



Article

An Accurate CT Saturation Classification Using a Deep Learning Approach Based on Unsupervised Feature Extraction and Supervised Fine-Tuning Strategy

Muhammad Ali , Dae-Hee Son, Sang-Hee Kang  and Soon-Ryul Nam *

Department of Electrical Engineering, Myongji University, Yongin 449-728, Korea; friendsspecial.ali@gmail.com (M.A.); sonking18@nate.com (D.-H.S.); shkang@mju.ac.kr (S.-H.K.)

* Correspondence: ptsouth@mju.ac.kr; Tel.: +82-31-330-6361

Received: 12 October 2017; Accepted: 7 November 2017; Published: 10 November 2017

Abstract: Current transformer (CT) saturation is one of the significant problems for protection engineers. If CT saturation is not tackled properly, it can cause a disastrous effect on the stability of the power system, and may even create a complete blackout. To cope with CT saturation properly, an accurate detection or classification should be preceded. Recently, deep learning (DL) methods have brought a subversive revolution in the field of artificial intelligence (AI). This paper presents a new DL classification method based on unsupervised feature extraction and supervised fine-tuning strategy to classify the saturated and unsaturated regions in case of CT saturation. In other words, if protection system is subjected to a CT saturation, proposed method will correctly classify the different levels of saturation with a high accuracy. Traditional AI methods are mostly based on supervised learning and rely heavily on human crafted features. This paper contributes to an unsupervised feature extraction, using autoencoders and deep neural networks (DNNs) to extract features automatically without prior knowledge of optimal features. To validate the effectiveness of proposed method, a variety of simulation tests are conducted, and classification results are analyzed using standard classification metrics. Simulation results confirm that proposed method classifies the different levels of CT saturation with a remarkable accuracy and has unique feature extraction capabilities. Lastly, we provided a potential future research direction to conclude this paper.

Keywords: current transformer (CT) saturation; deep neural networks (DNNs); autoencoder; classification; deep learning (DL); unsupervised feature extraction; supervised fine-tuning strategy

1. Introduction

1.1. Motivation

With the development of differential schemes, CT saturation has been one of the foremost problem for power engineers. CT saturation causes the erroneous measurement of primary currents. This can lead to severe malfunction of protection relays. Transformers are one of the most essential and expensive components of power systems. If an engineer does not possess a deep understanding of relay behavior during CT saturation, this can lead to false trips with catastrophic consequences.

As per IEEE standard C37.110, CTs are designed so that the CT cores are not-saturated when the direct current (DC) offset on the primary side rises nearly to 20 times of rated current and burden on the secondary side does not exceed its nominal value [1,2]. Conversely, the short circuit capacity of substation rises due to the expansion of interconnected power systems. Therefore, CT saturation has become an unavoidable event due to the growth of significant burden impedance in CT, remnant flux, and large primary currents due to a fault [3].

Over the years, several approaches have been proposed by researchers to mitigate the CT saturation problem [4–6]. Among them, the harmonic restraint scheme is extensively used to reduce the impact of CT saturation. Harmonic information is utilized to detect the CT saturation detection and compensation. However, these methods use filters along with harmonic components to secure the relays from magnetic inrush or over-excitation situations. The filtering procedure can delay the protection process [7,8]. Besides this, if harmonics are present in the unsaturated CT secondary signal then the detector output may be high even though there is no saturation. In such situations, harmonic based methods may mal-operate [9]. Moreover, Shah and Bhalja [10] carried out a brief analysis of artificial intelligence (AI) methods and conventional harmonic schemes considering the various disturbances and CT saturation conditions. Comparison results reveal that AI methods perform better than conventional harmonic schemes. Furthermore, conventional protection schemes [11–15] use various signal processing techniques to extract the meaningful features. Due to manual feature selection, we sometimes capture unimportant or useless features, which increases the model complexity. On the other hand, AI methods based on deep learning (DL) are generic in nature and does not need the hand engineered feature extraction. They have a better ability to find complex relation (features) between data as compared to conventional methods. Importantly, the presented DL method in this study is independent of the harmonic present in its operating signal, so no filtering process is required.

One of the key ways to protect CTs is to classify saturated and unsaturated regions accurately and rapidly. Failure to classify the saturated and unsaturated regions can affect both the stability and reliability of the entire power system. Recently, DL [16–18] approaches have brought massive revolution in the field of AI due to their unsupervised feature extraction abilities. Specifically, DL algorithms produce promising results in classification tasks as they extract high-level features from data through a hierarchical learning process.

1.2. Prior Works

This section will highlight some of the promising AI approaches that have been proposed previously to cope with CT saturation problem. All methods are summarized in Table 1. An algorithm based on Gaussian Mixture Models (GMMs) is reported in [19]. The GMMs were trained with the secondary current of CT under a variety of conditions to classify the saturated and unsaturated regions. However, these type of models only classify low-dimensional data and performs badly on high dimensional complex data. Additionally, these models used supervised classification techniques which heavily rely on human designed features and suffer mostly from overfitting issues.

Table 1. Summary of past research based on artificial intelligence (AI) methods.

Studies	Technique Used	Classified Objects	Feature Extraction Method	Accuracy
M. Haji; Vahidi, [19]	Gaussian Mixture Models	2 classes <ul style="list-style-type: none"> • Saturated • Not-saturated 	Supervised	94.96%
Rebizant; Bejmert, [20]	Genetic algorithms	Population values 2 classes	Supervised	97.83%
Moradi; Alinejad, [21]	Particle Swarm Optimization (PSO)	<ul style="list-style-type: none"> • Saturated • Not-saturated 	Supervised	95.00%
Segatto; Coury, [22]	Elman ANN	2 classes <ul style="list-style-type: none"> • Saturated • Not-saturated 	Supervised	98.5%
Sheng; Rovnyak, [23]	Decision Trees and Wavelet Analysis	Wavelet detail coefficients	Supervised	95.00%
This Study	Deep Neural Networks with Autoencoders	3 classes <ul style="list-style-type: none"> • Not-saturated • Heavy-saturated • Light-saturated 	Unsupervised	99.9%

Rebizant et al. [20] further introduced a more advanced method based on genetic algorithms (GA). Although GAs have proven effective in many optimization problems, they tend to converge near local optima rather than the global optimum. Due to the slow convergence rate, these algorithms are hard to implement in real world problems. An intelligent approach based on particle swarm optimization (PSO) technique was presented in [21] to discriminate CT saturation in power transformers. Compared to GAs, PSOs are easy to implement and computationally more efficient. However, the main defect of PSO is that global convergence cannot be guaranteed and it can easily overfit the data. In fact, it is not an ideal approach for classification due to the fitness function. The algorithm described in [22] only attempts to correct the distorted waveforms caused by CT saturation. The authors did not consider the various levels (heavy, light, normal) of saturation. Also, the proposed scheme relies heavily on human-crafted feature extraction, which dramatically reduces the generalization performance of the network.

Furthermore, Yong and Steven [23] used decision trees and wavelets for CT saturation analysis. This scheme used signal processing techniques to extract features by wavelet decomposition in addition to the differential and restraining currents. As a supervised learning method, this scheme requires extensive expertise in signal processing techniques to extract meaningful features.

It appears that all of these methods have at least one of the following restraints. Firstly, these methods are heavily dependent on signal processing techniques. Secondly, most of the methods are based on supervised feature extraction, which means that human interference is required to extract useful features. A considerable amount of engineering skill and domain expertise is required to construct meaningful features (patterns) for a certain task. Thirdly, these traditional methods are easily trapped into local optima that lead to a poor generalization of the network. Fourthly, most studies suffer from the curse of dimensionality and overfitting issues. Lastly, most of these traditional methods are only limited to CT saturation detection and compensation. Different levels of saturation are highly underestimated.

1.3. Key Contribution

Motivated by the above-mentioned problems, we aim to present an effective deep learning approach based on deep neural network (DNN) and autoencoders (AEs). A DNN is constructed with an autoencoder to capture the high-level features from CT saturation data. It is worth mentioning that proposed method uses an unsupervised feature extraction technique to extract the useful features automatically without human interference and classifies different levels of CT saturation with high accuracy. This is different to traditional methods where features are extracted through human expertise or signal processing techniques. To overcome the problems of local optima and network generalization from previous works, we employed a greedy layer-by-layer unsupervised training strategy which brings better generalization ability and avoids local optimization problems. In addition, we applied a popular “L2 regularization” technique to overcome the curse of dimensionality and combat overfitting issues. Finally, we optimized the results from the DNN with a fine-tuning strategy. To validate the effectiveness of the proposed method, several simulation tests were conducted, and performance is evaluated using standard performance metrics. Lastly, an empirical comparison is carried out with the traditional machine learning methods to check the superiority of proposed method. To the best of authors’ knowledge, this is the first study that makes use of deep learning for CT saturation classification. The outcome of this paper will provide promising research options that can enrich the deep learning technology and widen its range of applications in power systems.

The rest of the paper is organized as follows: Section 2 presents the framework of the proposed method. The data collection and preprocessing steps are demonstrated in Section 3. Section 4 provides the basic literature review of DNN and AEs for CT saturation classification. In Section 5, we describe the unsupervised feature extraction method and fine-tuning strategy. Experimental results and discussions along with performance comparison with traditional classification methods are presented in Sections 6 and 7. Lastly, some conclusions are drawn with potential future research directions in Section 8.

2. Methodology

The proposed method consists of three main parts and is outlined in Figure 1. In the first part, data is collected using Power System Electromagnetic Transients Simulation Software (PSCAD/EMTDC (4.5)) based designed model, and results are stored in the output file. The collected data are preprocessed before sending them to the DNN. The second part is devoted to the feature extraction process, where an AE is used for unsupervised feature extraction. The type of autoencoder we have trained is a sparse autoencoder (SAE). Two AEs are trained in unsupervised fashion by using greedy layer-by-layer pre-training strategy. The application of greedy layer-by-layer training strategy has solved the problems of poor local optima and generalization in DNN. During training, the first autoencoder is trained, and features are extracted in the hidden layer. Likewise, the second AE is trained in similar fashion, except that we used the features of the first AE as inputs of the second AE. The extracted features are then fed into softmax layer, which is trained in a supervised manner. As the AE learns compact data competently, it can be stacked to build deep networks. Hence, we stacked the two AEs with softmax layer to form a deep model. After building a deep model, results are finally optimized by retraining the whole model in supervised fashion. This procedure is often known as fine-tuning. The classification results show that fine-tuning approach brings significant improvement in network performance. The final part of the study demonstrates model evaluation and performance analysis. We evaluate the proposed method and compare it to traditional classification methods using cross-validation, confusion matrices and a variety of performance metrics, such as accuracy, precision, recall, specificity, F1-score, Matthews correlation coefficient (MCC) and receiver operating characteristic (ROC) curves.

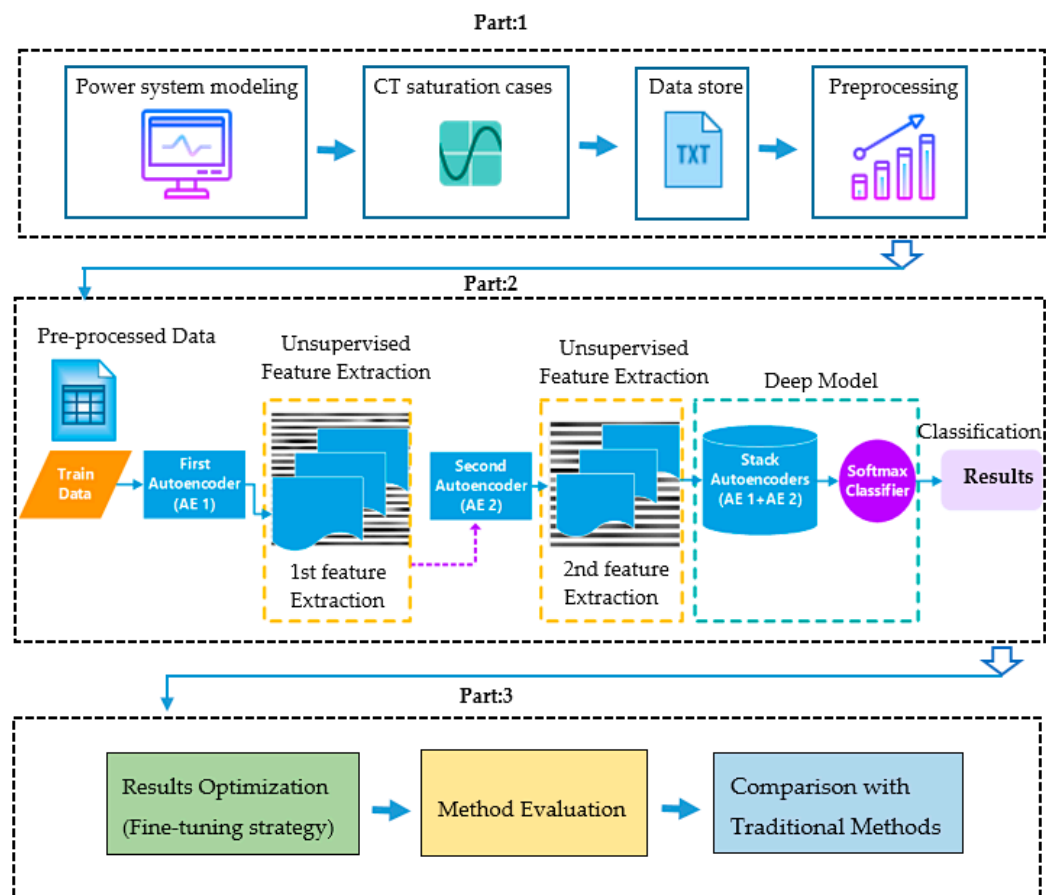


Figure 1. Framework of the proposed approach.

3. Data Collection and Preprocessing

3.1. Power System Modeling and Data Collection

To prepare the CT saturation data, we have designed a power system model using the PSCAD/EMTDC software package [24], and results are stored in the output file. We simulated a PSCAD/EMTDC-based 230-kV transmission system with two sources and a transmission line, as shown in Figure 2. The two AC power sources are denoted with S1 and S2, and the equivalent impedances are denoted with Z_{S1} and Z_{S2} , respectively. The model incorporated all of the power system components in the PSCAD/EMTDC master library. Moreover, we used the standard Lucas CT model described in [25] to study the saturation of CT. The presented model is simple and accurate for the CT saturation as implemented in [26].

When CT is saturated, the magnetizing current increases sharply due to the growth of CT core flux. As a result, the secondary current is distorted. The secondary and magnetizing current waveforms undergo abrupt changes when saturation begins and ends. Figure 3 shows different levels of CT saturation obtained from simulation studies. We generated a large number of CT saturation cases by varying parameters such as the CT burden, fault resistance, remnant flux, and fault type, then stored the results in an output file. The generated data in PSCAD/EMTDC are further processed in MATLAB (2016b) with a sampling frequency of 3.84 kHz (64 samples/cycle) under the 60-Hz system. A dataset of 16,863 cases with different levels of saturation (heavy, light, and normal) has been generated. The data are then randomly partitioned into a training set (75%) and a testing set (25%). This resulted in a training set containing 12,646 cases in total, and a test set with 4216 cases. For statistical validation of results, we employed a k -fold cross-validation technique, where the value of k is set to four. The generated CT saturation cases were used as input data to the DNN whereas the desired output of the network was three possible levels of saturation. We converted the outputs to binary using the one-hot encoding method [27].

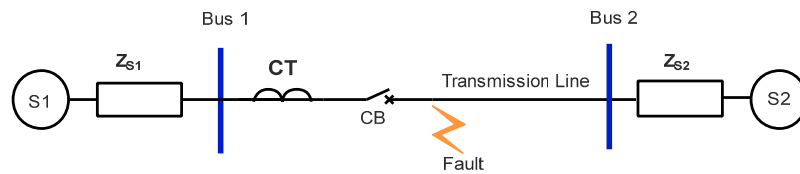


Figure 2. Simulated power system model in PSCAD.

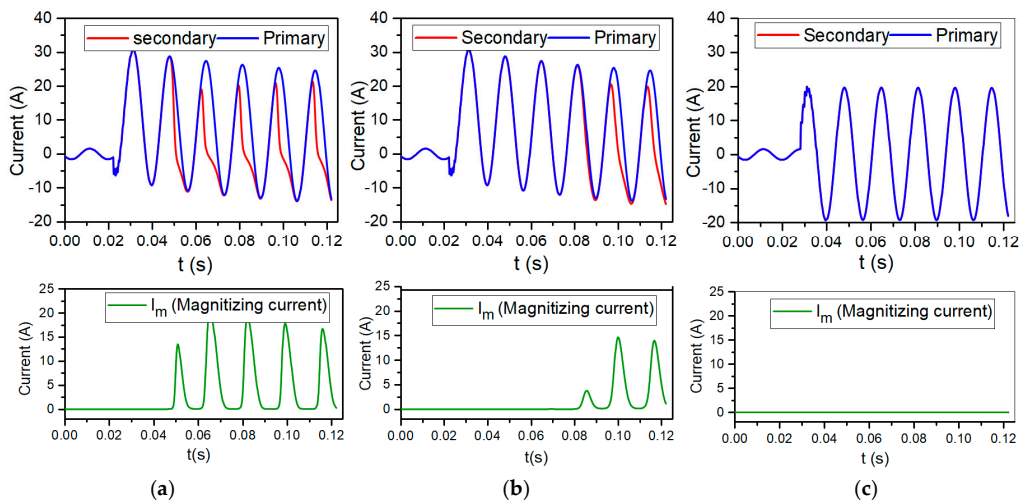


Figure 3. Different levels of the current transformer (CT) saturation obtained from simulation results: (a) Heavy Saturation; (b) Light Saturation; (c) No Saturation.

3.2. Preprocessing Steps

To improve the accuracy of the classification and prevent complications arising from convergence during training, we normalized the sample data into the range (± 1) . We used the following mapping formula to normalize the sample data:

$$X' = \frac{2 \times (x_i - x_{\min})}{x_{\max} - x_{\min}} - 1, \quad i = 1, 2, 3 \dots, N \quad (1)$$

where X' is the normalized value; x_i is the i th sample of initial input data; x_{\max} and x_{\min} denotes the maximum and minimum values of the dataset, respectively. N is the total number of input data.

In addition, we used a sliding window to check the saturation conditions at each instant. The sliding window removes an old sample and adds a new sample into the window. The sampling frequency was 3.84 kHz, which corresponds to the 64 samples per cycle, as shown in Figure 4. Consequently, sliding window updates the new samples of the model iteratively. A sliding window is applied across the current samples as follows:

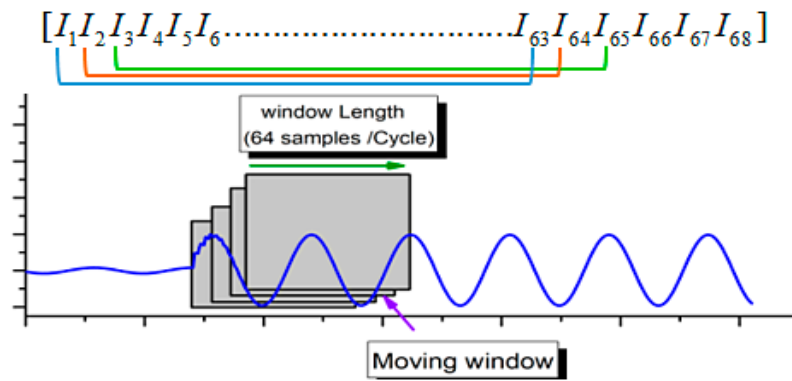


Figure 4. A sliding window to continuously monitor the CT saturation states at each instant. It discards the old samples from the window and inserts new samples towards the right.

4. Deep Neural Network and Autoencoders for CT Saturation Classification

DL has completely revolutionized modern AI due to its pattern recognition abilities. As a consequence, DL algorithms based on AEs and DNNs have been used extensively in recent years [28]. In general, there are three key approaches to DL: (a) AEs; (b) convolutional neural networks (CNNs); and (c) deep belief networks (DBNs). This study mainly aims at AEs with DNNs to classify CT saturation levels. In the next section, we outline some of the basic concepts behind AEs and DNNs.

4.1. Fundamentals of DNN and AEs

A DNN is a family of ANNs that consists of many hidden layers. The deep architecture of DNN consists of several hidden layers, and every hidden layer bears a nonlinear transformation from the previous layer to next one [16]. In the context of DL, the idea of pre-training was introduced by Hinton and Salakhutdinov in 2006 [17]. In this paper, we based our usage of pre-training and the fine-tuning strategy on the ideas presented in [17]. The pre-training algorithm ensures generalization and fine-tuning allows deep networks to train more efficiently. Due to multiple hidden layers in DNNs, they possess great potential to capture powerful abstracted features from training data [29].

Furthermore, an AE is an unsupervised learning algorithm that utilizes backpropagation to set the target output to be its input as shown in Figure 5. The specialty of AE is that it finds patterns in a dataset by detecting the key features. As a result, they are used extensively in feature extraction applications, learning generative models, dimensionality reduction, and even compression. Figure 5

shows an AE that contains three layers, namely, input layer that represents the input, hidden layer that shows the learned features and output layer that denotes the reconstruction.

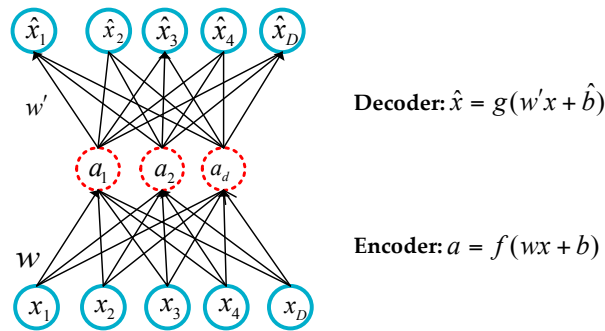


Figure 5. Structure of a typical autoencoder model.

The input and hidden layers from encoder part are responsible for mapping the input data into hidden representations. However, the hidden and output layers from decoder part are responsible for rebuilding the original input data from the learned hidden representations.

A dataset with the input vector x , the representation vector a_d and reconstruction vector \hat{x} can be represented by Equations (2) and (3), where w and w' signifies the weights of encoder and decoder respectively. The mapping functions are shown with $f(\cdot)$ and $g(\cdot)$. In this study, we used a sigmoid mapping function. Furthermore, the reconstruction error between x and \hat{x} is given in Equation (4). The overall cost function in the case of m training samples is defined in Equation (5). Further details are presented in [30].

$$a = f(wx + b) \quad (2)$$

$$\hat{x} = g(w'x + \hat{b}) \quad (3)$$

$$J(W, b; x, \hat{x}) = \frac{1}{2} \|h_{W,b}(x) - \hat{x}\|^2 \quad (4)$$

$$J(W, b) = \left[\frac{1}{m} \sum_{i=1}^m J(W, b; x, \hat{x}) \right] + \frac{\lambda}{2} \sum_{l=1}^{n_l-1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} (W_{ji}^{(l)})^2 \quad (5)$$

In Equation (5), λ is a weight decay term, n_l shows the number of layers in the network and $W_{ji}^{(l)}$ specifies the weight of i^{th} neuron of layer l . The expression b represents a bias term. The first term in Equation (5) represents the reconstruction error, and the second term is a weight decay term, also known as a regularization term. The purpose of the regularization term is to minimize the magnitude of the weights to avoid overfitting.

4.2. Sparsity of the Autoencoder (AE)

Generally, an AE simply copies the input layer to the hidden layer and then reconstructs the input data in the output layer. However, we cannot extract meaningful features in this way, despite the input data being perfectly reconstructed at the output layer. Researchers usually introduce a sparse constraint to the cost function of the AE [31–33]. This results in a better representation of the features and helps to identify more general structures in the input data. We imposed a sparsity restraint on the hidden units of the AE to obtain more robust features from the input data, even in the case where the number of hidden units is bigger than the number of inputs nodes. Equation (6) shows the overall cost function of sparse autoencoder (SAE), where we minimize the cost function $J(W, b)$ from Equation (5) to obtain a sparse representation. Here, β is the weight of sparsity term and S_2 is the number of hidden

units. Moreover, the average activation of the hidden unit j in the AE is defined in Equation (7). Here $\hat{\rho}_j$ is equal to ρ which is known as sparsity parameter. It has a small value typically close to 0 [34].

$$J_{\text{sparse}}(W, b) = J(W, b) + \beta \sum_{j=1}^{s_2} \text{KL}(\rho \| \hat{\rho}_j) \quad (6)$$

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^m \left[a_j^{(2)}(x^{(i)}) \right] \quad (7)$$

$$\text{KL}(\rho \| \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j} \quad (8)$$

where $\text{KL}(\rho \| \hat{\rho}_j)$ in Equation (8) is the Kullback-Leibler (KL) divergence [35], which forms the basis of our penalty term. The SAE performs better than a traditional AE and discovers sparse representations of features. By minimizing the cost function $J_{\text{sparse}}(W, b)$ using a backpropagation algorithm, it extracts more information from the input data, instead of simply converting the inputs. In order to construct an improved model, a scaled conjugate gradient (SCG) descent algorithm is utilized to update the parameters W and b .

5. Learning CT Saturation Features by Stacking AEs

A stacked AE consists of several layers of SAEs, where the output of every layer is used as the input to the next layer, as shown in Figure 6. By stacking an autoencoder, the deep architectures of input data can be learned efficiently.

In this paper, we used the greedy layer-wise unsupervised pre-training technique to extract features layer by layer in an unsupervised setting. The key advantages of this approach are that it not only yields better local minima by proper initialization of the weights in the DNN and training each autoencoder in sequence but also reduces the generalization limitations of conventional machine learning algorithms. We implemented the greedy layer-wise unsupervised pre-training an SAE on the raw inputs $x^{(k)}$ to extract primary features $h_k^{(1)}$ in the hidden layer, as shown in Figure 6a. The AE uses regularizers to generate a sparse representation of the first layer. The impact of these regularizers can be controlled by adjusting the following parameters (Table 2):

Table 2. Parameters of autoencoders.

Parameter	Value
L2 Weight Regularization (λ)	0.001
Sparsity Regularization (β)	4
Sparsity Proportion (ρ)	0.10

where

- L2 WeightRegularization: weight decay term that helps to prevent overfitting. It should be small;
- Sparsity Regularization: imposes sparsity constraint and controls the weights of the penalty term;
- Sparsity Proportion: controls the average number of activation on a hidden layer. This term is included to ensure that AE works when the number of hidden units is relatively large with respect to the number of input units.

After training the first AE, we then trained the second AE by using the primary features $h_k^{(1)}$ of the first AE as the “raw input” to next SAE to extract secondary features $h_k^{(2)}$ from the dataset, as shown in Figure 6b. The key difference between two AEs was that we use the features that were produced from the first AE as the training input to the second. This greedy layer-wise process is known as “pre-training”.

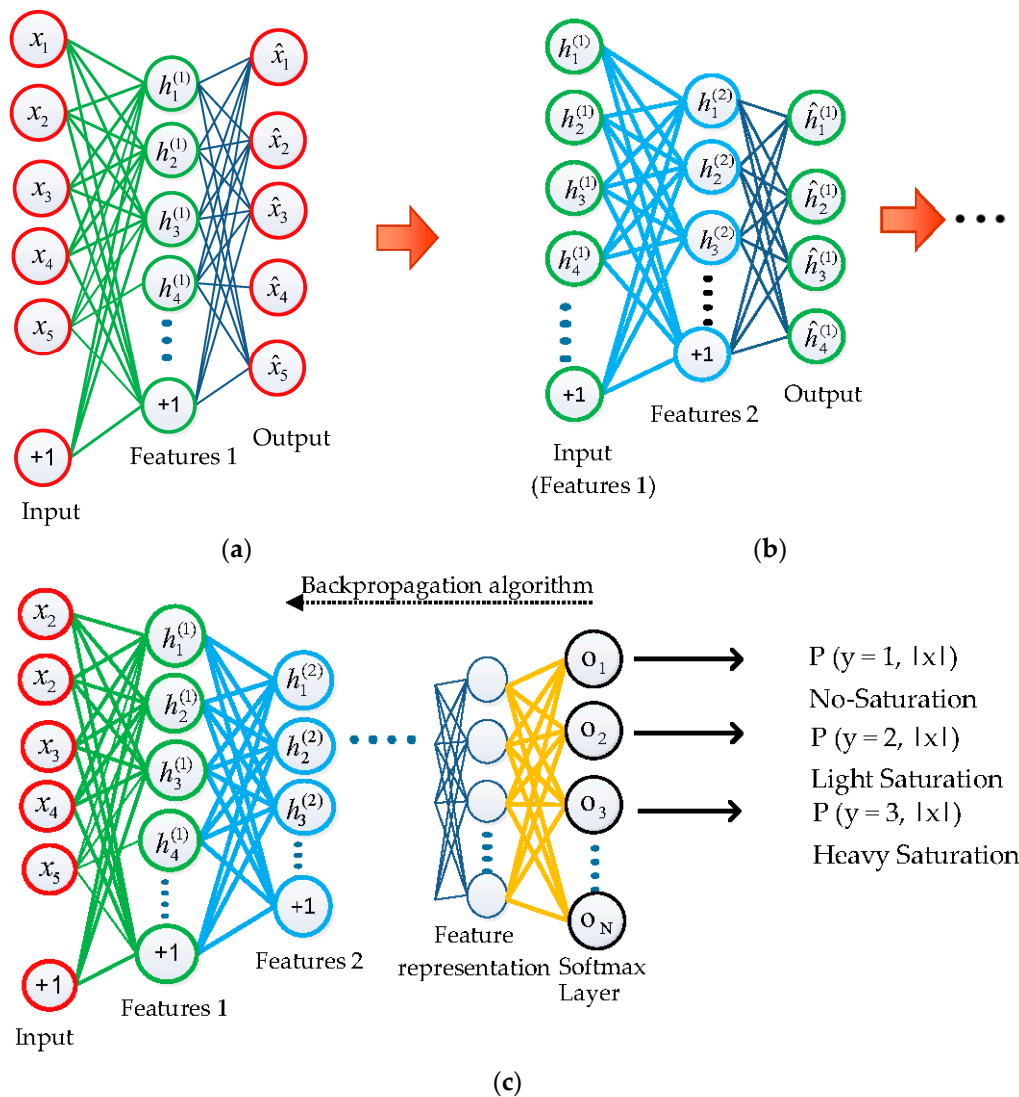


Figure 6. Layer-wise unsupervised pre-training and supervised fine-tuning strategy: (a) 1st stage of autoencoder Learning; (b) 2nd Stage of autoencoder; (c) Supervised fine-tuning.

Furthermore, the extracted features $h_k^{(2)}$ from the second autoencoder were used as raw input to a softmax layer. Finally, we obtained a deep network by training the softmax layer and stacking the AE layers. Consequently, the softmax classifier is capable of classifying the different levels of CT saturation. The features obtained by the stacked autoencoders are high-level patterns. The use of these features significantly improves the classification accuracy.

In addition to unsupervised pre-training, we adopted a fine-tuning strategy to further improve the classification results. We retrained the whole network in a supervised manner using a backpropagation algorithm as shown in Figure 6c. This supervised optimization step is called ‘fine-tuning’. The intuition behind the fine-tuning using backpropagation algorithm is outlined in Table 3.

Table 3. Fine-tuning strategy using backpropagation.

Algorithm	
Step 1	Forward the training data to the input of network. Apply feed-forward pass and calculate activation vectors of every layer until the last output layer.
Step 2	For each node in the output layer, compute network error and measure the difference between the target output value and actual output value.
Step 3	Back-propagate the output error from the softmax layer to the input layer.
Step 4	Update each layer parameters by using gradient descent.
Step 5	Repeat the process from step one to four until algorithm reaches its maximum iterations.

Fine tuning approach is usually adopted in deep learning applications, and it significantly improves the performance of a stacked AEs. It considers all layers of stacked AEs as a single model. Thus, in each iteration, we refine all weights of stacked autoencoder. In addition to fine-tuning, the inclusion of L2 and sparsity regularization in stacked autoencoders prevents overfitting issues and improves generalization of the network. It is worth mentioning that fine-tuning strategy exploits global supervision, which helps network to reach its global minimum. Subsequently, stacked AEs ensure the deep representation of input features with robust feature extraction ability.

5.1. Softmax Classifier for Classification

After fine-tuning, we classify the data by feeding the results from the stacked AE to a softmax regression classifier. The softmax regression classifier is a generalized form of logistic regression and can handle multi-class classification problems. Unlike, logistic regression which can cope only two classes [36], softmax regression classifier can classify multiple classes. As we are dealing with three classes for CT saturation in this paper, so we used softmax regression classifier to resolve the multi-class problem, which outputs multi-class labels rather than performing binary classification. For a given test input x , softmax regression classifier estimates the probability $p(y = j|x)$ of each class $j = 1, \dots, k$, where $k = 3$ represents three different classes. Concretely, the probabilities of each class are calculated as follows:

$$p(y^{(i)} = j|x^{(i)}; \theta) = \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^k e^{\theta_l^T x^{(i)}}} \quad (9)$$

where $y^{(i)}$ is the output class corresponding to input vector $x^{(i)}$ and θ_j represents the parameter vector of softmax regression model for the class j , $j = 1, 2, 3 \dots, k$. Consequently, for the three classes of CT saturation problem, the maximum probability of each class is determined by:

$$Class(x^{(i)}) = \arg_{j=1, \dots, k} \max p(y^{(i)} = j|x^{(i)}; \theta) \quad (10)$$

where $x^{(i)}$ represents the CT saturation class type with the highest probability. The softmax layer outputs probabilities in the range of [0–1]. Therefore, we have converted multi-class targets to binary labels by using one-hot encoding method [27], as shown in Table 4.

Table 4. Binary target values using one-hot encoding.

Class	Labels	Binary Form
1	No saturation	1 0 0
2	Light Saturation	0 1 0
3	Heavy Saturation	0 0 1

6. Results and Discussion

6.1. Testing Strategy and Model Evaluation

We validated the effectiveness of the proposed method and evaluated its performance by conducting a series of simulation tests on the test dataset. The model validation procedure is shown in Figure 7. All simulations were carried out in MATLAB environment with Intel Core™ i7-6700 CPU@3.40 GHz and 8 GB of RAM. We used Neural Network Toolbox™ [37] to train and test the model and assessed the classification performance using a k -fold cross-validation strategy, a confusion matrix and a variety of performance measures, such as the accuracy, precision, recall, specificity, and F1-score, MCC and ROC curves.

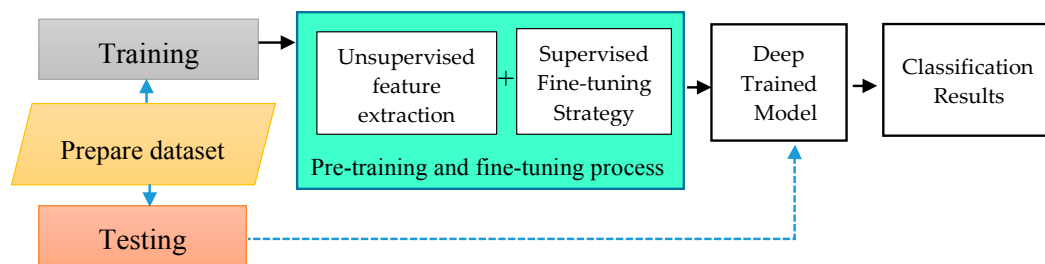


Figure 7. Flowchart showing the procedure used to validate the model. The deep trained model was tested on unseen data.

6.2. Evaluation of Classification Accuracy

We used a standard deep learning technique called k -fold cross-validation to evaluate the classification accuracy and prediction error of the model. In fact, k -fold cross-validation is a model validation method where data is divided into K smaller sets and $(k - 1)$ part of which is used to train the model, and the rest of the part is used as validation data to test the model. This process is repeated on all k subsets so that each subset is used as validation data exactly once, and as training data $(k - 1)$ times. Finally, we averaged the classification accuracy and error rates of the k experiments to obtain a single performance metric. In k -fold cross-validation, k is typically 10, which is a suitable choice for most applications. To reduce the computational time, we used 4-fold cross-validation in this study, as shown in Figure 8.

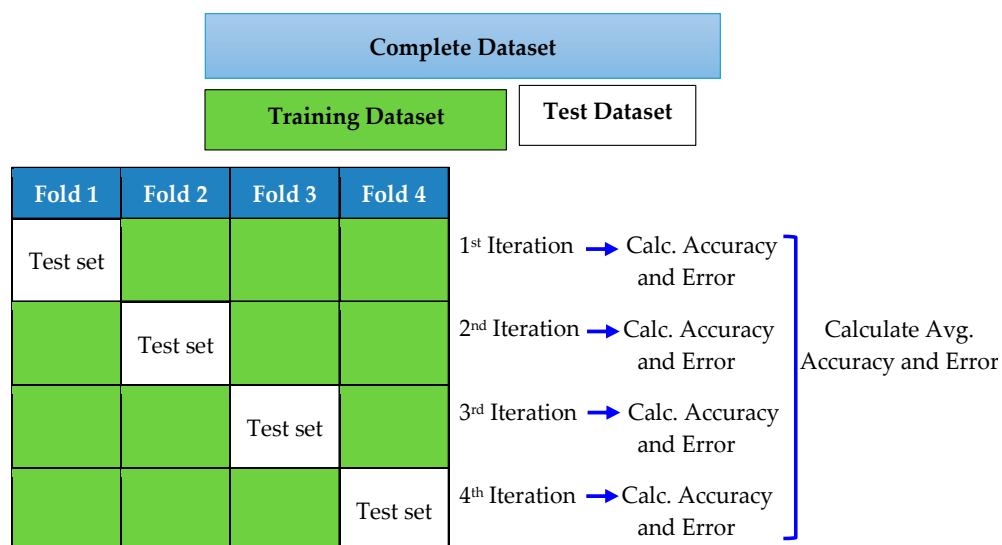


Figure 8. 4-Fold cross-validation scheme.

In Figure 8, the complete dataset is divided into four subsets, and the model is trained and tested four times. In each iteration, one subset was used to test the model arising from training the classifier on the other three subsets. We assessed the average performance of the model by calculating the estimated error for each subset.

6.3. Classification Performance of the Proposed Method

To assess the classification performance of proposed method, we employed k-fold cross-validation and observed the accuracy on each fold. The most meaningful way to evaluate the performance of a prediction model is to assess its ability to classify on unseen data. Thus, we performed a variety of simulations on the test set to validate the performance of our proposed method. A pre-processed CT saturation dataset was used to train and test the model. The classification results obtained from 4-fold cross-validation are summarized in Table 5. The accuracy of each fold is enlisted with their respective error rates.

Table 5. Classification accuracy calculated by 4-fold cross-validation on the test dataset.

K-Fold	Classification Accuracy (%)	Error Rate (%)
K = 1	99.88	0.12
K = 2	99.97	0.03
K = 3	99.90	0.10
K = 4	99.92	0.08
	Average: 99.92	0.08

From Table 5, we can see that, for each subset, the classification accuracy was almost 100%. The maximum error was noticed at iteration 1, which was only 0.12%. Similarly, 0.10% of misclassifications were observed in iteration 3. Finally, we averaged the results of all tests to obtain a single metric for classification performance. The average classification accuracy was 99.92% without overfitting the data. The key benefits of k-fold cross validation are that it not only evaluates the classification performance but also combats the overfitting issues that often occur in DL models.

6.4. Confusion Matrix

To visualize the performance of proposed method, we used a classification matrix known as “confusion matrix”. A confusion matrix summarizes the classification performance and provides a visual representation of actual versus predicted class accuracies. In the confusion matrix, each column represents the predicted class whereas each row represents the actual class respectively. A simple structure of confusion matrix having three class is shown in Table 6.

Table 6. Confusion matrix having 3 classes.

Confusion Matrix		Predicted		
		Class 1	Class 2	Class 3
Actual	Class 1	A	B	C
	Class 2	D	E	F
	Class 3	G	H	I

■ True positives
 ■ True Negatives
 ■ Misclassified cases.

Furthermore, a confusion matrix is a table that reports the counts of true positives, false positives, true negatives and false negatives which are defined as:

- True Positive (TP). The label belongs to the class, and it is correctly predicted.
- False Positive (FP). The label does not belong to the class, but classifier predicted as positive.
- True Negative (TN). The label does not belong to the class, and it is correctly predicted.
- False Negative (FN). The label does belong to the class but is predicted as negative.

To assess the performance of a classification model for each class, we simulated different cases of CT saturation and results are displayed in confusion matrix as shown in Table 7. The results of confusion matrix are then displayed with the waveform in Figure 9 to visualize the number of misclassifications graphically. In confusion matrix, the green cells show the correctly classified cases whereas red cells indicate the number of misclassified cases. The yellow cell in the bottom right of the table shows the overall accuracy. The actual class is taken directly from original test dataset whereas predicted class values are obtained from the classifier on the test dataset. The description of three classes in confusion matrix is given as:

- **Class 1:** No-saturation
- **Class 2:** Light saturation
- **Class 3:** Heavy saturation

Table 7. Confusion matrix results on the test dataset. An overall accuracy is shown in the bottom right. Misclassified cases are pointed with red circle.

Confusion Matrix		Predicted			False Negative (FN)
		Class 1	Class 2	Class 3	
Actual	Class 1	2813	0	0	0
	Class 2	0	819	3	3
	Class 3	0	0	581	0
False Positive (FP)		0	0	3	Overall Accuracy 99.93%

■ True positives ■ True Negatives ■ Misclassified cases.

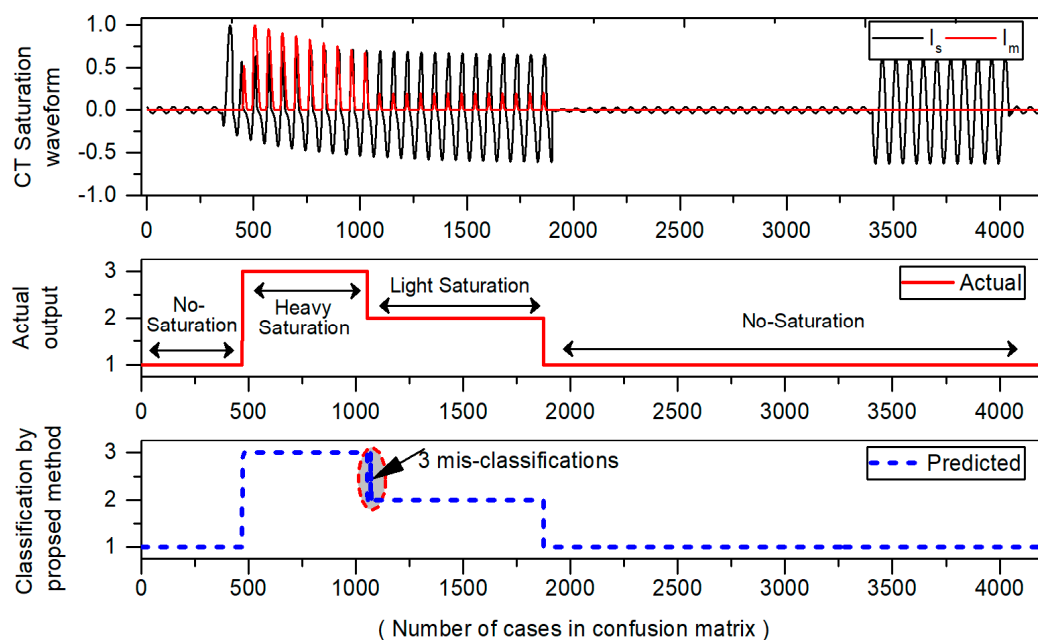


Figure 9. Waveform showing misclassifications of the confusion matrix. All misclassified cases are encircled with red color.

Results of Table 7 show that that class 1 (No-saturation), and class 2 (light saturation) are classified successfully without any misclassifications. However, in the case of class 3 (heavy saturation), it has 584 samples in the test dataset, and only three cases are misclassified as class 2 (light saturation). Furthermore, the number of misclassifications in confusion matrix are also presented with CT saturation waveform as shown Figure 9. Only three misclassifications are found which are encircled

with red color. In Table 7, the three diagonal cells in green color demonstrate the number of correct classifications by the trained network. For example, 2813 cases belonging to class 1 are correctly classified as No-saturation regions. Likewise, 819 and 581 cases belonging to class “2”, “3” are correctly classified as light and heavily saturated regions respectively. Only three misclassifications were noticed, while rest of the cases are correctly classified with a high accuracy of 99.93%. This analysis not only provides us with a way where mistakes are made, but it also provides a feasible way to increase the model performance.

Classification accuracy itself is typically not enough evidence to decide where your designed model is good enough to make robust predictions. The problem with accuracy is that it cannot discriminate among diverse kinds of misclassifications. So, additionally, we computed other performance metrics like precision, recall, specificity, F1-score, MCC [38] by means of the confusion matrix. These terms are normally defined for binary classification problems where the outcome is either “positive” or “negative”. As, we have three classes and dealing with the multi-class problem, so we computed, precision, recall, F1-score, etc. while calculating TN, TP, FP, and FN of each class separately. A cleaner representation with different colors is shown in Table 8, where various performance measures were obtained from the confusion matrix.

Table 8. Computing different performance measures from Confusion Matrix. Precision and recall of each class is calculated.

Confusion Matrix		Predicted			False Negative (FN)	Recall
		Class 1	Class 2	Class 3		
Actual	Class 1	A	B	C	B + C	$A/(A + B + C)$
	Class 2	D	E	F	D + F	$E/(D + E + F)$
	Class 3	G	H	I	G + H	$I/(G + H + I)$
	False Positive (FP)	D + G	B + H	C + F	Overall Accuracy = $A + E + I / (\text{Sum of red and green squares})$	
	Precision	$A/(A + D + G)$	$E/(B + E + H)$	$I/(C + F + I)$		

■ True positives ■ True Negatives ■ Misclassified cases ■ False Positives ■ False Negatives.

From Table 8, it can be clearly seen that out of three classes “1”, “2” and “3”, each class has different FP and FN values. For example, for class “1” FP and FN are highlighted with light blue and pink color. Likewise, light green and dark green boxes show TP and TN for class “1” respectively. Now, with the application of these calculated TN, TP, FP, FN, we extracted the information of other performance measures for each class separately. For instance, performance metrics for class 1 (No-saturation) were calculated in Table 9.

Table 9. Performance metrics extracted from confusion matrix for class 1 (No-saturation).

Performance Metric	Formula	Calculations
Accuracy	$(TP + TN)/(TP + TN + FP + FN)$	$(A + E + I)/(A + E + I + D + G + B + C)$
Precision	$TP/(TP + FP)$	$A/(A + D + G)$
Recall	$TP/(TP + FN)$	$A/(A + B + C)$
Specificity	$TN/(FP + TN)$	$(E + I)/(D + G + E + I)$
F-score	$2TP/(2TP + FP + FN)$	$(2 * A)/(2 * A + D + G + B + C)$
MCC	$TP * TN - FP * FN / \sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}$	$(A * E + I - D + G + B + C) / \sqrt{(A + D + G) * (A + B + C) * (E + I + D + G) * (E + I + B + C)}$

TP: True positives; TN: True negatives; FP: False positives; FN: False negatives. MCC: Matthews Correlation Coefficient. ■ True positives ■ True Negatives ■ Misclassified cases ■ False Positives ■ False Negatives.

As illustrated in Table 9, every performance metrics have different color representation which was extracted from confusion matrix of Table 8. For example, accuracy is the proportion of correctly classified cases to a total number of cases. It was calculated as

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{A+(E+I)}{A+(B+C)+(E+I)+(D+G)} = \frac{2813+(819+581)}{2813+1403+0+0} = 100\% \quad (11)$$

Similarly, other performance metrics have been computed using the color scheme in Table 9. Besides accuracy, we also calculated precision measure as it uncovers the differences in performance that go unobserved when using accuracy. It is the ratio of TP divided by FP and TP. On the other hand, recall is the ratio of TP divided by the number of TP and the FN. It is also termed as Sensitivity. Recall is another popular term which shows classifiers completeness. A small value of recall specifies many false negatives. Extending the previous results of precision and recall, we calculated another performance measure known as the F1-score. It is the weighted average of both precision and recall. Lastly, MCC was calculated, which is a best off-the-shelf evaluating tool for classification tasks. Performance of the proposed method on each class is recorded in Table 10.

Table 10. Performance of proposed method on the test dataset. Different performance metrics were calculated to validate the performance of on each class.

Performance Metric	Class 1 (No-Saturation)	Class 2 (Light-Saturation)	Class 3 (Heavy-Saturation)
Accuracy (%)	100	99.92	99.92
Error/Loss (%)	0	0.07	0.07
Precision (%)	100	100	99.48
Recall (%)	100	99.63	100
Specificity (%)	100	100	99.9
F-score (%)	100	99.81	99.74
MCC (%)	100	99.77	99.70

From Table 10, it is interesting to see that; no misclassifications have been noted in class 1. Only few misclassifications were observed in case of class 2 and 3. Importantly, all these performance metrics were calculated from the confusion matrix. The results of Table 10 indicate that our proposed method achieved significant performance on widely accepted performance metrics that include accuracy, precision, recall, specificity, F-score, and MCC. All CT saturation classes are classified correctly with relative high accuracy.

6.5. Performance Analysis Using Receiver Operating Characteristic (ROC) Curves

ROC Curve is an excellent technique for visualizing the performance based on their classification accuracy. Recently, these curves are heavily used by researchers due to their decision making abilities in classification problems [39]. To extend ROC curve for multiclass classification, we have binarized the outputs. One ROC curve has been drawn per class. Compared to recall, precision, and F1 score, ROC curves solely depict a relationship between false positives rate (FPR) and true positive rate (TPR). The ROC curve is generated by plotting the FPR on the x-axis against the TPR on the y-axis as shown in Figure 10.

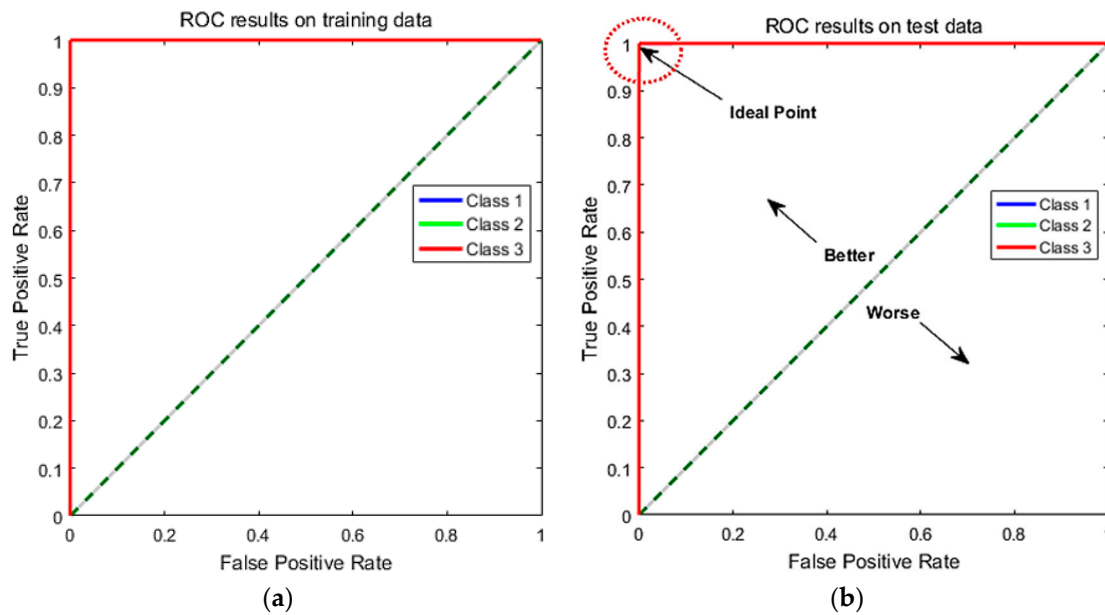


Figure 10. ROC curves for model evaluation. One ROC curve has been drawn per class.: (a) Performance on training dataset; (b) Performance on test dataset while indicating ideal and worst point in ROC space.

In Figure 10, we have shown the obtained results for both training set and test set. The TPR on y-axis describes the number of positive cases that were correctly identified during the test. Whereas, FPR on x-axis defines the number of positive cases that were incorrectly classified as negative during the test. The diagonal line indicates the baseline. The ideal ROC curves should be above the diagonal. In general, models with ROC curves touching the top left corner of the ROC curve indicates better classification performance. The best point on the curve would be $(0, 1)$, where all positive cases are classified correctly, and no negative cases are misclassified as positive. From Figure 10a,b, we can see that both training and test curves touch the upper left corner $(0, 1)$, which indicates a perfect classification on both training and test set. All the three levels of CT saturation (no saturation, light and heavy saturation) were classified successfully. The obtained ROC results validate that suggested method achieved accurate results for CT saturation classification.

6.6. Results With Fine-Tuning Strategy

To observe the influence of fine-tuning strategy on classification performance, we compared the results with fine-tuning and without a fine-tuning strategy. As discussed previously in Section 5, we have implemented the fine-tuning strategy in our work to improve performance even further. The results obtained from both fine-tuning and without fine-tuning are presented in Figures 11 and 12. Remarkably, fine-tuning approach improves classification results by achieving an overall accuracy of 99.9%. Misclassification rate was only 0.1%. The classification accuracy without using the fine-tuning strategy was 83.8%, of which 13.8% was the misclassification rate. The obtained classification results and error rates are presented in Table 11. From the table results, we can see that fine-tuning reduces the training error and improves the classification performance of DNN. The fine-tuning strategy not only improves the network performance but also contributes to obtain discriminative features for CT saturation classification. Moreover, the results displayed in Figure 12 show that ROC curves without fine-tuning are far away from the upper left corner $(0, 1)$. This indicates a poor classification performance with high misclassification rate. However, ROC curves with fine-tuning touch the ideal point $(0, 1)$, which indicates the perfect classification.



Figure 11. Performance of fine-tuning strategy is displayed with confusion matrix: (a) Results without fine-tuning; (b) Results with the fine-tuning approach.

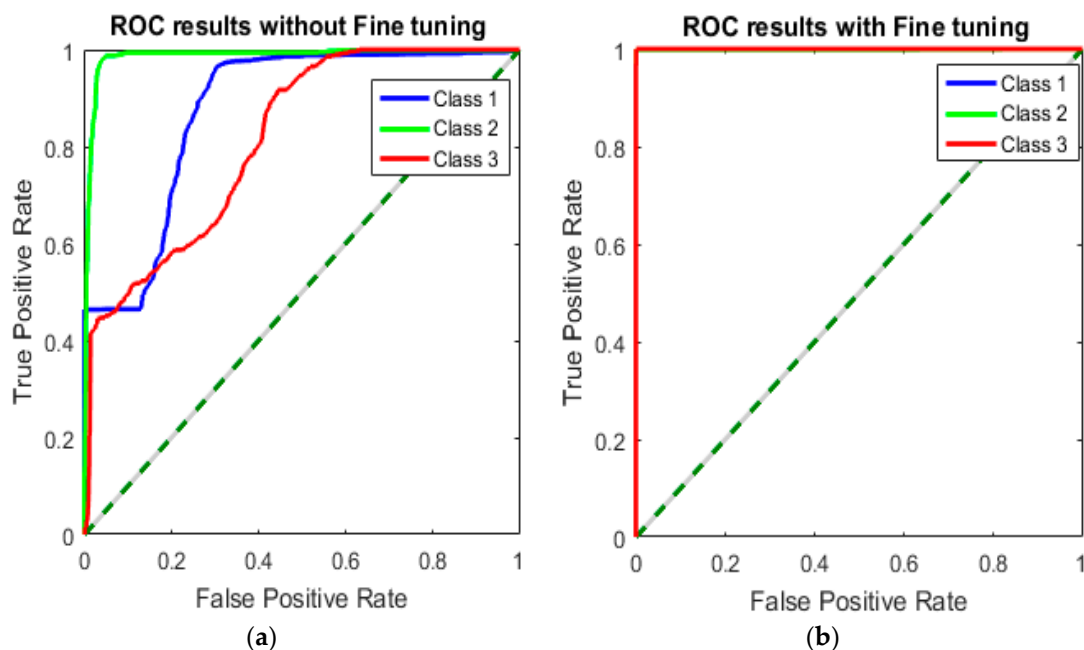


Figure 12. Performance comparison of fine-tuning with ROC curves: (a) Results without fine-tuning; (b) Results with the fine-tuning approach.

Table 11. Classification performance with and without a fine-tuning strategy.

	Without Fine-Tuning	With Fine-Tuning
Classification Accuracy	83.8%	99.9%
Misclassification rate	13.2%	0.1%

The presented results in Figures 11 and 12, concludes that fine-tuning strategy brought a significant improvement in classification accuracy and achieved state-of-the-art performance for CT saturation classification.

7. Comparison with Traditional Methods

To conduct a fair comparison with traditional methods, we experimented the same dataset on three widely used methods known as shallow neural networks (SNN), support vector machines (SVM) and random forests (RF). Three state-of-the-art methods SNN, SVM, and RF were compared with the proposed method. It is important to mention here that we used best hyperparameters to train the models. In case of neural networks, we performed a random search [40] across models for hyper-parameter optimization whereas for SVM we used grid search [41] technique. We trained all methods on the same training dataset and later evaluated their performance on the test dataset to calculate their classification accuracies. Each method was evaluated using standard testing method k -fold cross-validation and variety of performance metrics such as accuracy, precision, recall, specificity, F1-score, MCC and ROC curves with their respective area under the curve (AUC) values. In SNN, the fully-connected feed-forward hidden layer is used. Moreover, for SVM results, we used MATLAB function “fitcecoc” from the Statistics and Machine Learning Toolbox™ (MathWorks). A one-versus-all approach along with “RBF Kernel” was used as they deliver more stable results as compared to other kernel functions. An RF method [42] which is based on ensemble learning and decision trees are also used to analyze and compare the classification results.

7.1. Experiments Using K-Fold Cross-Validation

In this study, we used 4-fold cross-validation to compare the classification performance of proposed method with traditional classification methods. We computed the classification accuracy of each method with respect to their corresponding folds. The classification performance of all methods are displayed in Figure 13. We can see that performance of the proposed method is better on each fold as compared to the traditional classification methods using the same inputs. Furthermore, Figure 13a shows that SNN has worst classification rate of 92.1% on first iteration ($k = 1$). However, there is a slight increase of accuracy on second iteration ($k = 2$). SVM achieved better performance as compared to SNN on each fold having less misclassification rate. Importantly, RF shows a promising performance compared to both SNN and SVM. However, proposed method dominates the other three approaches with the higher classification accuracy and least error rate. In addition, we also computed the misclassification rate of each method which is shown in Figure 13b.

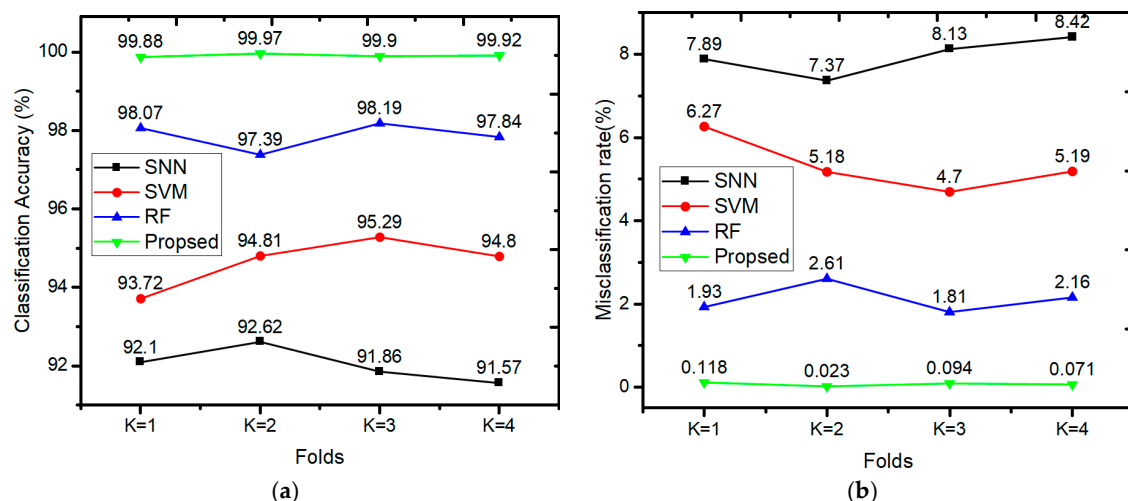


Figure 13. Performance comparison of different classification methods on each fold: (a) Classification accuracy on each fold; (b) Error (misclassification rate) on each fold.

After computing accuracy on each fold, we averaged the accuracy of all folds to obtain a final measure of performance. Inspected mean accuracy and standard deviation of each method is tabulated

in Table 12. We calculated the accuracy on both the training and test dataset. It can be clearly seen that the performance of the SNN, SVM, and RF-based methods are lower as compared to the proposed method. The SNN method achieved a classification accuracy of 92.4% on the test set, with a standard deviation of 0.86%. However, for the SVM-based method, the average accuracy was 94.8%, with a standard deviation of 0.54%. The misclassification rate of SVM was 5.4%, compared to 7.7% of SNN. RF shows a relatively high performance with average accuracy of 97.8% and standard deviation of 0.35%. Although all methods did well on the test set, comparison results indicate that proposed method has better classification ability compared to other methods with an accuracy of 99.9% and a standard deviation 0.0390%. It is worth noting that proposed method yields superior performance for CT saturation classification having an error rate of only 0.1%.

Table 12. Average classification accuracy of different methods after $k = 4$ experiments. Their corresponding standard deviations are also listed.

Network	Classification Accuracy		Classification Error		Standard Deviation
Type	Train (%)	Test (%)	Train (%)	Test (%)	Test (%)
SNN	92.4	92.3	7.6	7.7	± 0.8600
SVM	94.8	94.6	5.2	5.4	± 0.5496
RF	99.3	97.8	0.7	2.2	± 0.3562
Proposed method	100	99.9	0.0	0.1	± 0.0390

7.2. Comparison Using Standard Performance Metrics

Although, accuracy metric has significantly helped us to compare algorithms and construct robust experimental designs. But, classification accuracy itself is not competent enough to make a final decision about the performance of the model. In such situation, we further extend our analysis to find other performance measures such as precision, recall, F-score, etc. A systematic analysis of different methods on each class of dataset has been carried out in Table 13. The performance for each class along with their mean performance is specified. All the performance measures are calculated from their corresponding confusion matrices.

Table 13 summarizes the results obtained from different methods for each class of CT saturation. It can be seen from the table that our proposed method achieved optimal performance with mean values ranging from 0.99 to 1. Both precision and recall (also known as sensitivity) were used to measure the exactness and quality of the model. For instance, if SNN model predicts 2718 out of 4216 cases as “Not-saturated” and only 2684 of those 2718 are truly “Not-saturated,” then the precision is $2684/2718 = 0.9875$. Likewise, for recall, if 2699 out of 4216 cases are truly “not-saturated” and model correctly predicted 2684 of them. Then, the recall will be $2684/2831 = 0.9479$. High values of precision indicate that an algorithm largely returned more related results than unrelated ones, while large values of recall show that an algorithm achieved maximum relevant results.

Importantly, recall only focuses on true positive rates or sensitivity, so there is need of another performance measure which calculates the true negative rate. Specificity is ideal performance measure to compute the accuracy of negative cases. For example, if 100 cases with “No Saturation” are tested, and 96 of them appeared as a negative result, then the test has 96% specificity. Furthermore, to benefit from calculations of precision and recall, we have calculated F-score of each class. F-score uses both precision and recall to estimate the score and gives its best value at 1 and worst at 0. In Table 13, SNN model achieved 0.9910 F-score for “No-saturation” class whereas 0.6392 F-score was achieved for “Heavy saturation” class. Apart from all these, MCC is another metric that can be used to evaluate a classifier’s performance. MCC equal to 1 shows a perfect prediction. The MCC of proposed method is 0.9995 for “No-saturation class” which indicates an accurate prediction. Better accuracy results and lowest error rate on all performance metrics indicate the superiority of proposed approach as compared to other methods.

Table 13. Systematic analysis of different methods on each class of CT saturation. Classification performance is analyzed by taking the mean of each performance metrics.

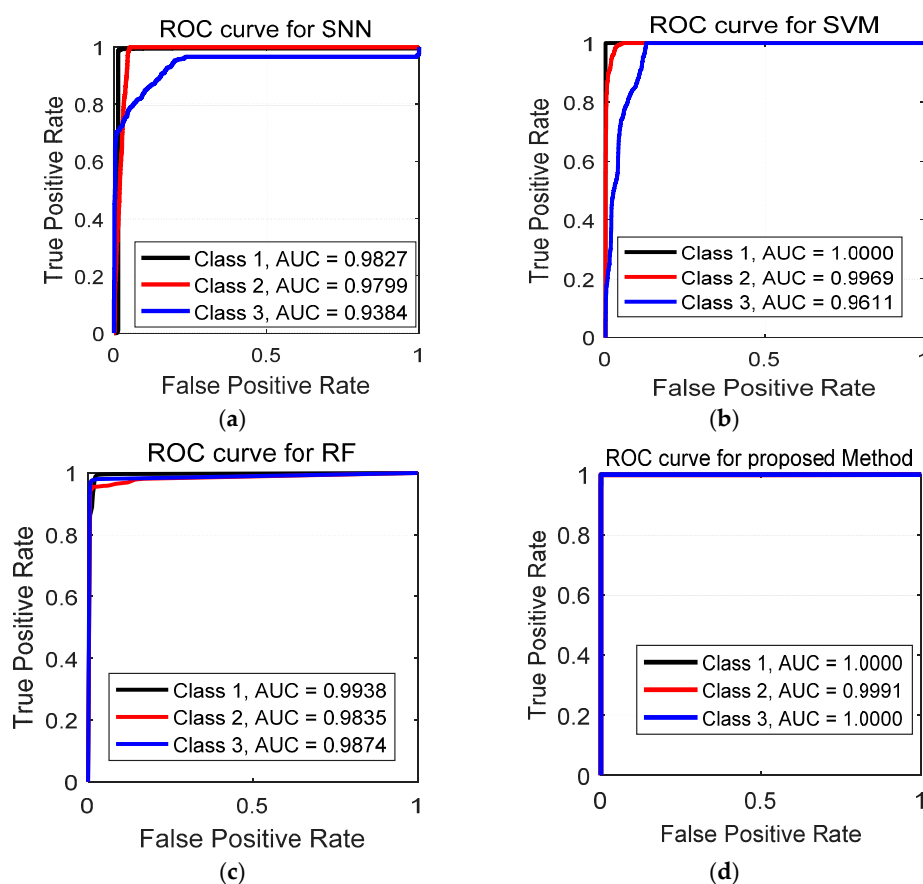
Performance Metrics	Classes	Methods			
		SNN	SVM	RF	Proposed Method
Accuracy	No-Saturation	0.9884	1.0000	0.9855	1.0000
	Heavy-Saturation	0.9191	0.9613	0.9822	0.9992
	Light saturation	0.9250	0.9613	0.9891	0.9992
	Mean	0.9442	0.9742	0.9856	0.9995
Error	No-Saturation	0.0116	0.0000	0.0145	0.0000
	Heavy-Saturation	0.0809	0.0387	0.0178	0.0007
	Light saturation	0.0750	0.0387	0.0109	0.0007
	Mean	0.0558	0.0258	0.0144	0.0005
Precision	No-Saturation	0.9875	1.0000	0.9883	1.0000
	Heavy-Saturation	0.9902	0.9232	0.9661	1.0000
	Light saturation	0.7351	0.8663	0.9477	0.9948
	Mean	0.9043	0.9298	0.9674	0.9983
Recall	No-Saturation	0.9944	1.0000	0.9900	1.0000
	Heavy-Saturation	0.4719	0.7852	0.9444	0.9963
	Light saturation	1.0000	0.9552	0.9714	1.0000
	Mean	0.8221	0.9135	0.9686	0.9988
Specificity	No-Saturation	0.9776	1.0000	0.9765	1.0000
	Heavy-Saturation	0.9992	0.9895	0.9917	1.0000
	Light saturation	0.9054	0.9629	0.9918	0.9991
	Mean	0.9607	0.9841	0.9867	0.9997
F-Score	No-Saturation	0.9910	1.0000	0.9892	1.0000
	Heavy-Saturation	0.6392	0.8487	0.9552	0.9981
	Light saturation	0.8473	0.9086	0.9594	0.9974
	Mean	0.8258	0.9191	0.9679	0.9985
MCC	No-Saturation	0.9747	1.0000	0.9674	1.0000
	Heavy-Saturation	0.6525	0.8302	0.9442	0.9977
	Light saturation	0.8158	0.8858	0.9532	0.9970
	Mean	0.8143	0.9054	0.9549	0.9982

7.3. Comparison With ROC Curves and AUC Values

To further validate the classification performance of the proposed method, we compared the proposed method with other methods (SNN, SVM and RF) using ROC curves and area under the curve (AUC) values. The AUC is a valuable statistic for the performance evaluation of classification algorithm. It is a plot of sensitivity (true positive rate) and specificity (true negative rate) that describes the inherent validity of diagnostic tests. Normally, there is a misconception that ROC curves are only limited to two classes but many researchers [43,44] employed ROC analysis for multi-class classification. In our study, we extended ROC analysis for three classes by pairwise comparisons. We carried out pairwise comparison using one class vs. all classification approach. It means one class is compared with rest of the classes. The experimental results of three different methods with respect to their classes are shown in Table 14 and Figure 14 respectively.

Table 14. Performance comparison of the different methods using ROC-AUC curve.

Performance Metrics	Methods				
	Classes	SNN	SVM	RF	Proposed Method
AUC	No-Saturation	0.9827	1.000	0.9938	1.000
	Heavy-Saturation	0.9799	0.9969	0.9835	0.9991
	Light saturation	0.9384	0.9611	0.9874	1.000
	Mean	0.9670	0.9860	0.9882	0.9997

**Figure 14.** Displays the ROC-AUC curve of different classification models: (a) Performance of SNN on each class; (b) ROC-AUC curve on SVM experiments; (c) RF performance on each class; (d) Performance of proposed method.

Generally, the value of AUC lies between 0 to 1.0, and the larger AUC value specifies a higher classification accuracy. A perfect method has an AUC of 1.0. In Figure 14a, we can see that SNN produced an AUC of 0.9827, 0.9799 and 0.9384. AUC of Class 3 is 0.9384, which is significantly lower than other two classes. Meanwhile, Figure 14b indicates that SVM performs better than SNN. AUC for each class is 1.000, 0.9969, and 0.9611 respectively. Compared to SNN and SVM, RF achieved a prominent performance. Likewise, Figure 14c shows the AUC of the proposed method where each class has AUC almost 1. This illustrates the superiority of the proposed method as compared to other three methods in terms of distinguishing the various levels of CT saturation.

8. Conclusions

This study was motivated by the need for an efficient CT saturation classification method using deep learning approach based on DNN and AEs. We validated the proposed approach by

conducting a variety of simulation tests and comparisons. The validation results reveal that proposed method correctly classifies the different levels of CT saturation with high accuracy and has superior performance to traditional classification methods. The four principal contributions of this paper are as follows. First, we investigated the use of a DNN with AEs for CT saturation problem. This enabled us to exploit the deep architectures of DNNs. Second, we implemented unsupervised feature extraction method using AEs. Third, we performed experiments using a greedy unsupervised layer-wise training approach which resolves the problem of local minima by initializing the weights in a region near a good local minimum, thus bringing better generalization in the deep network. Finally, we adopted a fine-tuning strategy that uses backpropagation to fine-tune the weights of the pre-trained DNN. This significantly improves the classification performance. The results show that our proposed method yields high classification accuracy for CT saturation and has the advantages of simplicity, easier interpretation, model complexity reduction, and better learning capability.

Deep learning methods have reached new heights of success in many difficult tasks, especially in classification problems. However, to fully exploit the power of this methodology, we need to develop new ways of thinking to address a wide range of technical issues. In future, we intend to implement our approach in real time situations with GPU to enable robust decision-making with respect to CT saturation classification. In this study, we only focused on accurate classification of CT saturation using deep learning (DL) approach. Compensation of CT saturation (reconstruction of distorted CT secondary current) using unsupervised feature engineering is our future work. Apart from DNN and AEs, application of long short-term memory (LSTM) networks and convolutional neural network (CNN) for CT saturation classification will also be a significant future work.

Acknowledgments: This work was supported in part by “Human Resources Program in Energy Technology” of the Korea Institute of Energy Technology Evaluation and Planning (KETEP), granted financial resource from the Ministry of Trade, Industry & Energy, Republic of Korea (No. 20174030201790). This research was also supported in part by Korea Electric Power Corporation. (Grant number: R17XA05-2).

Author Contributions: Muhammad Ali prepared the manuscript and completed the methodology and the simulations. Soon-Ryul Nam supervised the study and coordinated the main theme of this paper. Dae-Hee Son and Sang-Hee Kang discussed the results and implications, and commented on the manuscript. All of the authors read and approved the final manuscript.

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

References

1. C37.110-2007—IEEE Guide for the Application of Current Transformers Used for Protective Relaying Purposes; Revision of Std C37.110-1996; IEEE: Piscataway, NJ, USA, 2008; pp. 1–90. [[CrossRef](#)]
2. IEEE Guide for Protective Relay Applications to Power System Buses; IEEE Std C37.234-2009; IEEE: Piscataway, NJ, USA, 2009; pp. C1–115. [[CrossRef](#)]
3. Gil, M.; Abdoos, A.A. Intelligent busbar protection scheme based on combination of support vector machine and S-transform. *IET Gener. Transm. Distrib.* **2017**, *11*, 2056–2064. [[CrossRef](#)]
4. Hajipour, E.; Vakilian, M.; Sanaye-Pasand, M. Current-Transformer Saturation Compensation for Transformer Differential Relays. *IEEE Trans. Power Deliv.* **2015**, *30*, 2293–2302. [[CrossRef](#)]
5. Kang, Y.C.; Ok, S.H.; Kang, S.H.; Crossley, P.A. Design and evaluation of an algorithm for detecting current transformer saturation. *IEE Proc. Gener. Transm. Distrib.* **2004**, *151*, 27–35. [[CrossRef](#)]
6. Schettino, B.M.; Duque, C.A.; Silveira, P.M. Current-Transformer Saturation Detection Using Savitzky-Golay Filter. *IEEE Trans. Power Deliv.* **2016**, *31*, 1400–1401. [[CrossRef](#)]
7. Segatto, E.C.; Coury, D.V. A differential relay for power transformers using intelligent tools. *IEEE Trans. Power Syst.* **2006**, *21*, 1154–1162. [[CrossRef](#)]
8. Tripathy, M. Power Transformer Differential Protection Based on Neural Network Principal Component Analysis, Harmonic Restraint and Park’s Plots. *Adv. Artif. Intell.* **2012**, *2012*, 930974. [[CrossRef](#)]
9. Koley, E.; Shukla, S.K.; Ghosh, S.; Mohanta, D.K. Protection scheme for power transmission lines based on SVM and ANN considering the presence of non-linear loads. *IET Gener. Transm. Distrib.* **2017**, *11*, 2333–2341. [[CrossRef](#)]

10. Shah, A.M.; Bhalja, B.R. Fault discrimination scheme for power transformer using random forest technique. *IET Gener. Transm. Distrib.* **2016**, *10*, 1431–1439. [[CrossRef](#)]
11. Pan, J.; Vu, K.; Hu, Y. An efficient compensation algorithm for current transformer saturation effects. *IEEE Trans. Power Deliv.* **2004**, *19*, 1623–1628. [[CrossRef](#)]
12. Saleh, S.A.; Rahman, M.A. Modeling and protection of a three-phase power transformer using wavelet packet transform. *IEEE Trans Power Deliv.* **2005**, *20*, 1273–1282. [[CrossRef](#)]
13. Zadeh, M.R.D.; Zhang, Z. A new DFT-based current phasor estimation for numerical protective relaying. In Proceedings of the 2014 IEEE PES General Meeting | Conference Exposition, National Harbor, MD, USA, 27–31 July 2014; p. 1.
14. Esmail, E.M.; Elkalashy, N.I.; Kawady, T.A.; Taalab, A.M.I.; Lehtonen, M. Detection of Partial Saturation and Waveform Compensation of Current Transformers. *IEEE Trans. Power Deliv.* **2015**, *30*, 1620–1622. [[CrossRef](#)]
15. Nam, S.R.; Park, J.Y.; Kang, S.H.; Kezunovic, M. Phasor Estimation in the Presence of DC Offset and CT Saturation. *IEEE Trans. Power Deliv.* **2009**, *24*, 1842–1849. [[CrossRef](#)]
16. Schmidhuber, J. Deep Learning in Neural Networks: An Overview. *Neural Netw.* **2015**, *61*, 85–117. [[CrossRef](#)] [[PubMed](#)]
17. Hinton, G.E.; Salakhutdinov, R.R. Reducing the Dimensionality of Data with Neural Networks. *Science* **2006**, *313*, 504–507. [[CrossRef](#)] [[PubMed](#)]
18. Arel, I.; Rose, D.C.; Karnowski, T.P. Deep Machine Learning—A New Frontier in Artificial Intelligence Research [Research Frontier]. *IEEE Comput. Intell. Mag.* **2010**, *5*, 13–18. [[CrossRef](#)]
19. Haji, M.M.; Vahidi, B.; Hosseini, S.H. Current Transformer Saturation Detection Using Gaussian Mixture Models. *J. Appl. Res. Technol.* **2013**, *11*, 79–87. [[CrossRef](#)]
20. Rebizant, W.; Bejmert, D. Current-Transformer saturation detection with genetically optimized neural networks. *IEEE Trans. Power Deliv.* **2007**, *22*, 820–827. [[CrossRef](#)]
21. Moradi, A.R.; Alinejad Beromi, Y.; Kiani, K.; Moravej, Z. Artificial Intelligence Based Approach for Identification of Current Transformer Saturation from Faults in Power Transformers. *Int. J. Smart Electr. Eng.* **2014**, *3*, 37–46.
22. Segatto, E.C.; Coury, D.V. A power transformer protection with recurrent ANN saturation correction. In Proceedings of the IEEE Power Engineering Society General Meeting, San Francisco, CA, USA, 16 June 2005; Volume 2, pp. 1341–1346.
23. Sheng, Y.; Rovnyak, S.M. Decision trees and wavelet analysis for power transformer protection. *IEEE Trans. Power Deliv.* **2002**, *17*, 429–433. [[CrossRef](#)]
24. Manitoba HVDC Center: ‘PSCAD/EMTDC User’s Manual’; HVDC Center: Winnipeg, MB, Canada, 2016.
25. Lucas, J.R.; McLaren, P.G.; Keerthipala, W.W.L.; Jayasinghe, R.P. Improved simulation models for current and voltage transformers in relay studies. *IEEE Trans. Power Deliv.* **1992**, *7*, 152–159. [[CrossRef](#)]
26. Lin, X.; Zou, L.; Tian, Q.; Weng, H.; Liu, P. A series multiresolution morphological gradient-based criterion to identify CT saturation. *IEEE Trans. Power Deliv.* **2006**, *21*, 1169–1175. [[CrossRef](#)]
27. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006.
28. Qu, Y.; He, M.; Deutsch, J.; He, D. Detection of Pitting in Gears Using a Deep Sparse Autoencoder. *Appl. Sci.* **2017**, *7*, 515. [[CrossRef](#)]
29. Ryu, S.; Noh, J.; Kim, H. Deep Neural Network Based Demand Side Short Term Load Forecasting. *Energies* **2016**, *10*, 3. [[CrossRef](#)]
30. Backpropagation Algorithm—Ufldl. Available online: http://ufldl.stanford.edu/wiki/index.php/Backpropagation_Algorithm (accessed on 9 August 2017).
31. Yang, Z.-X.; Wang, X.-B.; Zhong, J.-H. Representational Learning for Fault Diagnosis of Wind Turbine Equipment: A Multi-Layered Extreme Learning Machines Approach. *Energies* **2016**, *9*, 379. [[CrossRef](#)]
32. He, P.; Jia, P.; Qiao, S.; Duan, S. Self-Taught Learning Based on Sparse Autoencoder for E-Nose in Wound Infection Detection. *Sensors* **2017**, *17*, 2279. [[CrossRef](#)] [[PubMed](#)]
33. Li, C.; Ding, Z.; Zhao, D.; Yi, J.; Zhang, G. Building Energy Consumption Prediction: An Extreme Deep Learning Approach. *Energies* **2017**, *10*, 1525. [[CrossRef](#)]
34. Ng, A. Sparse autoencoder. *CS294A Lect. Notes* **2011**, *72*, 1–19.
35. Kullback, S.; Leibler, R.A. On information and sufficiency. *Ann. Math. Stat.* **1951**, *22*, 79–86. [[CrossRef](#)]
36. Park, B.; Hur, J. Accurate Short-Term Power Forecasting of Wind Turbines: The Case of Jeju Island’s Wind Farm. Section 2.1. *Energies* **2017**, *10*, 812. [[CrossRef](#)]

37. MATLAB. *Statistics and Machine Learning Toolbox*; The MathWorks Inc.: Natick, MA, USA, 2016.
38. Sokolova, M.; Japkowicz, N.; Szpakowicz, S. Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. In *AI 2006: Advances in Artificial Intelligence; Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1015–1021.
39. Ji, L.; Wu, J.; Zhou, Y.; Hao, L. Using Trajectory Clusters to Define the Most Relevant Features for Transient Stability Prediction Based on Machine Learning Method. *Energies* **2016**, *9*, 898. [[CrossRef](#)]
40. Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
41. Hsu, C.W.; Chang, C.C.; Lin, C.J. *A Practical Guide to Support Vector Classification*; Department of Computer Science National Taiwan University: Taipei, Taiwan, 2003.
42. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [[CrossRef](#)]
43. Djemal, R.; Bazyed, A.G.; Belwafi, K.; Gannouni, S.; Kaaniche, W. Three-Class EEG-Based Motor Imagery Classification Using Phase-Space Reconstruction Technique. *Brain Sci.* **2016**, *6*, 36. [[CrossRef](#)] [[PubMed](#)]
44. Fawcett, T. An introduction to ROC analysis. *Pattern Recognit. Lett.* **2006**, *27*, 861–874. [[CrossRef](#)]



© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).