|------------|------|-----------------|----------|----------------|----------|
| 2                      | 8-6-10-3      | 450               | 1924             | 1.92E-04       | 2.71E-04   | 0.0125 | 0.0167          |          |                |          |
| 8-10-20-3             |               | 600               | 2438             | 1.13E-04       | 1.66E-04   | 0.0095 | 0.0133          |          |                |          |
| 8-15-50-3             | 1000          | 3906              | 9.47E-05         | 0.0085         | 1.47E-04   | 0.0071 | 0.0115          |          |                |          |
| 8-20-80-3             | 1200          | 4460              | 8.60E-05         | 0.0071         | 1.29E-04   | 0.0069 | 0.0122          |          |                |          |
| 8-30-120-3            | 850           | 4589              | 8.25E-05         | 0.0069         | 1.36E-04   | 0.0078 | 0.0141          |          |                |          |
| 8-50-200-3            | 1100          | 8961              | 6.66E-05         | 0.0078         | 1.13E-04   | 0.0071 | 0.0125          |          |                |          |
| 3                      | 8-6-6-10-3    | 1000              | 4048             | 1.66E-04       | 2.25E-04   | 0.0115 | 0.0152          |          |                |          |
| 8-10-10-20-3          | 550           | 2112              | 1.49E-04         | 0.0114         | 2.21E-04   | 0.0114 | 0.0168          |          |                |          |
| 8-15-15-50-3          | 1200          | 7689              | 8.47E-05         | 0.0084         | 1.24E-04   | 0.0082 | 0.0125          |          |                |          |
| 8-20-20-80-3          | 1000          | 5001              | 1.12E-04         | 0.0082         | 1.57E-04   | 0.0088 | 0.0133          |          |                |          |
| 8-30-30-120-3         | 1000          | 8157              | 1.27E-04         | 0.0088         | 1.89E-04   | 0.0084 | 0.0158          |          |                |          |
| 8-50-50-200-3         | 1400          | 10478             | 8.87E-05         | 0.0084         | 1.22E-04   | 0.0071 | 0.0125          |          |                |          |

| Number of Hidden Layers | DBN Structure | Number of Tunings | Training Time (s) | Training Error | Test Error | Mean | Average Variance | Variance | Mean Deviation | Deviation |
|------------------------|---------------|-------------------|------------------|----------------|------------|------|-----------------|----------|----------------|----------|
| 4                      | 8-6-6-10-15-3 | 650               | 3161             | 3.23E-04       | 3.99E-04   | 0.0164 | 0.0314          |          |                |          |
| 8-10-10-15-20-3        | 1000          | 4059              | 0.0059           | 0.0063         | 0.0594     | 0.0059 | 0.0594          |          |                |          |
| 8-15-15-20-50-3        | 1000          | 4396              | 2.60E-04         | 3.16E-04       | 0.0201     | 0.0141 | 0.0233          |          |                |          |
| 8-20-20-30-80-3        | 1200          | 7082              | 1.57E-04         | 2.27E-04       | 0.0118     | 0.0122 | 0.0166          |          |                |          |
| 8-30-30-120-3         | 1400          | 11353             | 1.59E-04         | 2.25E-04       | 0.0128     | 0.0122 | 0.0151          |          |                |          |
| 8-50-50-200-3         | 1500          | 23335             | 2.08E-04         | 2.58E-04       | 0.0127     | 0.0127 | 0.0158          |          |                |          |

Figure 5. (a) The stuck fault at the angle of attack of 1.8318°; (b) The constant gain fault of the angle of attack (two-fold); (c) The constant deviation fault of the angle of attack (+2°).
Figure 6. (a) The stuck fault at the angle of pitch of 7.7144°; (b) The constant gain fault of the angle of attack (by 2.5 times); (c) The constant deviation fault of the angle of pitch (+3°).

Figure 7. (a) The stuck fault at the angular rate of pitching of 5°/s; (b) the constant gain fault of the angle of attack (by 3 times); (c) The constant deviation fault of the angle of attack (+4°).

5. Conclusions
In this paper, the observer designed on the deep belief net basis is adopted for the sensor fault diagnosis of the flight control system by utilizing the excellent feature extraction capability of deep learning. In the simulation, firstly, the observer was trained by unsupervised pre-training, and then parameters were tuned up and down by the backpropagation algorithm based on the conjugate gradient algorithm. After various functions were tried, a new one was presented and adopted, which improved the estimation accuracy. Finally, the trained observer could estimate the sensor output rapidly and accurately, as well as realize the fault diagnosis of sensor and feedback signal reconstruction. As compared to the conventional BP observer, the DBN one has the advantages of shorter total training time and higher estimation accuracy. Moreover, it can reduce the number of observers, which makes the fault diagnosis system more concise. The proposed method is readily implemented and meets the real-time requirements, concerning a timely warning and provision of necessary data for the reconstruction and independent maintenance of the flight control rules. It is found to be instrumental in the fault diagnosis, and its functionality will be refined in the follow-up studies.

References
[1] Hinton G E and Salakhutdinov R R 2006 Reducing the dimensionality of data with neural networks Sci. (313) 504.
[2] Hinton G E, Osindero S and Teh Y W 2006 A fast learning algorithm for deep belief nets Neural Computation (18) 1527.
[3] Seide F, Gang L and Dong Y 2012 Conversational speech transcription using context-dependent deep neural networks International Conference on Machine Learning p 437.
[4] Gao J F, He X D and Yih W T 2014 Learning continuous phrase representations for translation modeling Proc. ACL2014 p 699.

[5] Le D and Provost E M 2013 Emotion recognition from spontaneous speech using hidden Markov models with deep belief networks IEEE Automatic Speech Recognition & Understanding p 216.

[6] Takahisa K and Simon L D 2004 Evaluation of an enhanced bank of Kalman filters for in-flight aircraft engine sensor fault diagnostics Journal of Engineering for Gas Turbines and Power (127) 497.

[7] Wang H, Song Z and Ping L 2000 Improved PCA with optimized sensor locations for process monitoring and fault diagnosis IEEE Conference on Decision & Control p 4353.

[8] Zhang X M and Luo Q F 2012 Flight control system sensor online fault diagnosis based on neural network observer Computer Measurement & Control (20) 1781.

[9] LeCun Y, Bengio Y and Hinton G E 2015 Deep learning Nature (521) 436.

[10] Zheng X F, Guo C, Yao B and Feng H X 2017 Fault diagnosis method for aerial sensor based on deep learning Computer Engineering (43) 281.

[11] Hao T, Tang Y Z and Ren Y Q 2008 Application of BP neural network in sensor fault diagnosis for flight control system Computer Measurement & Control (16) 613.

[12] Carreira-Perpinan M A and Hinton G E 2005 On contrastive divergence learning Proc. ICASSP p 33.

[13] Livieris I E, Sotiropoulos D G and Pintelas P 2009 On descent spectral CG algorithms for training recurrent neural networks Panhellenic Conference on Informatics p 65.