https://doi.org/10.31436/iiumej.v26i1.2818

# AUTOENCODER ARTIFICIAL NEURAL NETWORK MODEL FOR AIR POLLUTION INDEX PREDICTION

**NOR IRWIN BASIR[1], KATHLYN KAIYUN TAN[1], DANNY HARTANTO DJARUM[1], ZAINAL AHMAD[1*], DAI-VIET N. VO[1], JIE ZHANG[2]**

*[1]School of Chemical Engineering, Universiti Sains Malaysia, Engineering Campus, Seri Ampangan, 14300, Nibong Tebal, Penang, Malaysia*

*[2]School of Engineering, Merz Court, Newcastle University, Newcastle upon Tyne NE1 7RU, United Kingdom*

**Corresponding author: chzahmad@usm.my*

*(Received: 3 April 2024; Accepted: 4 September 2024; Published online: 10 January 2025)*

***ABSTRACT:*** Air pollution, a significant global challenge driven by industrialization, urbanization, and population growth, is caused by the emission of harmful gases, particulates, and biological molecules into the atmosphere, posing serious risks to health and the environment. Key sources include power plants, industrial activities, vehicles, and residential heating. Thus, effective air quality monitoring and forecasting are crucial to mitigating the adverse impacts of pollution. This paper presents shallow and deep sparse autoencoder artificial neural network models to improve the prediction of the Air Pollution Index (API) in Perak Darul Ridzuan, Malaysia, as a case study. The results show that the deep sparse autoencoder achieves better prediction accuracy with $\boldsymbol{MSE}$ and $\boldsymbol{R^2}$ values of 0.1474 and 0.8331, respectively, compared to 0.1515 and 0.8300 for the shallow sparse autoencoder. The performance of these autoencoder models is also compared with other models, such as feedforward artificial neural networks (FANN) and principal component analysis (PCA). The findings confirm that both autoencoder models enhance API prediction accuracy, with the deep sparse autoencoder emerging as the optimal model, highlighting the potential of deep learning in improving air quality prediction.

***ABSTRAK:*** Pencemaran udara, merupakan satu cabaran global yang didorong oleh perindustrian, urbanisasi pesat, dan pertumbuhan populasi, adalah disebabkan oleh pelepasan gas, partikel, dan molekul biologi merbahaya ke atmosfera, menimbulkan risiko serius kepada kesihatan dan alam sekitar. Sumber utama termasuk loji janakuasa, aktiviti industri, kenderaan, dan pemanasan kediaman. Oleh itu pemantauan dan ramalan kualiti udara penting bagi mengurangkan kesan buruk pencemaran. Kajian ini membentangkan model rangkaian neural tiruan pengauto kod jarang 'cetek' dan pengauto kod jarang 'dalam' memperbaiki ramalan Indeks Pencemaran Udara (API) di negeri Perak Darul Ridzuan, Malaysia sebagai kes kajian. Dapatan kajian menunjukkan bahawa pengautokod jarang 'dalam' mencapai ketepatan ramalan lebih baik, dengan nilai MSE dan R2 masing-masing sebanyak 0.1474 dan 0.8331, berbanding 0.1515 dan 0.8300 bagi pengautokod jarang 'cetek'. Prestasi model pengautokod ini juga dibandingkan dengan model lain, seperti rangkaian neural tiruan suapan hadapan (FANN) dan analisis komponen utama (PCA). Hasil kajian mengesahkan bahawa kedua-dua model pengautokod meningkatkan ketepatan ramalan API, dengan pengautokod jarang 'dalam' muncul sebagai model paling optimum, menonjolkan potensi pembelajaran mendalam 'dalam' meningkatkan ramalan kualiti udara.

***KEYWORDS:*** *Air pollution index, Shallow sparse autoencoder, Deep sparse autoencoder, Prediction.*

https://doi.org/10.31436/iiumej.v26i1.2818

# 1. INTRODUCTION

Research on environmental quality assessment and prediction-related fields can be generally categorized into two main groups: deterministic methods and statistical methods [1]. Deterministic methods incorporate statistical methods and meteorological theoretical principles in representing the processes of diffusion, dispersion, elimination, emanation, and transformation of air pollutants fundamentally based on atmospheric physical and chemical reactions. This method is considered model-based since its architecture is predefined with theoretical assumptions; the output/target data can be calculated with precise knowledge of the model parameters prior. On the other hand, statistical methods utilize statistic-based techniques, such as autoregressive moving average (ARMA) [2], multiple linear regression (MLR) [3], support vector regression (SVR) [4] and artificial neural network (ANN), in predicting or forecasting air quality instead of employing complex theoretical techniques.

Artificial neural network (ANN) is a topic of great interest in the research world in its development to be used in air quality model prediction due to its ability and capability to handle a high dimensionality of real data and its self-adaptivity in performing dimensionality reduction, features representation, and relationship learning between the input data and the output/target data. In a typical ANN, the raw input data is remodeled into new interpretable data with smaller dimensions; thus, this process is called 'dimensionality reduction.' This process is essential to preserve significant information of the input data (features extraction) for further analysis of the data, whereby in this paper, it is for Air Pollution Index (API) prediction purposes. Air pollution is detrimental to health. It is caused by industrialization, rapid urbanization, and population growth. It is a common problem faced on a worldwide scale. In particular, power plant energy production, industrial processes, fuel-burning vehicles, residential heating, and natural catastrophes are the usual causes of the problem. The effects of air pollution can be generally categorized into two groups: short-term consequences and long-term consequences. Among the short-term consequences, human health-related effects are the utmost significant concerns, specifically at the metropolises; on the other hand, the long-term consequences encompass global climate-related effects such as the greenhouse effect and global warming. In conjunction with the escalation of air pollution issues, enhanced public awareness concerning air quality has resulted in both developed and developing countries.

The air quality monitoring and forecasting tools are indeed essential so that precautionary measures can be taken by minimizing the potential negative effects of predicted pollution peaks on the surrounding ecosystem and habitat. In general, the monitoring of the air quality is conducted in a manual manner continuously to detect the ambient air quality variations that may pose adverse effects to human health and the environment. The Department of Environment (DOE), Malaysia, carries out ambient air quality monitoring through a network that consists of 51 monitoring stations. All the aforementioned stations are situated strategically in residential, industrial, and heavy-traffic areas for air pollution control purposes. The Air Pollution Index (API) is used to describe and report the ambient air quality in Malaysia instead of utilizing the actual air pollutant concentration due to the simplicity in terms of presentation of the former. API not only reflects the intensity of the air pollution effects on human health, which ranges from hazardous to good, but also can be categorized in terms of the action criteria in accordance with the National Haze Action Plan. In this paper, sparse autoencoder ANN with both shallow and deep architecture models are proposed to improve API prediction performance. In this study, both proposed types of architecture are modeled using the same training data.

An autoencoder has various applications in various disciplines, as validated and proven by other researchers. An autoencoder (formerly known as an auto-associator) functions as a robust

https://doi.org/10.31436/iiumej.v26i1.2818

device with self-supervised learning [5]. It aims to transform input data to output data with a minimum degree of distortion. It is trained autoassociatively using its three main components (see Figure 1), i.e., encoder, bottleneck, and decoder [6, 7]. It learns to encode and compress the input into a latent space representation, which preserves significant information of the input to be stored in the bottleneck layer with smaller dimensions as features. It then learns to decode and reconstruct the input information at the output layer. Backpropagation is employed in the learning algorithm of an autoencoder [8, 9]. The autoencoder learning algorithm is able to generate a significant range of behaviors that are psychologically related, such as distortion, generalization, inferencing, recalling, and recognition [10]. A shallow autoencoder that employs a linear activation function is equivalent to a principal component analysis (PCA) model [11]. One of the significant differences between the two models is that an autoencoder is capable of learning both linear and nonlinear feature representation, while a traditional PCA model is capable of learning linear feature representation only. Thus, an autoencoder has become one of the highly considered artificial neural network models in constructing a statistical model based on a set of training data.

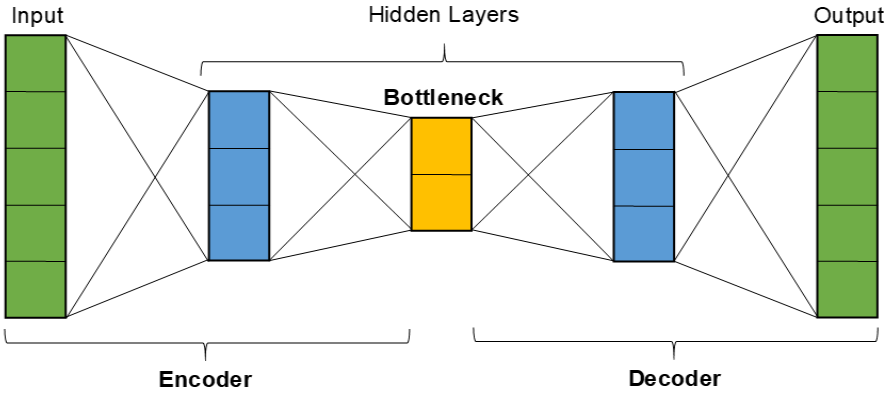



Figure 1. A typical autoencoder ANN architecture

Due to the high dimensionality of real data, a deep architecture approach is often employed in developing an autoencoder from a shallow autoencoder into a deep autoencoder. An autoencoder with one hidden layer of bottleneck is known as a shallow autoencoder. A deep autoencoder is technically an extension of a shallow autoencoder with more than a single hidden layer. Figure 1 illustrates a deep autoencoder with three hidden layers: one bottleneck hidden layer and one hidden layer each in its encoder and decoder. The additional hidden layers in the architecture of a deep autoencoder allow the ANN to learn the underlying features of the input data with higher complexity. The first hidden layer may learn the first-order features, while the second hidden layer may learn the second-order features. A deep autoencoder with a hierarchy of hidden layers tends to learn higher-order features and capture unknown data structures but takes longer training time.

Research on the application of autoencoders for prediction via relationship learning between the input data and the output/target data has been proven to be valid and successful by other researchers in a wide range of disciplines. It has been proposed for use as a structural condition monitoring tool [12]. The relationship between the modal information, such as mode shapes and frequencies, and the structural stiffness parameters is significant for structural damage detection in order to assess the safety conditions of a civil infrastructure under certain operating conditions. Excellent structural stiffness prediction was achieved using the autoencoder via a nonlinear dimensionality reduction of the modal information (input data) features followed by a nonlinear regression against the structural stiffness parameters for relationship learning purposes between the input modal information (concatenated feature

vector) and the output structural stiffness parameters. A similar approach was taken by proposing an autoencoder for use in the estimation of sea state bias (SSB) according to radar altimeter data [13]. SSB is defined as an altimeter-ranging error due to the presence of ocean waves on the surface of the sea. Relationship learning has been carried out between the input data (automatic gain control, backscatter coefficient, sea surface height, significant wave height, and wind speed) and the output data (SSB) using an autoencoder. It is presented that the proposed autoencoder model yields both higher prediction accuracy and operational efficiency compared to the conventional parametric model and nonparametric model.

Apart from that, the application of autoencoders for prediction through relationship learning in the climate-related discipline, which is closely related to our target model, has also been proven to be valid for weather forecasting [14]. Being inspired by successful applications of autoencoder for prediction purposes, it is, thus, believed that autoencoder ANN is able to solve practical nonlinear environmental modelling problems, improving API prediction performance with proper tuning and optimization as proposed in this paper. In recent years, research on air pollution modeling has been conducted with significant efforts. There are several types of ANN being applied in the development of air quality prediction models, such as feedforward artificial neural network (FANN) [15, 16], combined all multiple neural network (MNN), forward selection (FS) aggregated multiple neural network (MNN) and backward elimination (BE) aggregated multiple neural network (MNN) [16], and principal component analysis – feedforward artificial neural network (PCA-FANN) [17]. These ANN models, which have been trained, might sometimes fail to predict the target data with significant accuracy when being applied to test/unseen data due to the learning models converging to non-desirable local minima and/or overfitting of the noise present in the training data [18]. Thus, in this paper, an autoencoder with proper tuning and optimization is proposed for modeling an API predictor based on real data with the fundamental objective of enhancing the prediction accuracy and reliability of the air quality prediction model.

This study is significant and beneficial as it enhances air quality forecasting by introducing a more precise and dependable model. It can guide individuals, communities, or relevant entities interested in air quality control and forecasting. The search for enhanced prediction accuracy can aid in more effective planning and decision-making related to air quality management. This research can be a noteworthy contribution to the existing knowledge in the field of air quality forecasting using artificial intelligence methodologies.

This paper is organized as follows: the Materials and Methods section describes the case study, including the air quality data sampling location in Malaysia and the proposed shallow, sparse autoencoder and deep sparse autoencoder with the predictor. The Results and Discussions section presents the proposed autoencoder's results and discussions, and the last section concludes this paper.

## 2. MATERIALS AND METHODS

Please refer to Figure 2 for a detailed flowchart illustrating the various stages of the research process, from problem identification to conclusion. Each stage provides a comprehensive overview of the methodology employed in this study. As an overview, the research process starts with identifying the air pollution problem and understanding the need for accurate prediction models. Relevant air quality data is collected and prepared for modeling. The proposed artificial neural network models are then developed, trained, and tested to enhance prediction performance. The models are evaluated using specific metrics, and the results are analyzed to validate their effectiveness. The research concludes by summarizing the

findings and suggesting future work. This process ensures a thorough and significant contribution to air quality prediction.

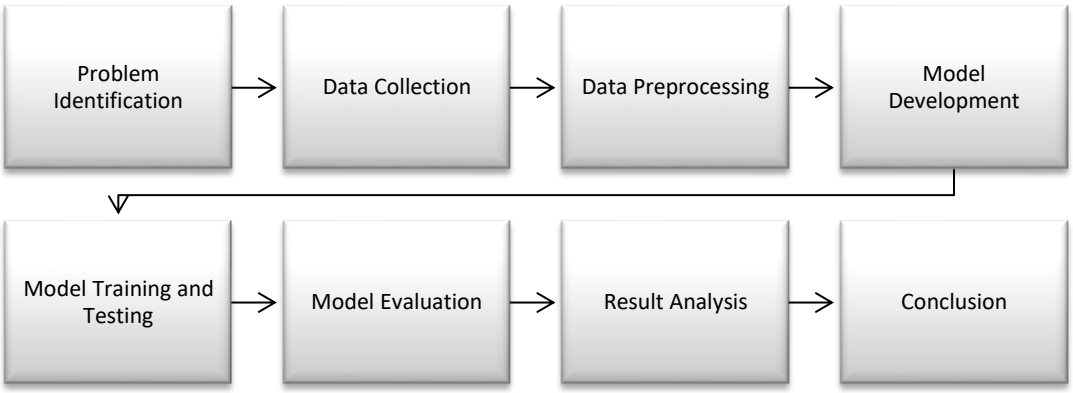



Figure 2. Overall Process Flow Diagram of the Research Processes

### 2.1. Case Study: Perak Darul Ridzuan State Air Monitoring Station, Malaysia

Most air quality data was acquired directly from the air quality monitoring stations or via the remote sensing instruments. In this study, the air quality data was collected by the Department of Environment (DOE), Malaysia, from 4 monitoring stations around Perak that are situated at CA0020, CA0041, CA0045, and CA0046, as shown in Figure 3 and described in Table 1. Reference [19] stated that the continuous air quality monitoring (CAQM) stations are strategically located in residential, industrial, and traffic areas to detect any significant changes in the air quality that may threaten the surrounding ecosystem and habitat.

In this study, the air quality data comprises data recorded for 5 years (from the 1st of January 2006 till the 31st of December 2010) with 8 input variables and 1 output/target variable. For API prediction modeling, the input variables involved include meteorological variables (air temperature, relative humidity, and wind speed) and air pollutants (carbon monoxide (CO), nitrogen dioxide ($NO_2$), ozone ($O_3$), particulate matter $PM_{10}$ and sulfur dioxide $SO_2$) concentration variables. In contrast, the output/target variable is API.

Table 1. Locations of Perak air monitoring stations [19]

| Station ID | Air Monitoring Station | Alternate Name | Latitude (N) | Longitude (E) |
|---|---|---|---|---|
| CA0020 | Sekolah Kebangsaan Ayer Puteh | Taiping | 4°89.881 | 100°67.912 |
| CA0041 | Pejabat Daerah Manjung | Manjung | 4°12.020 | 100°39.800 |
| CA0045 | Universiti Pendidikan Sultan Idris | Tanjung Malim | 3°68.758 | 101°52.438 |
| CA0046 | Sekolah Menengah Pegoh | Ipoh | 4°55.330 | 101°08.017 |

Table 2. Input and output/target variables for API prediction modeling.

| Input Variables | Output/Target Variable |
|---|---|
| Wind Speed (km/hr) | API |
| Air Temperature (°C) | |
| Relative Humidity (%) | |
| $O_3$ concentration (mg/L) | |
| CO concentration (mg/L) | |
| $PM_{10}$ concentration (µg/m$^3$) | |

https://doi.org/10.31436/iiumej.v26i1.2818

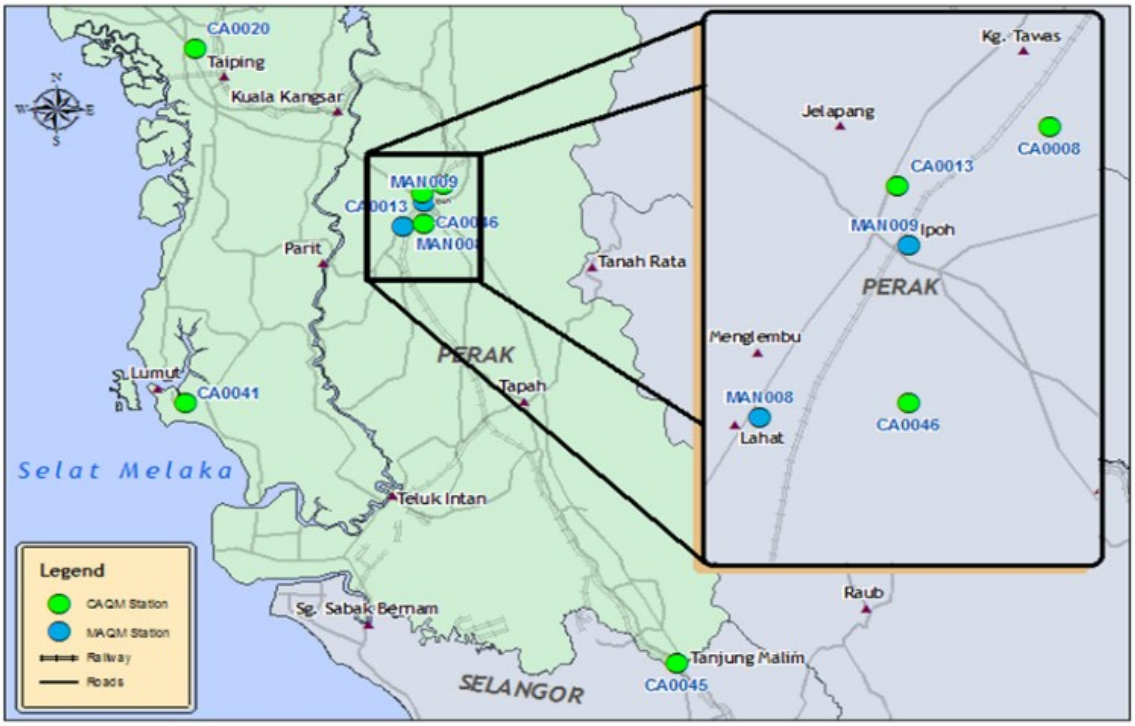



Figure 3. Perak air monitoring stations

As the $SO_2$ and NO2 concentrations displayed infinitesimal change throughout the 5 years, the involvement of both variables in API prediction modeling would yield a negligible effect on the proposed model performance. Thus, to increase the efficiency and effectiveness of the proposed model and develop a parsimonious model (a model that depends on a few input variables as necessary), the number of predictor input variables is reduced without losing important information. In this study, only 6 input variables were selected for the API prediction modeling, as summarized in Table 2. In this study, a total of 1826 samples are used for modeling and analysis, while the missing data is denoted as Not a Number (NaN).

## 2.2. Sparse Autoencoder Artificial Neural Network (ANN) with Predictor Model Development

In this case study, 1826 sampling instances were obtained from the DOE, Malaysia database from the 1st of January 2006 to the 31st of December 2010. MATLAB® was used for the API prediction modeling. All data was normalized on the same scale to zero mean ($\mu \approx 0$) and unit standard deviation ($\sigma \approx 1$) to cope with the difference in magnitude across the variables, introducing a common ground for the equal treatment of the features extracted after this during the ANN learning process. The time series data was transposed from a matrix of 1826 samples × 7 elements into a matrix form where the proposed and developed ANN model can process 7 elements × 1826 samples; it is noted that the first 6 elements represent the 6 input variables, respectively while the 7th element represents the output/target variable. Then, the scaled data was divided randomly into 2 sets of data: 85 % for training (1552 samples) and the remaining 15 % for testing (274 samples). Static time series data was applied in the API prediction modeling where the output/target data was a function of the input data, as shown in Eq. (1):

$$y_i(t) = f\left(x_{i1}(t), x_{i2}(t), \cdots, x_{in}(t)\right) \quad (1)$$

where $y_i(t)$ is the output/target data $y$ (API) at time $t$ in $i^{\text{th}}$ sample, $x_i(t)$ is the input data $x$ at time $t$ in $i^{\text{th}}$ sample where the subscripts 1 and 2 denote the first and second input variables,

https://doi.org/10.31436/iiumej.v26i1.2818

respectively, and $n$ represents the total number of model input variables. In this case study, the model input variables consist of air temperature, relative humidity, wind speed, CO concentration, $O_3$ concentration, and $PM_{10}$ concentration, making up a total number of 6 ($n$ = 6).

### 2.3. Sparse Autoencoder Artificial Neural Network (ANN)

As mentioned in Section 1, an autoencoder is a type of ANN that encodes and compresses the input into its latent space representation for reconstruction and decodes the input information as the output [20]. The most important part of the learning process is to preserve the important information from the input. In this study, a sparse autoencoder with 2 architectural models is proposed to extract significant features of the input data, which contains the data from the 6 input variables as mentioned in Table 2.

### 2.4. Shallow Sparse Autoencoder Artificial Neural Network (ANN)

A shallow autoencoder is an autoencoder with a single input layer, a single hidden layer (bottleneck), and a single output layer. The proposed shallow autoencoder architecture in this study is shown in Figure 4.

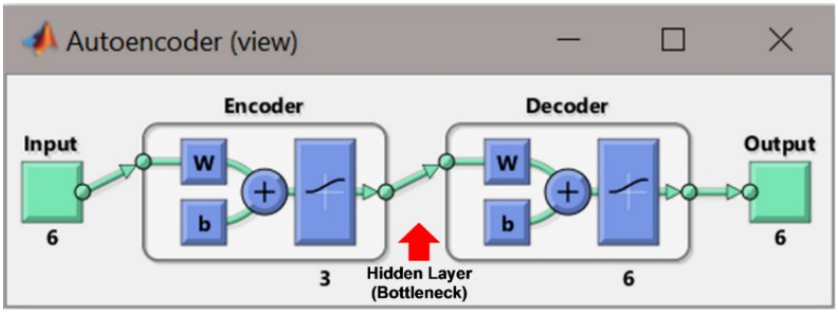



Figure 4. Shallow autoencoder architecture

An autoencoder is an ANN that implements 2 major transformations; the encoder transforms $d$ dimensional input into $r$ dimensional latent representation ($encode(x)$: $R^d \rightarrow R^r$) and the decoder transforms $r$ dimensional latent representation back into $d$ dimensional reconstructed input ($decode(h)$: $R^r \rightarrow R^d$). A set of training samples is given as $\{x_1, x_2, \cdots, x_m\}$ where, in this case, $x$ is the input data sample whereby each sample contains 6 input variable data, the subscripts 1 and 2 denote the first and second training input samples, respectively, and m represents the total number of model training input samples, whereby in this case $m = 1552$, such that $x_i \in R^d$ where the subscript $i = 1, 2, \cdots, m$. Firstly, in general, an autoencoder encodes and compresses the input vector $x$ into the hidden layer as the representation $h$ whose computational function is expressed as Equation 2; then, the representation $h$ is decoded back to the $x$ dimension as reconstruction $z$ whose computational function is described in Equation 3 as follows;

$$h = encode(x) = \Phi(W_1 x + b) \quad (2)$$

$$z = decode(h) = \Phi(W_2 h + c) \quad (3)$$

where $W_1$ is the weight matrix for the optimization process, $b$ is the encoding bias vector, $W_2$ is the weight matrix for the decoding process, $c$ is the decoding bias vector, and $\Phi$ is the activation/transfer function. In this study, $\Phi$ was set to be a logistic sigmoid function, as shown in Eq. (4), for both encode ($x$) and decode ($h$) to allow the developed autoencoder to learn nonlinear feature representation.

https://doi.org/10.31436/iiumej.v26i1.2818

$$\Phi(x) = \frac{1}{1+e^{-x}} \tag{4}$$

An autoencoder is trained by minimizing the reconstruction error, which is the difference between the original and reconstruction outputs. The autoencoder model parameters of $W_1$, $b$, $W_2$ and $c$ are optimized to minimize the average reconstruction error whose computational equation is shown in Eq. (5), where the loss function $L(x_{ij}, z_{ij})$ The traditional squared error function, shown in Eq. (6), was employed in this study. For optimization purposes, the scaled conjugate gradient descent algorithm was employed to train the autoencoder.

$$[W_1^*, b^*, W_2^*, c^*] = arg\ min_{W_1,b,W_2,c}\ \frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{n} L(x_{ij}, z_{ij}) \tag{5}$$

$$L(x_{ij}, z_{ij}) = (x_{ij} - z_{ij})^2 \tag{6}$$

An ideal autoencoder balances two main reconstruction criteria: sensitive enough to capture significant features of the input to reconstruct the encoded data as the output with high accuracy and minimum distortion and insensitive enough to prevent memorizing and overfitting of the noise present in the training data [21]. Thus, to force the autoencoder to preserve only the important and useful variations present in the input essential for reconstruction without holding onto the redundancies present in the input, the sparsity constraint method was applied, transforming the autoencoder model into a sparse autoencoder model [22]. In order to obtain sparse representation, Eq. (5) was optimized and modified by embedding a sparsity constraint into it, as expressed in Eq. (7).

$$[W_1^*, b^*, W_2^*, c^*] = arg\ min_{W_1,b,W_2,c}\left[\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{n}(x_{ij} - z_{ij})^2 + (\alpha \times \Omega_{weights}) + (\beta \times \Omega_{sparsity})\right] \tag{7}$$

$$\Omega_{weights} = \frac{1}{2}\sum_{k=1}^{p}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(W_{ij}^{(k)}\right)^2 \tag{8}$$

$$\Omega_{sparsity} = \sum_{l=1}^{r} KL(\rho \,||\, \widehat{\rho_l}) \tag{9}$$

$$KL(\rho \,||\, \widehat{\rho_l}) = \rho \log\left(\frac{\rho}{\widehat{\rho_l}}\right) + (1-\rho)\log\left(\frac{1-\rho}{1-\widehat{\rho_l}}\right) \tag{10}$$

$$\widehat{\rho_l} = \frac{1}{m}\sum_{i=1}^{m}(h_l \times x_i) \tag{11}$$

where $\alpha$ is the coefficient for the $L_2$ regularization term $\Omega_{weights}$ whose mathematical equation is expressed in Equation 8, $\beta$ is the coefficient for the sparsity regularization term $\Omega_{sparsity}$ whose mathematical equation is expressed in Eq. (9), $p$ is the number of hidden layers, $W$ is the weight matrix, $r$ is the number of hidden units, $KL$ is the Kullback-Leibler divergence between two Bernoulli random variables with the means equal to $\rho$ and $\widehat{\rho_l}$, respectively as defined in Eq. (10), $\rho$ is the desired sparsity parameter whose value defines the desired proportion of training samples a hidden unit reacts to, and $\widehat{\rho_l}$ is the average output activation value of the hidden unit l over the training data set, as expressed in Eq. (11).

The sparsity enforce constraint, as expressed mathematically in Eq. (12), was applied. Typically, $\rho$ is a small value close to 0. In this case, with shallow architecture, $\rho$ was set to 0.05 for the logistic sigmoid activation/transfer function. The setting enforces $\widehat{\rho_l}$ to be as close to 0.05 as possible, leading to the activation value of the hidden unit mostly nearing 0. Kullback-Leibler divergence was employed as it fastens the sparsity constraint during the coding process, penalizing $\widehat{\rho_l}$ from diverging away from $\rho$ significantly $KL(\rho \,||\, \widehat{\rho_l}) \approx 0$ if $\widehat{\rho_l} \approx \rho$.

$$\widehat{\rho_l} \approx \rho \tag{12}$$

https://doi.org/10.31436/iiumej.v26i1.2818

The list of significant shallow, sparse autoencoder ANN training parameters for the API prediction model is shown in Table 3. The features captured in the bottleneck hidden layer of the proposed and developed shallow sparse autoencoder, which contains significant representative information of the input data, were then utilized for successive API prediction via the relationship learning process.

Table 3. Shallow sparse autoencoder ANN training parameters for API prediction model

| Training Parameter | Description/Value (MATLAB® Code) |
|---|---|
| Number of hidden nodes (bottleneck hidden layer) | 3 |
| Activation/Transfer function for encoder | Logistic sigmoid (logsig) |
| Activation/Transfer function for decoder | Logistic sigmoid (logsig) |
| Maximum number of training epochs/iterations | 2000 |
| $L_2$ weight regularizer coefficient, $\boldsymbol{\alpha}$ | 0.001 |
| Desired sparsity parameter, $\boldsymbol{\rho}$ | 0.05 |
| Sparsity regularizer coefficient, $\boldsymbol{\beta}$ | 1 |
| Training algorithm | Scaled conjugate gradient descent (trainscg) |

## 2.5. Deep Sparse Autoencoder Artificial Neural Network (ANN)

A deep autoencoder is technically the extension of a shallow autoencoder with a single input layer, more than one hidden layer, and a single output layer. The proposed deep autoencoder architecture in this study is shown in Figure 5. It is noted that Figure 5 depicts the encoding architecture of the proposed deep autoencoder but not the generic structure of the whole deep autoencoder architecture, which includes the decoding part.

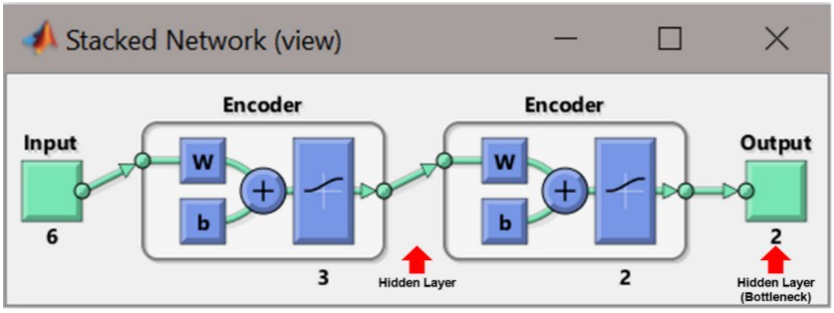



Figure 5. Deep autoencoder encoding architecture

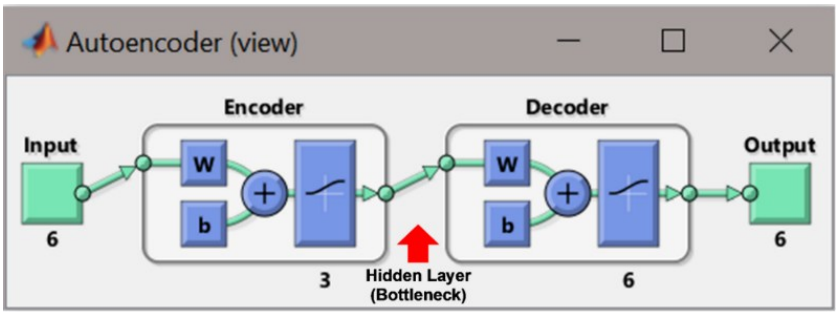



Figure 6. First shallow autoencoder architecture (encoding hidden layer of deep autoencoder)

During the construction of the proposed deep autoencoder for this study, 2 shallow sparse autoencoder models were first developed, each using similar computational approaches and functions as described in Section 2.4. In order to train a deep autoencoder effectively, pretraining of one of the hidden layers at a time sequentially (layer-wise training) was

https://doi.org/10.31436/iiumej.v26i1.2818

performed. The first shallow sparse autoencoder model architecture with the input data x as the model input is shown in Figure 6, and the second shallow sparse autoencoder model architecture with the features extracted from the first shallow sparse autoencoder h as the model input is shown in Figure 7. Then, these developed shallow sparse autoencoder models were stacked together to create a deep architecture, as illustrated in Figure 5, forming a deep sparse autoencoder.

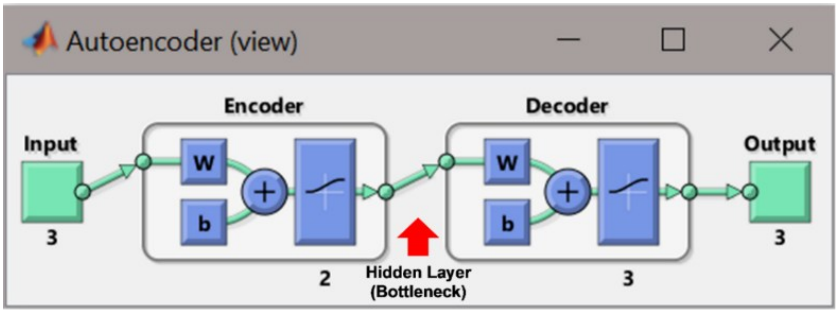



Figure 7. Second shallow autoencoder architecture (bottleneck hidden layer of deep autoencoder)

Table 4. First and second shallow sparse autoencoder ANN training parameters for deep architectural model

| **First shallow autoencoder (encoding hidden layer of deep autoencoder)** | |
|---|---|
| **Training Parameter** | **Description/Value (MATLAB® Code)** |
| Number of hidden nodes (bottleneck hidden layer) | 3 |
| Activation/Transfer function for encoder | Logistic sigmoid (logsig) |
| Activation/Transfer function for decoder | Logistic sigmoid (logsig) |
| Maximum number of training epochs/iterations | 2000 |
| $L_2$ weight regularizer coefficient, $\boldsymbol{\alpha}$ | 0.001 |
| Desired sparsity parameter, $\boldsymbol{\rho}$ | 0.05 |
| Sparsity regularizer coefficient, $\boldsymbol{\beta}$ | 1 |
| Training algorithm | Scaled conjugate gradient descent (trainscg) |
| **Second shallow autoencoder (bottleneck hidden layer of deep autoencoder)** | |
| **Training Parameter** | **Description/Value (MATLAB® Code)** |
| Number of hidden nodes (bottleneck hidden layer) | 2 |
| Activation/Transfer function for encoder | Logistic sigmoid (logsig) |
| Activation/Transfer function for decoder | Logistic sigmoid (logsig) |
| Maximum number of training epochs/iterations | 1000 |
| $L_2$ weight regularizer coefficient, $\boldsymbol{\alpha}$ | 0.00001 |
| Desired sparsity parameter, $\boldsymbol{\rho}$ | 0.025 |
| Sparsity regularizer coefficient, $\boldsymbol{\beta}$ | 0.9 |
| Training algorithm | Scaled conjugate gradient descent (trainscg) |

In the proposed and developed deep sparse autoencoder architecture, as illustrated in Figure 5, the first hidden layer is the bottleneck hidden layer indicated by the middle arrow in Figure 6 (first shallow sparse autoencoder), while the second hidden layer is the bottleneck hidden layer indicated by the middle arrow in Figure 7 (second shallow sparse autoencoder). In the deep sparse autoencoder architecture, as shown in Figure 5, the first hidden layer (encoding hidden layer) performs a feature fusion process on the 6 input variables of air temperature, relative humidity, wind speed, CO concentration, $O_3$ concentration, and $PM_{10}$ concentration via nonlinear dimensionality reduction, while the second hidden layer (bottleneck hidden layer) performs further feature extraction on the low dimensional features representation learned in the first hidden layer. Stated generically, for a deep autoencoder with

$p$ hidden layers and $k = (1, 2, \cdots, p)^{th}$ hidden layer, training of the first ($k = 1^{st}$) hidden layer is conducted with the training data set as its input; then, training of the successive $(k + 1)^{th}$ hidden layer is carried out with the output of the $k^{th}$hidden layer as its input. Sequential autoencoder models are stacked hierarchically depending on the depth (number of hidden layers) of the desired deep autoencoder.

The list of significant first and second shallow sparse autoencoder ANN training parameters for deep sparse autoencoder model development is shown in Table 4. It is noted that the value for each training parameter of the second shallow autoencoder is relatively smaller than the one listed under the first shallow autoencoder; this is because the dimensionality of the model input whose features representation is to be learned by the second shallow autoencoder is relatively lower compared to the first shallow autoencoder. The bottleneck hidden layer of the proposed and developed deep sparse autoencoder contains more feature abstraction than the encoding hidden layer. The features captured in the bottleneck hidden layer that contain significant representative information of the input data were then utilized for successive API prediction via the relationship learning process.

### 2.6. Relationship Learning

The primary objective of this process is to learn the relationship between the feature extracted h by the sparse autoencoder ANN models that have been developed and the output/target API values. This study embedded a 2 layered shallow feedforward neural network model into the developed sparse autoencoder for API prediction purposes via supervised relationship learning. A shallow feedforward neural network was used as the predictor due to its robustness in relationship learning to fit practical functions and its simplicity in utilization and architectural development. The proposed model consists of 3 layers: an input layer (features extracted h by sparse autoencoder ANN), a hidden layer, and an output layer (predicted API), as illustrated in Figure 8.

$$a = \frac{2}{1+e^{-2h}} - 1 \tag{13}$$

$$\hat{y} = a \tag{14}$$

The feedforward neural network predictor was pretrained with a hyperbolic tangent sigmoid activation/transfer function, as mathematically expressed in Equation 13, to learn the nonlinear relationship and linear activation/transfer function as mathematically expressed in Equation 14 in the output layer where a is the output of hidden layer and $\hat{y}$ is the network's predicted data (predicted API). The network was trained using a scaled conjugate gradient backpropagation training function.

The list of significant feedforward neural network training parameters for API prediction is shown in Table 5. It is noted that the sparsity constraint was enforced in the dimensionality reduction (features extraction) component (autoencoder) only and not in the relationship learning component (feedforward neural network) due to its proven effectiveness in dimensionality reduction and its inability and poor performance in learning efficient mapping [23]. The feedforward neural network was adapted with weight and bias learning rules.

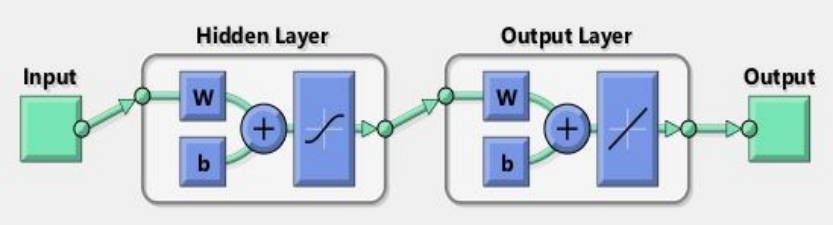



Figure 8. Shallow feedforward neural network architecture (relationship learning)

Table 5. Shallow feedforward neural network training parameters for API prediction model

| Training Parameter | Description/Value (MATLAB® Code) |
|---|---|
| Number of hidden nodes (hidden layer) | 11 |
| Activation/Transfer function (hidden layer) | Hyperbolic tangent sigmoid (tansig) |
| Activation/Transfer function (output layer) | Linear (purelin) |
| Maximum number of training epochs/iterations | 2000 |
| Training function | Scaled conjugate gradient backpropagation (trainscg) |

## 2.7. Fine Tuning of Sparse Autoencoder Artificial Neural Network (ANN) with Predictor Model

In this study, the dimensionality reduction (features extraction) component (sparse autoencoder) and the relationship learning component (feedforward neural network) were combined as a deep ANN, as shown in Figure 9, for API prediction. All of the constructed layers were pretrained in a layer-wise manner with a scaled conjugate gradient descent backpropagation algorithm being employed throughout the API prediction ANN model for optimization of each layer purpose. However, it is studied that the application of the aforementioned approach in training deep ANN may yield relatively poor performance [24-26]. This is because a deep ANN with small initial weights tends to generate tiny gradients at the bottom layers, reducing the applicability of the training ANN with numerous hidden layers; on the other hand, a deep ANN with large initial weights tends to generate relatively poor local minima [27]. Due to the complexity of the algorithm underlying the deep ANN architecture in nature, it is often challenging to preserve the relevancy between the parameters across the layers. To tackle this issue, a fine-tuning procedure was employed on the developed full sparse autoencoder ANN-based API prediction model for final joint optimization to further train the deep ANN more effectively with the fundamental goal of minimizing the prediction error [28, 29].

https://doi.org/10.31436/iiumej.v26i1.2818

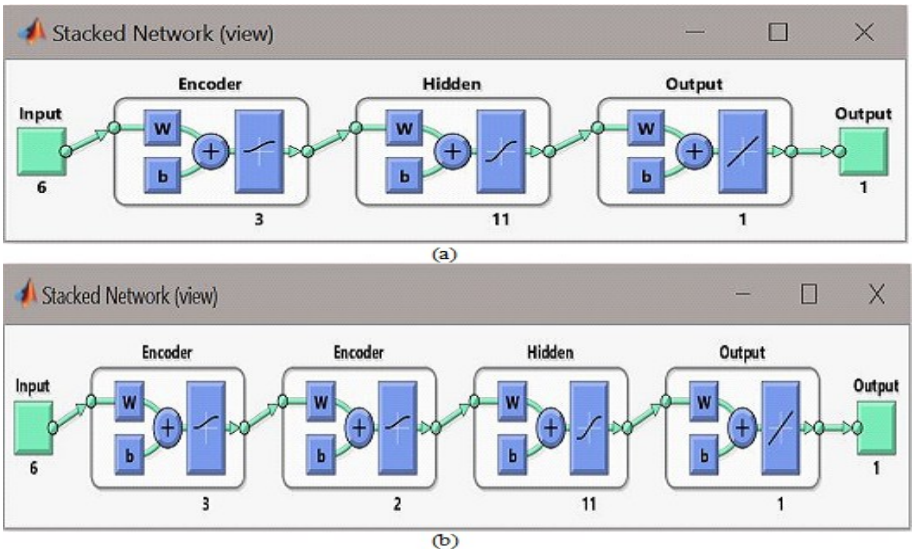



Figure 9. API prediction model architecture with: (a) shallow sparse autoencoder, (b) deep sparse autoencoder

The principal idea of this technique consists of 2 stages; the first stage is to conduct pretraining of the full deep ANN layer-wise in a bottom-up manner, and the subsequent stage is to carry out fine-tuning (updating) of the entire deep ANN parameters in a top-down fashion via a backpropagation algorithm whereby, in this study, it is the scaled conjugate gradient descent backpropagation algorithm. During the second stage, the developed deep ANN was fine-tuned by retraining the entire architecture on the training data in a supervised manner, adjusting the weights of the trained model from the final layer to optimize all the constructed layers.

### 2.8. Model Performance Evaluation

The performance of the proposed and developed sparse autoencoder ANN-based API prediction model was evaluated in terms of mean squared error $MSE$ and coefficient of determination $R^2$ between observed true API $y$ and predicted API $\hat{y}$. $MSE$ and $R^2$ are mathematically expressed as Equation 15 and Equation 16, respectively.

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{15}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{m}\hat{y}_i^2 - \frac{\sum_{i=1}^{m}\hat{y}_i^2}{m}} \tag{16}$$

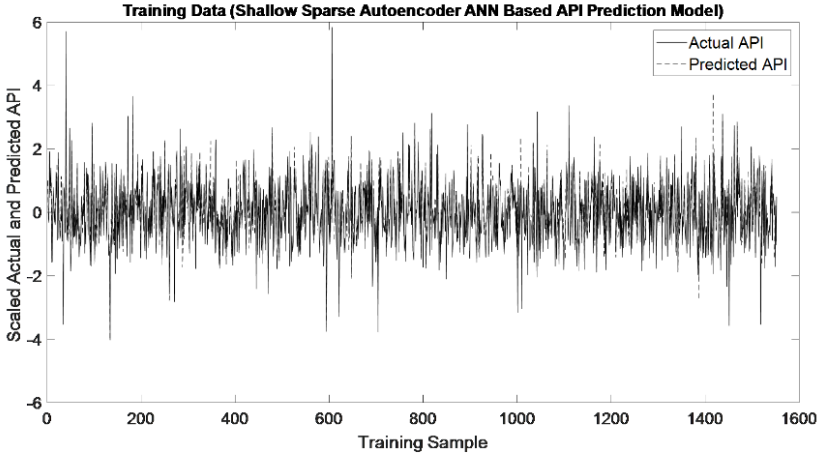
where $m$ is the total number of evaluation samples, $y_i$ is the observed true target variable data (API) of the $i^{th}$ sample, and $\hat{y}_i$ is the predicted target variable data (API) of the $i^{th}$ sample by the model. The smaller the $MSE$ and the closer the $R^2$ to the value of 1 ($R^2 \leq 1$) obtained from the model, the higher the model's performance in terms of prediction accuracy. A critical comparative analysis was performed in terms of $MSE$ and $R^2$ among FANN by [15] and PCA-FANN by [16], as proposed in previous research.

## 3. RESULTS AND DISCUSSION

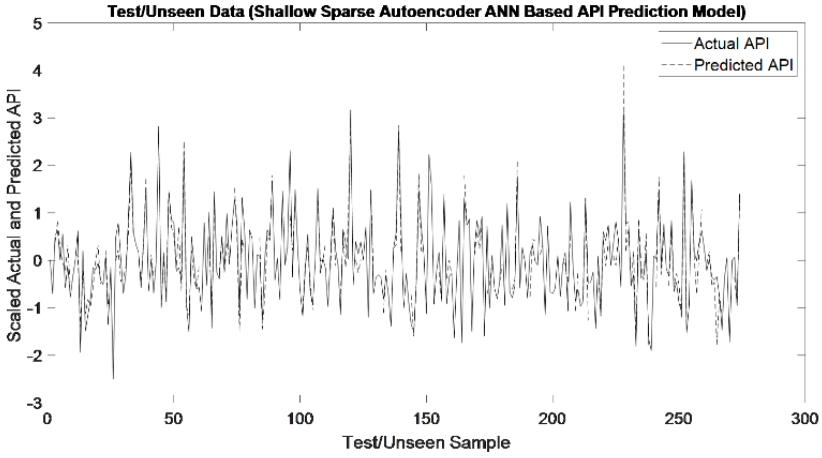
The full architecture of the developed sparse autoencoder ANN-based API prediction model with shallow, sparse autoencoder and deep sparse autoencoder is shown in Figure 9(a) and Figure 9(b), respectively. The input for the API prediction model consists of the data from 6 variables: air temperature, relative humidity, wind speed, CO concentration, $O_3$

https://doi.org/10.31436/iiumej.v26i1.2818

concentration, and $PM_{10}$ concentration. The shallow sparse autoencoder was constructed with one bottleneck hidden layer, while the deep sparse autoencoder was developed with one encoding hidden layer and one bottleneck hidden layer. They played a role in dimensionality reduction (features extraction) on the input and were both connected with a shallow feedforward neural network, each for relationship learning purposes between the features captured in the bottleneck hidden layer of the respective autoencoder and the output/target variable data (API). The encoder part of the sparse autoencoder was trained with a scaled conjugate gradient descent algorithm and logistic sigmoid activation/transfer function. The hidden layer and the output layer of the feedforward neural network were both trained with scaled conjugate gradient backpropagation function and were trained with hyperbolic tangent sigmoid activation/transfer function and linear activation/transfer function, respectively.




(a)



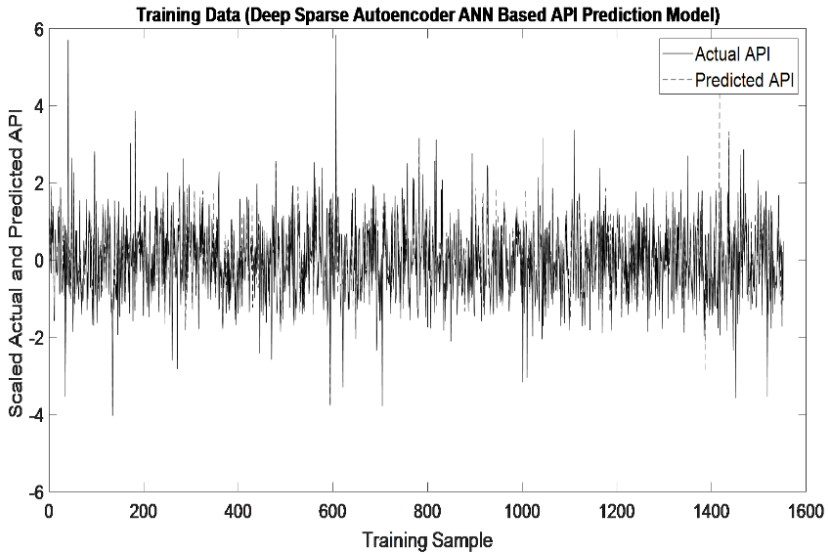

(b)

Figure 10. Actual and predicted API of (a) training sample and (b) test/unseen sample with shallow sparse autoencoder

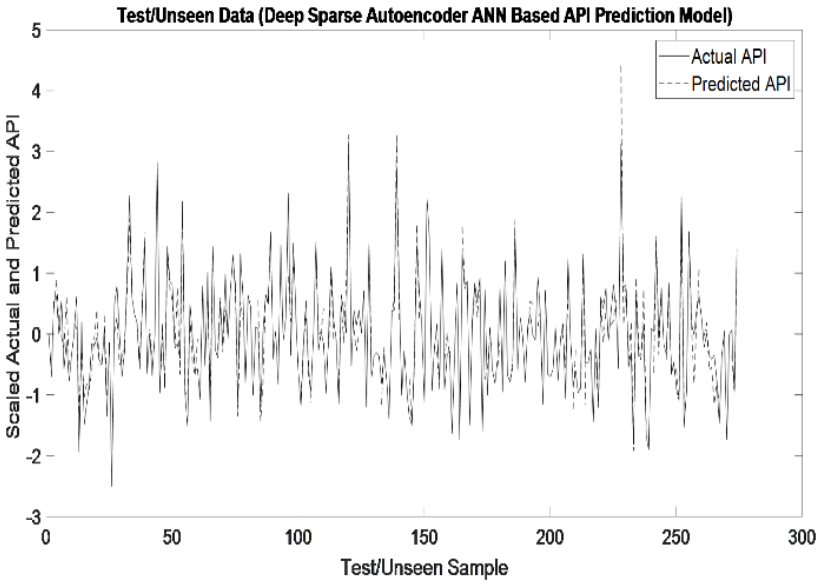
The shallow sparse autoencoder was developed with 3 hidden nodes in its bottleneck hidden layer, while the deep sparse autoencoder was developed with 3 hidden nodes in its encoding hidden layer and 2 hidden nodes in its bottleneck hidden layer. The shallow feedforward neural network was constructed with 11 hidden nodes in its hidden layer and 1 node in its output layer. The output for the API prediction model is the predicted target variable data (API). The model was first trained with the training data set to determine the optimum model parameter values and algorithms by applying theoretical knowledge and employing trial

https://doi.org/10.31436/iiumej.v26i1.2818

and error methods. The trained model was then tested with the test/unseen data set. The API prediction model performance was evaluated in terms of $MSE$ and $R^2$ Two critical comparative analyses are being conducted: a comparative analysis between the observed true API and predicted API for both training and test/unseen data sets and a comparative analysis between the developed sparse autoencoder ANN-based API prediction model and the other prediction models proposed in previous research. Figure 10 shows the API prediction performance of the developed shallow, sparse autoencoder ANN-based model on both training and test/unseen data sets in graphical form. The x-axis is the sample number, and the y-axis is the scaled actual and predicted API.




(a)



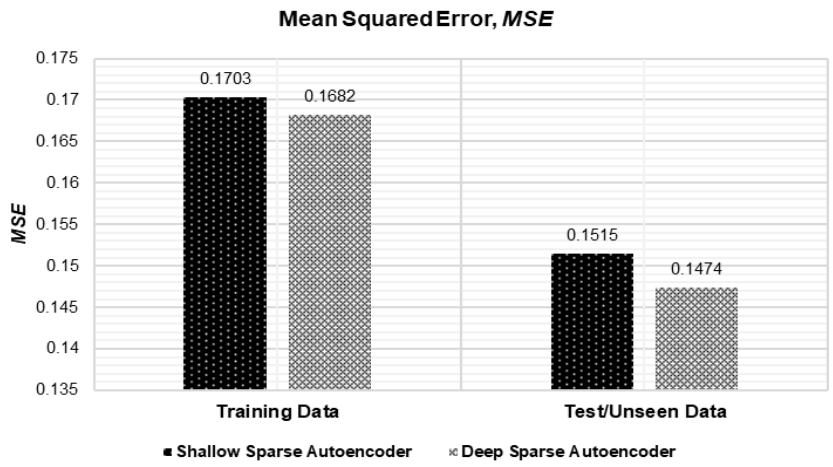

(b)

Figure 11. Actual and predicted API of (a) training sample and (b) test/unseen sample with deep sparse autoencoder

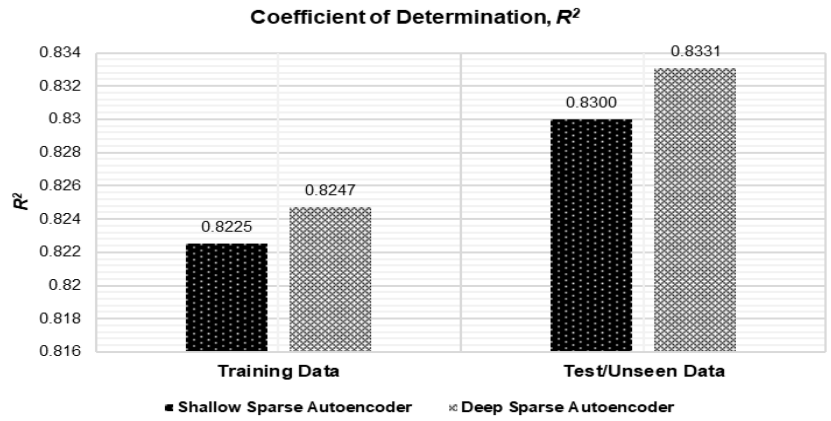
Figure 11 shows the API prediction performance of the developed deep sparse autoencoder ANN-based model on both training and test/unseen data sets in a graphical form where the x-

axis is the sample number, and the y-axis is the scaled actual and predicted API. Based on Figure 10 and Figure 11, it is clearly observed that the API prediction of the proposed shallow/deep sparse autoencoder ANN-based model emulates the actual API behavioral pattern relatively well for both training and test/unseen data.

To further evaluate the API prediction performance from a statistical point of view, the $MSE$ and $R^2$ are computed between the actual API and the predicted API on both training and test/unseen data sets to the architecture (shallow/deep) of the sparse autoencoder, which serves as the base of the proposed API prediction model as tabulated in Table 6 and illustrated in Figure 12. Based on Table 6 and Figure 12, it is observed that the predicted API values are relatively close to the actual API values of both training and test/unseen samples regardless of shallow sparse autoencoder or deep sparse autoencoder as the base of the API prediction model with low $MSE (MSE < 0.1800)$ and $R^2$ which is close to 1 $(0.8000 < R^2 < 1)$. In other words, the degree of discrepancy between the actual and predicted data by both architectures is very small. Thus, the proposed and developed sparse autoencoder ANN-based model in API prediction is proven to be valid due to the ability of the model to generalize and model the problem effectively with significantly high relevancy in target variable data prediction.




(a) MSE




(b) $R^2$

Figure 12. Statistical analysis of the API prediction model performance

https://doi.org/10.31436/iiumej.v26i1.2818

Table 6. Statistical analysis of the API prediction model performance

| Architecture of Sparse Autoencoder | Training Data | | Test/Unseen Data | |
|---|---|---|---|---|
| | $MSE$ | $R^2$ | $MSE$ | $R^2$ |
| Shallow | 0.1703 | 0.8225 | 0.1515 | 0.8300 |
| Deep | 0.1682 | 0.8247 | 0.1474 | 0.8331 |

Besides, it is observed that the API prediction model with deep sparse autoencoder achieves higher prediction performance with higher prediction accuracy due to lower MSE and closer $R^2$ to the value of 1 obtained compared to the API prediction model with shallow sparse autoencoder. The results and findings obtained in this study are in accordance with the theoretical studies where numerous hidden layers in the deep sparse autoencoder architecture allow the ANN to learn higher-order features of the input and capture unknown data structure, further increasing the model generalization performance and effectiveness. Thus, the proposed and developed sparse autoencoder ANN-based API prediction model is further validated.

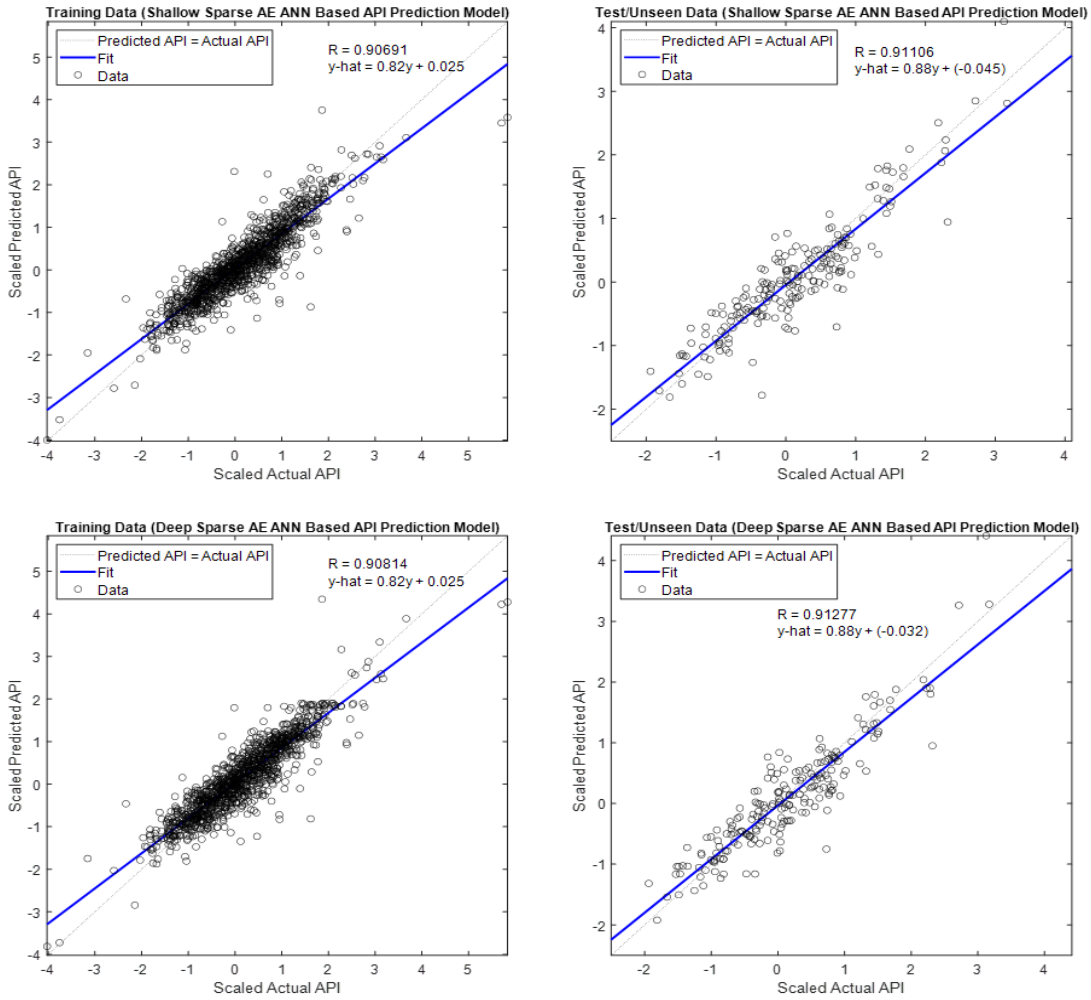



Figure 13. Linear regression plot of actual API relative to predicted API, where AE stands for 'autoencoder'

A linear regression of actual API relative to predicted API is plotted as shown in Figure 13 on both training and test/unseen data sets, each with respect to the architecture (shallow/deep) of the sparse autoencoder, which serves to be the base of the proposed API prediction model. A comparative analysis between the developed shallow/deep sparse

https://doi.org/10.31436/iiumej.v26i1.2818

autoencoder ANN-based API prediction model and other proposed API prediction models in previous research, which are feedforward artificial neural network (FANN) and principal component analysis – feedforward artificial neural network (PCA-FANN), is conducted further to validate the proposed API prediction model in this study. Table 7 presents the overall statistical analysis of the performance of all the API prediction models in terms of $MSE$ and $R^2$ on unseen data.

Table 7. Overall statistical analysis of the API prediction model performance on unseen data

| **API Prediction-Based Model** | ***MSE*** | $\boldsymbol{R^2}$ |
|---|---|---|
| Shallow sparse autoencoder | 0.1515 | 0.8300 |
| Deep sparse autoencoder | 0.1474 | 0.8331 |
| FANN from [16] | 0.1856 | 0.7950 |
| PCA from [17] | 7.5620 ($RMSE$) | 0.7360 |

Based on Table 7, it is observed that the proposed sparse autoencoder ANN-based API prediction model, regardless of shallow or deep sparse autoencoder, has the highest performance on the unseen data among the other proposed models in previous research, with $MSE$ value of 0.1515 and $R^2$ value of 0.8300 for shallow sparse autoencoder, and $MSE$ value of 0.1474 and $R^2$value of 0.8331 for deep sparse autoencoder. It is evidently clear that the proposed API prediction model in this study further improves the accuracy of API prediction. Hence, the proposed model in this study is further validated once again. the smallest $MSE$ and the largest $R^2$(closest to 1) are both achieved by the developed deep sparse autoencoder ANN-based API prediction model; thus, the model is determined to be the best model in API prediction and selected as the final model structure in this study.

The research paper demonstrated the flexible application of shallow and deep sparse autoencoder artificial neural network models, presenting a novel alternative method in the field of API prediction. The deep sparse autoencoder model, in particular, demonstrates improved prediction performance, making it a promising tool for API prediction. However, there are several limitations to this research that need to be addressed for this model to be considered the best for API prediction. Firstly, the model's training and testing are based on data from specific monitoring stations in Perak, which may limit its applicability to other regions. Secondly, the study only considers six input variables, potentially overlooking other relevant factors that could influence API prediction. Thirdly, the model uses data from 2006 to 2010, which may not reflect more recent trends or changes in air pollution patterns. Lastly, the deep sparse autoencoder, while effective, is a complex model that requires longer training times compared to simpler models. Addressing these limitations in future research could enhance the model's performance and its utility in API prediction.

## 4. CONCLUSION

This study proposed data-driven modeling for API prediction using autoencoder ANN with the fundamental objective of improving the API prediction performance. Two API prediction models were developed. One used a shallow sparse autoencoder, while the other used a deep sparse autoencoder. The model with shallow sparse autoencoder could predict the API with promising accuracy, with an $MSE$ value of 0.1515 and $R^2$ value of 0.8300 on the test/unseen data. However, the shallow sparse autoencoder may lack robustness due to the presence of only one single hidden layer in the representation of learning features. When

applied to test/unseen data, it may face difficulties and challenges. To further improve the model generalization and prediction accuracy, an API prediction model with a deep sparse autoencoder was proposed. Based on the results and findings, it is concluded that the model with the deep sparse autoencoder does improve the prediction performance compared to the model with the shallow sparse autoencoder, with $MSE$ value of 0.1474 and $R^2$ value of 0.8331 on the test/unseen data. Nevertheless, both proposed architectures are concluded to be valid as they both show significant improvement in API prediction performance with relatively small values of $MSE$ and $R^2$ values that are relatively close to 1 as compared to other API prediction models proposed in previous research. Since the proposed and developed deep sparse autoencoder ANN-based API prediction model achieved the smallest $MSE$ value with a $R^2$ value significantly close to 1, the model is, thus, determined to be the best model in API prediction and selected as the final model structure in this study.

## ACKNOWLEDGEMENT

This work was supported by Kementerian Pendidikan Malaysia (KPM) through the Fundamental Research Grant Scheme (FRGS) grant number FRGS/1/2022/TK05/USM/01/5. Special gratitude to the Department of Environmental (DOE) Malaysia for providing the air quality data for this study and Universiti Sains Malaysia (USM) for the support.

## REFERENCES

[1] Li X, Peng L, Yao X, Cui S, Hu Y, You C, Chi T (2017) Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation. Environmental Pollution 231:997-1004. https://doi.org/10.1016/j.envpol.2017.08.114

[2] Box GEP, Jenkins GM, Reinsel GC, Ljung GM (2015) Time series analysis: Forecasting and control, 5th edn. Wiley, Hoboken.

[3] Li C, Hsu NC, Tsay S-C (2011) A study on the potential applications of satellite data in air quality monitoring and forecasting. Atmospheric Environment 45:3663-3675. https://doi.org/10.1016/j.atmosenv.2011.04.032

[4] García Nieto PJ, Combarro EF, del Coz Díaz JJ, Montañés E (2013) A SVM-based regression model to study the air quality at local scale in Oviedo urban area (Northern Spain): A case study. Applied Mathematics and Computation 219(17):8923-8937. https://doi.org/10.1016/j.amc.2013.03.018.

[5] McClelland JL, Rumelhart DE (1986) A distributed model of human learning and memory. In: McClelland JL, Rumelhart DE, the PDP Research Group (eds) Parallel distributed processing: Explorations in the microstructure of cognition, Vol 2: Psychological and biological models. MIT Press, Cambridge, 170-215.

[6] Anderson JA, Silverstein JW, Ritz SA, Jones RS (1977) Distinctive features, categorical perception, and probability learning: Some applications of a neural model. Psychological Review 84(5):413-451. https://doi.org/10.1037/0033-295X.84.5.413

[7] Kohonen T (1977) Associative memory: A system-theoretical approach. Springer-Verlag, New York.

[8] Rumelhart DE, Hinton GE, Williams RJ (1986) Learning internal representations by error propagation. In: Rumelhart DE, McClelland JL (eds) Parallel distributed processing: Explorations in the microstructure of cognition, Vol 1: Foundations. MIT Press, Cambridge, 318-362.

[9] Rumelhart DE, Durbin R, Golden R, Chauvin Y (1996) Backpropagation: The basic theory. In: Smolensky P, Mozer MC, Rumelhart DE (eds) Mathematical perspectives on neural networks. Lawrence Erlbaum Associates, Mahwah, 533-566.

[10] Rumelhart DE (1989) Toward a microstructural account of human reasoning. In: Vosniadou S, Ortony A (eds) Similarity and analogical reasoning. Cambridge University Press, New York, 298-312.

[11] Baldi P, Hornik K (1989) Neural networks and principal component analysis: Learning from examples without local minima. Neural Networks 2:53-58. https://doi.org/10.1016/0893-6080(89)90014-2.

[12] Pathirage CSN, Li J, Li L, Hao H, Liu W (2018) Application of deep autoencoder model for structural condition monitoring. Journal of Systems Engineering and Electronics 29(4):873-880. https://doi.org/10.21629/JSEE.2018.04.22.

[13] Miao X, Miao H, Jia Y, Guo Y (2018) Using a stacked-autoencoder neural network model to estimate sea state bias for a radar altimeter. PLoS ONE 13(12): e0208989. https://doi.org/10.1371/journal.pone.0208989.

[14] Liu JNK, Hu Y, He Y, Chan PW, Lai L (2015) Deep neural network modeling for big data weather forecasting. In: Pedrycz W, Chen S-M (eds) Information granularity, big data, and computational intelligence. Springer, Heidelberg, 389-408.

[15] Rahim NA, Ahmad Z (2017) Graphical user interface application in Matlab environment for water and air quality process monitoring. Chemical Engineering Transactions 56:97-102. https://doi.org/10.3303/CET1756017.

[16] Ahmad Z, Rahim NA, Bahadori A, Zhang J (2017) Air pollution index prediction using multiple neural networks. International Islamic University Malaysia Engineering Journal 18(1):1-12. https://doi.org/10.31436/iiumej.v18i1.684

[17] Azid A, Juahir H, Latif MT, Zain SM, Osman MR (2013) Feed-forward artificial neural network model for air pollutant index prediction in the southern region of Peninsular Malaysia. Journal of Environmental Protection 4:1-10. https://doi.org/10.4236/jep.2013.412A1001

[18] McLoone S, Irwin GW (2001) Improving neural network training solutions using regularisation. Neurocomputing 37(1-4):71-90. https://doi.org/10.1016/S0925-2312(00)00314-3.

[19] Department of Environment, Malaysia (2012) Malaysia environmental quality report 2012. Enviro Knowledge Management Center (EKMC). https://enviro2.doe.gov.my/ekmc/digital-content/86803/. Accessed 2 January 2021.

[20] Xu G, Fang W (2016) Shape retrieval using deep autoencoder learning representation. In: Proceedings of the 13th international computer conference on wavelet active media technology and information processing (ICCWAMTIP),227-230. Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/ICCWAMTIP.2016.8079843

[21] Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A (2010) Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research 11:3371-3408.

[22] Olshausen BA, Field DJ (1997) Sparse coding with an overcomplete basis set: A strategy employed by V1?. Vision Research 37(23):3311-3325. https://doi.org/10.1016/S0042-6989(97)00169-7.

[23] Pathirage CSN, Li J, Li L, Hao H, Liu W, Wang R (2019) Development and application of a deep learning-based sparse autoencoder framework for structural damage identification. Structural Health Monitoring 18(1):103-122. https://doi.org/10.1177/1475921718800363.

[24] Drucker H, Schapire R, Simard P (1993) Improving performance in neural networks using a boosting algorithm. In: Hanson SJ, Cowan JD, Giles CL (eds) Advances in neural information processing systems 5. Morgan Kaufmann Publishers, San Francisco, 42-49.

[25] Kambhatla N, Leen TK (1997) Dimension reduction by local principal component analysis. Neural Computation 9:1493-1516. https://doi.org/10.1162/neco.1997.9.7.1493

[26] Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319-2323. https://doi.org/10.1126/science.290.5500.2319

[27] Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504-507. https://doi.org/10.1126/science.1127647.

[28] Hinton GE, Osindero S, Teh Y-W (2006) A fast learning algorithm for deep belief nets. Neural Computation 18(7):1527-1554. https://doi.org/10.1162/neco.2006.18.7.1527

[29] Bengio Y, Lamblin P, Popovici D, Larochelle H (2007) Greedy layer-wise training of deep networks. In: Schölkopf B, Platt J, Hofmann T (eds) Advances in neural information processing systems 19. MIT Press, Cambridge, 153-160.