



MURAT YILMAZ 
FERHAT OZGUR CATAK
ENSAR GUL 

SENSOR-BASED CYBERATTACK DETECTION IN CRITICAL INFRASTRUCTURES USING DEEP LEARNING ALGORITHMS

Abstract

The technology that has evolved with innovations in the digital world has also caused an increase in many security problems. Day by day, the methods and forms of cyberattacks are becoming more complicated; therefore, their detection has become more difficult. In this work, we have used datasets that have been prepared in collaboration with the Raymond Borges and Oak Ridge National Laboratories. These datasets include measurements of the Industrial Control Systems related to chewing attack behavior. These measurements include synchronized measurements and data records from Snort and relays with a simulated control panel. In this study, we developed two models using these datasets. The first is a model we call the DNN model, which was build using the latest deep learning algorithms. The second model was created by adding the AutoEncoder structure to the DNN model. All of the variables used when developing our models were set parametrically. A number of variables such as the activation method, the number of hidden layers in the model, the number of nodes in the layers, and the number of iterations were analyzed to create the optimum model design. When we run our model with optimum settings, we obtained better results than those found in related studies. The learning speed of the model has a 100% accuracy rate, which is also entirely satisfactory. While the training period of the dataset containing about 4 thousand different operations lasts for about 90 seconds, the developed model completes the learning process at a level of milliseconds to detect new attacks. This increases the applicability of the model in the real-world environment.

Keywords

engineering, critical infrastructure, industrial systems, information security, cyber security, cyberattack detections

Citation

Computer Science 20(2) 2019: 213–244

1. Introduction

Nowadays, rapidly developing technology has replaced human power in many places. Especially in large industrial systems such as critical infrastructures that cannot be managed by human power, security problems could occur in their computer systems. Their computer systems were installed many years ago and are vulnerable to almost every current attack.

While the definition of critical infrastructure systems varies from country to country, it is generally defined as systems or entities that are necessary for the maintenance of vital social processes, security, and economic security [8].

Critical infrastructures can generally be classified as agriculture and food, water, public health and safety, emergency services, government, defense industry base, information and telecommunication, energy, transportation, banking and finance, industry and manufacturing, and mail and shipping. Each of these sectors has critical infrastructures, and interruptions of transactions or damage to these infrastructures may be a vital element in people's living standards that can have a life-saving impact and can even threaten human life. However, since they have large infrastructural investments, they can cause serious economic losses and weakness of states.

Critical infrastructures are not isolated systems. Anyone of these infrastructures that interact with each other will cause damage to the chain. For all of these reasons, the protection of these systems is vital [4]. The STUXNET attack (one example of an attack on critical infrastructure) has demonstrated this effect. This attack, which caused Iran to take its nuclear development activity back two years, caused only economic losses. However, it also revealed the possibility of the nuclear plant being damaged in the worse-case scenario, which could lead to a disaster in that area [5]. Day by day, the methods and forms of cyberattacks are becoming more complicated; therefore, their detection has become more difficult. According to a 2018 report by FireEye, a cyberattack's detection time is 101 days globally, 175 days for the EMEA region, and 498 days for the APAC region.

The purpose of this study is to reduce and automate the perception of increasing the dwell time of cyber threats by using deep learning algorithms.

1.1. Contribution

- We worked with an up-to-date dataset created in 2014 by Mississippi State University and Oak Ridge National Laboratory. The dataset is a reliable dataset used in many scientific research projects [10,12].
- We used the latest in-depth learning algorithms and technologies: AutoEncoder model, Tensorflow 1.4.0, Keras 2.1.1, Sklearn 0.19.1, Scipy 1.0.0, Numpy 1.13.3, and Pandas 0.21.0.
- We obtained better results than the classification performance obtained in the related studies that we examined.

2. Related Work

Various studies are used in this critical infrastructure dataset. Detailed explanations of the papers listed above are given in the coming subsections.

2.1. Developing hybrid intrusion detection system using data mining for power systems

An IDS (Intrusion Detection System) is a system that automates cyberattack detection. Many cyberattacks have similar characteristics. A signature is extracted from these properties; these signatures for similar cyberattacks simplify the work of IDS systems. In this study, feature-based systems are used along with signature-based systems. Normal system interruptions such as maintenance, normal operation, and cyberattack situations are taught to the IDS, and the IDS system's capabilities are developed for a possible cyberattack; these are aimed at detecting previously unseen cyberattacks such as zero-day clearance. The accuracy rate of these systems is 90.4%; an operation with an accuracy rate of 90.4% means that nearly 10 attacks cannot be detected per 100 attacks. When it is considered that this number is much higher in living systems, it is expected that such automatic systems will work with near-zero error.

2.2. Classification of disturbances and cyberattacks in power systems using heterogeneous time-synchronized data

It is mentioned that [11] has three contributions in the literature. First, they point out that cyberattacks (which show themselves as a normal system interruption) can be recognized and discerned from system outages and achieve better results than the work done with. Their approach uses less memory usage when compared to traditional machine learning algorithms. In this work, authors use the common path mining algorithm, thus indicating that they use less memory than traditional data mining methods. Third – it learns by separating the set of algorithm scenarios and the dataset that they use. Here, a common path-finding algorithm is developed to prevent over-adaptation. The algorithm used in the study had better results than algorithms like Random forest and JRip, and it performed more poorly than algorithm applications like Adaboost + Jrip. The accuracy rate of the algorithm for the multi-class problem is 93%. This rate is not suitable when considering the number of today's attacks.

2.3. Specification-based intrusion detection framework for cyber-physical environment in electric power system

In this article, a method has been proposed to detect scabby attacks on power systems or scans on physical breaks. This method reveals a specification-based intrusion detection system by monitoring the records of many devices, including existing cyber intrusion detection systems, simulated control panels, snort-, relay-, and network-monitoring software, and control room computers. Depending on the different control

data, causal relationships between cyberattacks and interruptions are established. In this work, the probabilistic network for generating IDS rules provides a method for mapping such data to a Bayesian network. The Bayesian network is known for its powerful heuristic for modeling interdependencies between variables as well as its ability to graphically show causal relationships from data and workflow records [6]. Based on a specific control scheme, this work illustrates the process of building such a Bayesian network and deriving different system scenarios. With the proposed method, the IDS tracks the transmission line; if there is any interruption in the power grid, the operator may be informed that it is caused by a system problem or cyberattack. The accuracy of the method used in the study is not explained with numeric results; however, it seems that the method is more effective against physical effects. It has been stated that the development of the IDS system based on the specification in the proposed method may be expensive and require expertise. This is not the preferred case either.

2.4. Machine learning for power system disturbance and cyberattack discrimination

In [2], a network specialist has developed a policy to decide whether an interruption is due to a cyberattack or a natural event. It is difficult for a person to distinguish between cyberattacks and natural phenomena because they have the same effect. For this reason, an algorithm that can be used as a decision support tool to automate this work has been studied. In this study, it was determined that the methods of teaching a machine are sufficient to establish the relationship between the measurements in the power system and the causes of interruption. The classification performance of various machine learning methods are evaluated, and the accuracy level of the proposed method is given. In the study, many algorithms were used; the performances of these algorithms are classified. When we analyzed the results of the study, the Adaboost + jRipper algorithm showed the best accuracy rate (99.1%). In our study, it is obvious that we get better results than the existing algorithms when we think that the ratio is 100% for binary- and triple-class task and 99.8% for multi-class task.

3. Preliminaries

3.1. Datasets

In collaboration with Raymond Borges and Justin Beaver of Oak Ridge National Laboratories, Adhikari et al. created three datasets that include measurements related to an electric transmission system's normal, control-maintenance, and cyberattack behaviors. Measurements in the dataset include synchrophasor measurements and data logs from Snort (a simulated control panel) and relays. The features information in the dataset is detailed in Table 1.

The design of the lab environment for the dataset used in the study is shown in Figure 1.

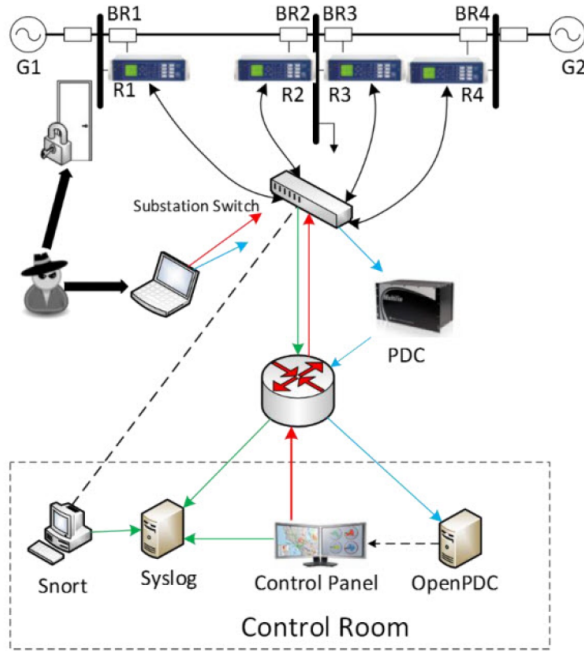


Figure 1. Dataset lab design

The datasets used in the study are classified according to their output labels. The label distribution of the binary-class dataset is given in Table 2. The label distribution of the triple-class dataset is given in Table 3. The label distribution of the multi-class dataset is given in Table 4.

Table 1
Features in dataset

Features	Description
PA1:VH – PA3:VH	Phase A - C Voltage Phase Angle
PM1: V – PM3: V	Phase A - C Voltage Phase Magnitude
PA4:IH – PA6:IH	Phase A - C Current Phase Angle
PM4: I – PM6: I	Phase A - C Current Phase Magnitude
PA7:VH – PA9:VH	Pos. - Neg. - Zero Voltage Phase Angle
PM7: V – PM9: V	Pos. - Neg. - Zero Voltage Phase Magnitude
PA10:VH – PA12:VH	Pos. - Neg. - Zero Current Phase Angle
PM10: V – PM12: V	Pos. - Neg. - Zero Current Phase Magnitude
F	Frequency for relays
DF	Frequency Delta (dF/dt) for relays
PA:Z	Appearance Impedance for relays
PA:ZH	Appearance Impedance Angle for relays
S	Status Flag for relays

Table 2
Event Scenarios (Binary)

Scenario	Description	Number of Rows
0	Normal operation	1100
1	Attack	3866

Table 3
Event Scenarios (Triple)

Scenario	Description	Number of Rows
-1	Natural Events	927
0	Normal operation	173
1	Attack	3866

Table 4
Event Scenarios (Multi)

Scenario	Description	Rows
-2	Fault from Line (Natural Events)	264
-1	Line maintenance (Natural Events)	663
0	Regular Operation (Normal operation)	173
1	Data Injection – SLG fault replay (Attack)	569
2	Command injection against single relay to R1, R2, R3, R4 (Attack)	346
3	Command injection against single relay to R1 and R2 or R3 and R4 (Attack)	106
4	Disabling relay function – single relay disabled & fault (Attack)	1675
5	Disabling relay function – two relays disabled & fault (Attack)	898
6	Disabling relay function – two relay disabled & line maintenance (Attack)	272

3.2. Autoencoder

AutoEncoder is a type of Neural Network that first compresses multi-dimensional data into a hidden area and then reconstructs the data from the compressed hidden area. AutoEncoders have three type layers; input layer, hidden area, or hidden layer and output layer. The number of nodes in the input layer is equal to the number of nodes in the output layer because AutoEncoder is meant to reconstruct the intended data; this is called the input layer and hidden area encoder. The encoder allows one

to reduce multi-dimensional data to a smaller size. The decoder is called the decoder between the hidden area and output layer. The decoder layer tries to reconfigure the input instance by increasing the size of the compressed hidden layer [1,9]. The dataset we use includes voltage measurements and consists of fractional numeric values that will not affect the end result. In our work, we used the AutoEncoder model to avoid being slowed down with these fractional numeric values and avoiding false positives in the learning process. The purpose of the AutoEncoder model is to increase the accuracy of the system by deleting unnecessary detail. For example, the detail in a three-dimensional image is unnecessary and must be reduced to two dimensions only for a shape-separating operation. Another example is the most commonly used noise-reduction methods. Noise-canceling images can be obtained when used with the AutoEncoder model [13].

An autoencoder always consists of two parts (the encoder and decoder) that can be defined as transitions ϕ and ψ ; the calculations of ϕ and ψ are given in the following equations:

$$\begin{aligned}\phi : X &\rightarrow F \\ \psi : F &\rightarrow X\end{aligned}\tag{1}$$

where

$$\arg \min_{\phi, \psi} \|X - (\phi \circ \psi)X\|^2\tag{2}$$

In the simplest case where there is one hidden layer, the encoder stage of an autoencoder takes input $\mathbf{x} \in \mathbb{R}^d$ and maps it to $\mathbf{z} \in \mathbb{R}^p$. \mathbf{z} is calculated from the following equation:

$$\mathbf{z} = \sigma(W\mathbf{x} + \mathbf{b})\tag{3}$$

\mathbf{z} is usually referred to as code, latent variable, or latent representation. Here, σ is an element-wise activation function such as a sigmoid function or the rectified linear unit. W is the weight matrix, and \mathbf{b} is the bias vector. After that, the decoder stage of the autoencoder maps \mathbf{z} to reconstruction $\hat{\mathbf{x}}$ of the same shape as \mathbf{x}

$$\hat{\mathbf{x}} = \sigma(W\mathbf{x} + \mathbf{b})\tag{4}$$

The AutoEncoder part that we added to our model is shown in Figure 2.

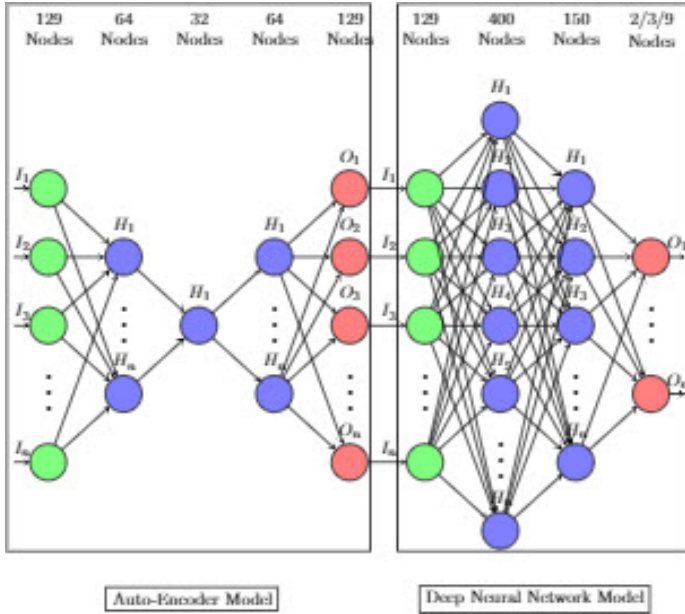


Figure 2. Deep Neural Network Model With AutoEncoder

3.3. Deep learning

A class of machine learning techniques where many layers of information-processing stages in hierarchical architectures are exploited for unsupervised feature learning and pattern analysis/classification. The essence of deep learning is to compute the hierarchical features or representations of the observational data where the higher-level features or factors [3]. The data set we use contains voltage information. It is not possible for a person to interpret this data coming from 128 sensors and create an attack pattern; however, Deep Neural Network approaches can easily do this, which is impossible for people. For this reason, we used Deep Neural Network Algorithms in our study, which have a very high ability to analyze nonlinear data.

3.4. Experimental results

Two separate models were designed in the study. The first is the DNN model, which is a model in which the activation methods, hidden layer, and number of nodes are changed dynamically. The second is the AutoEncoder model. In this model, the input values of the DNN model are not directly read from the file. The inputs were first passed through the AutoEncoder model and simplified, and these outputs were given as DNN model inputs. The results of these two studies are separately analyzed and presented below in detail.

3.4.1. DNN model results

In the model, there are three designs called Layer Mode 1, Layer Mode 2, and Layer Mode 3. The number of layers and nodes in these designs are given in Table 5.

Table 5
Nodes in layers

Layer Mode	DNN Model
1	129(input)-400-150-2/3/9(Output)
2	129(input)-400-650-400-150-2/3/9(Output)
3	129(input)-400-650-900-650-400-150-2/3/9(Output)

There are too many parameters in the operation. To provide a more meaningful representation of the outputs of the study, the results of the data classes called binary, triple, multi are presented separately below.

3.4.2. Binary data class results

The model is run separately for 15 datasets in the binary data class with binary tags with 0-Normal and 1-Attack. The Layer Mode 1 design results of the binary data class are given in detail in Table 6.

Table 6
Binary Data Class Results for Layer Mode 1

	Epochs	Accuracy	Precision	Recall	F Score
tanh	200	1.0	1.0	1.0	1.0
softplus	200	1.0	1.0	1.0	1.0
softsign	200	1.0	1.0	1.0	1.0
linear	200	1.0	1.0	1.0	1.0

An analysis of the data in the Layer Mode 1 design shows that the four most successful results are obtained with the 'tanh,' 'softplus,' 'softsign,' and 'linear' activation methods. The confusion matrix of these four methods is given in Table 7.

Table 7
Confusion Matrix for Layer Mode 1 with Test Dataset

	P	N		P	N		P	N		P	N
P	303	0	P	304	0	P	310	0	P	317	0
N	0	711	N	0	710	N	0	704	N	0	697
tanh			softplus			softsign			linear		

When we run our model with the binary data class, we obtained a 100% accuracy rate after nearly 30 epochs.

We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-150-2(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Tanh
- Epochs : 200
- Output Activation Method : sigmoid
- Output Loss Method : binary_crossentropy

With these parameters, the training duration of our model is 1:55 for 4055 different processes. The test duration is approximately 280 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.28 milliseconds. The Layer Mode 2 design results of the binary data class are given in detail in Table 8.

Table 8
Binary Data Class Results for Layer Mode 2

	Epochs	Accuracy	Precision	Recall	F Score
tanh	300	1.0	1.0	1.0	1.0
softsign	300	1.0	1.0	1.0	1.0

An analysis of the data in the Layer Mode 2 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 9.

Table 9
Confusion Matrix for Layer Mode 2 with Test Dataset

	P	N
P	292	0
N	0	722

tanh

	P	N
P	304	0
N	0	710

softsign

When we run our model with the binary data class, we obtained a 100% accuracy rate after nearly 50 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-400-150-2(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 300
- Output Activation Method : sigmoid
- Output Loss Method : binary_crossentropy

With these parameters, the training duration of our model is 11:00 for 4055 different processes. The test duration is approximately 514 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.5 milliseconds.

The Layer Mode 3 design results of the binary data class are given in detail in Table 10.

Table 10
Binary Data Class Results for Layer Mode 3

	Epochs	Accuracy	Precision	Recall	F Score
relu	500	0.999	0.999	0.999	0.999
softsign	500	0.999	0.999	0.999	0.999
linear	500	0.999	0.999	0.999	0.999

An analysis of the data in the Layer Mode 3 design shows that the three most successful results are obtained with the 'relu,' 'softsign,' and 'linear' activation methods.

The confusion matrix of these three methods is given in Table 11.

Table 11
Confusion Matrix for Layer Mode 3 with Test Dataset

	P	N		P	N		P	N
P	295	1	P	312	1	P	297	0
N	0	718	N	0	701	N	1	716
	relu			softsign			linear	

When we run our model with the binary data class, we obtained a 99.9% accuracy rate after nearly 80 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-900-650-400-150-2(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 500
- Output Activation Method : sigmoid
- Output Loss Method : binary_crossentropy

With these parameters, the training duration of our model is 40:00 for 4055 different processes. The test duration is approximately 1 second for 1014 different processes, and the detection time of the new incoming attack is 0.98 milliseconds.

3.4.3. Triple Data Class Results

The model is run separately for 15 datasets in triple data class with triple tags with -1 Natural, 0-Normal, 1-Attack. The Layer Mode 1 design results of the triple data class are given in detail in Table 12.

Table 12
Triple Data Class Results for Layer Mode 1

	Epochs	Accuracy	Precision	Recall	F Score
tanh	200	1.0	1.0	1.0	1.0
softsign	200	1.0	1.0	1.0	1.0

An analysis of the data in the Layer Mode 1 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 13.

Table 13
Confusion Matrix for Layer Mode 1 with Test Dataset

	-1	0	1		-1	0	1
-1	221	0	0	-1	241	0	0
0	0	70	0	0	0	67	0
1	0	0	723	1	0	0	706
	tanh				softsign		

When we run our model with the triple data class, we obtained a 100% accuracy rate after nearly 60 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-150-3(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 200
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 1:53 for 4055 different processes. The test duration is approximately 241 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.23 milliseconds.

The Layer Mode 2 design results of the triple data class are given in detail in Table 14.

Table 14
Triple Data Class Results for Layer Mode 2

	Epochs	Accuracy	Precision	Recall	F Score
tanh	300	0.999	0.999	0.999	0.999
softsign	300	0.999	0.999	0.999	0.999

An analysis of the data in the Layer Mode 2 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 15.

Table 15
Confusion Matrix for Layer Mode 2 with Test Dataset

	-1	0	1		-1	0	1
-1	257	0	1	-1	237	0	0
0	0	54	0	0	0	68	0
1	0	0	702	1	1	0	708
	tanh				softsign		

When we run our model with the triple data class, we obtained a 99.9% accuracy rate after nearly 100 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-400-150-3(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 300
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 9:46 for 4055 different processes. The test duration is approximately 751 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.74 milliseconds.

The Layer Mode 3 design results of the triple data class are given in detail in Table 16.

Table 16
Triple Data Class Results for Layer Mode 3

	Epochs	Accuracy	Precision	Recall	F Score
tanh	500	0.999	0.999	0.999	0.999
softsign	500	0.999	0.999	0.999	0.999

An analysis of the data in the Layer Mode 3 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 17.

Table 17
Confusion Matrix for Layer Mode 3 with Test Dataset

	-1	0	1		-1	0	1
-1	233	0	0	-1	258	1	0
0	0	66	0	0	0	55	0
1	0	1	714	1	0	0	700
	tanh				softsign		

When we run our model with the triple data class, we obtained a 99.9% accuracy rate after nearly 120 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-900-650-400-150-3(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 500
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 41:00 for 4055 different processes. The test duration is approximately 1 second for 1014 different processes, and the detection time of the new incoming attack is 1 millisecond.

3.4.4. Multi Data Class Results

The model is run separately for 15 datasets in the multi data class with multi tags with -2 Natural(Fault From Line), -1 Natural(Line maintenance), 0-Normal, 1-Attack (Data Injection), 2-Attack (Command Injection), 3-Attack (Command Injection), 4-Attack (Disabling relay function), 5-Attack (Disabling relay function), and 6-Attack (Disabling relay function).

The Layer Mode 1 design results of the multi data class are given in detail in Table 18.

Table 18
Multi Data Class Results for Layer Mode 1

	Epochs	Accuracy	Precision	Recall	F Score
tanh	200	0.991	0.991	0.991	0.991
softsign	200	0.998	0.998	0.998	0.998

An analysis of the data in the Layer Mode 1 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 19.

Table 19
Confusion Matrix for Layer Mode 1 with Test Dataset

	-2	-1	0	1	2	3	4	5	6
-2	50	0	0	0	0	0	0	0	0
-1	0	179	0	1	0	0	0	0	0
0	0	0	61	0	0	0	0	0	0
1	0	0	0	134	0	0	0	0	0
2	0	0	0	0	81	0	0	0	0
3	0	0	0	0	0	43	1	0	0
4	0	0	0	0	0	1	231	6	0
5	0	0	0	0	0	0	0	163	0
6	0	0	0	0	0	0	0	0	63

tanh

	-2	-1	0	1	2	3	4	5	6
-2	72	0	0	0	0	0	0	0	0
-1	0	183	0	0	0	0	0	0	0
0	0	0	63	0	0	0	0	0	0
1	0	0	0	122	0	0	0	0	0
2	0	0	0	0	82	0	0	0	0
3	0	0	0	0	0	50	0	0	0
4	0	0	0	0	0	0	219	0	0
5	0	0	0	0	0	0	1	135	0
6	0	0	0	0	0	0	0	0	86

softsign

When we run our model with the multi data class, we obtained a 99.8% accuracy rate after nearly 160 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-150-9(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 200
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 2:04 for 4055 different processes. The test duration is approximately 332 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.32 milliseconds.

The Layer Mode 2 design results of the multi data class are given in detail in Table 20.

Table 20
Multi Data Class Results for Layer Mode 2

	Epochs	Accuracy	Precision	Recall	F Score
tanh	300	0.989	0.989	0.989	0.989
softsign	300	0.995	0.995	0.995	0.995

An analysis of the data in the Layer Mode 2 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 21.

Table 21
Confusion Matrix for Layer Mode 2 with Test Dataset

	-2	-1	0	1	2	3	4	5	6
-2	77	1	0	0	0	0	0	0	0
-1	1	171	0	2	0	0	0	0	0
0	0	0	63	0	0	0	0	0	0
1	0	0	0	131	1	0	0	0	0
2	0	0	0	0	70	0	0	0	0
3	0	0	0	0	0	42	0	0	0
4	0	0	0	0	0	1	242	3	0
5	0	0	0	0	0	0	2	156	0
6	0	0	0	0	0	0	0	0	51

tanh

	-2	-1	0	1	2	3	4	5	6
-2	85	0	0	0	0	0	0	0	0
-1	0	187	1	1	0	0	0	0	0
0	0	0	65	0	0	0	0	0	0
1	0	0	0	111	1	0	0	0	0
2	0	0	0	0	69	0	0	0	0
3	0	0	0	0	0	40	0	0	0
4	0	0	0	0	0	0	224	2	0
5	0	0	0	0	0	0	0	156	0
6	0	0	0	0	0	0	0	0	72

softsign

When we run our model with the multi data class, we obtained a 99.5% accuracy rate after nearly 200 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-400-150-9(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 300
- Output Activation Method : softmax
- Output Loss Method : categorical.crossentropy

With these parameters, the training duration of our model is 10:25 for 4055 different processes. The test duration is approximately 912 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.9 milliseconds.

The Layer Mode 3 design results of the multi data class are given in detail in Table 22.

Table 22
Multi Data Class Results for Layer Mode 3

	Epochs	Accuracy	Precision	Recall	F Score
tanh	500	0.994	0.994	0.994	0.994
softsign	500	0.993	0.993	0.993	0.993

An analysis of the data in the Layer Mode 3 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 23.

Table 23
Confusion Matrix for Layer Mode 3 with Test Dataset

	-2	-1	0	1	2	3	4	5	6
-2	58	1	0	0	0	0	0	0	0
-1	0	194	0	0	0	0	0	0	0
0	0	0	73	0	0	0	0	0	0
1	0	0	1	125	0	0	0	0	0
2	0	0	0	0	66	0	0	0	0
3	0	0	0	0	0	36	0	0	0
4	0	0	0	0	0	0	233	1	0
5	0	0	0	0	0	0	3	142	2
6	0	0	0	0	0	0	0	0	81

tanh

	-2	-1	0	1	2	3	4	5	6
-2	72	0	0	0	0	0	0	0	0
-1	0	170	0	0	0	0	0	0	0
0	0	0	66	0	0	0	0	0	0
1	0	0	0	97	0	1	0	0	0
2	0	0	0	1	82	0	0	0	0
3	0	0	0	0	0	51	1	0	0
4	0	0	0	0	0	1	233	0	0
5	0	0	0	0	0	0	3	135	1
6	0	0	0	0	0	0	0	0	101

softsign

When we run our model with the multi data class, we obtained a 99.4% accuracy rate after nearly 300 epochs.

We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-900-650-400-150-9(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Tanh
- Epochs : 500
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 43:40 for 4055 different processes. The test duration is approximately 1.1 seconds for 1014 different processes, and the detection time of the new incoming attack is 1 millisecond.

3.5. AutoEncoder Results

For the best results of the AutoEncoder model, the activation methods were designed and run as sequence parameters as in the Deep Neural Network model.

The results were analyzed, and the best result with the DNN model was obtained by using the linear activation method; the linear activation method was used in the encode and decode layers of the AutoEncoder model.

In the model, there are three designs: Layer Mode 1, Layer Mode 2, and Layer Mode 3.

The numbers of the layers and nodes in these designs are given in Table 24.

Table 24
Nodes in Layers with AutoEncoder

Layer Mode	AutoEncoder Model	DNN Model
1	129-64-32-64-129	129-400-150-2/3/9
2	129-64-32-64-129	129-400-650-400-150-2/3/9
3	129-64-32-64-129	129-400-650-900-650-400-150-2/3/9

There are too many parameters in the operation. To provide a more meaningful representation of the outputs of the study, the results of the data classes called binary, triple, multi are presented separately below.

3.5.1. Binary Data Class Results with AutoEncoder Model

The model is run separately for 15 datasets in the binary data class with binary tags with 0-Normal and 1-Attack.

The Layer Mode 1 design results of the binary data class are given in detail in Table 25.

Table 25
Binary Data Class Results for Layer Mode 1

	Epochs	Accuracy	Precision	Recall	F Score
tanh	200	1.0	1.0	1.0	1.0
softplus	200	0.999	0.999	0.999	0.999
softsign	200	0.999	0.999	0.999	0.999

An analysis of the data in the Layer Mode 1 design shows that the three most successful results are obtained with the 'tanh,' 'softplus,' and 'softsign' activation methods.

The confusion matrix of these three methods is given in Table 26.

Table 26
Confusion Matrix for Layer Mode 1 with Test Dataset

	P	N		P	N		P	N
P	303	0	P	307	1	P	302	0
N	0	711	N	0	706	N	1	711
tanh			softplus			softsign		

When we run our model with the binary data class, we obtained a 100% accuracy rate after nearly 75 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- AE Nodes in the layers : 129(Input)-64-32-64-129(Output)
- DNN Optimizer Algorithm : SGD
- DNN, Nodes in the layers : 129(Input)-400-150-2(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Tanh
- Epochs : 200
- Output Activation Method : sigmoid
- Output Loss Method : binary_crossentropy

With these parameters, the training duration of our model is 4:00 for 4055 different processes. The test duration is approximately 341 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.33 milliseconds.

The Layer Mode 2 design results of the binary data class are given in detail in Table 27.

Table 27
Binary Data Class Results for Layer Mode 2

	Epochs	Accuracy	Precision	Recall	F Score
tanh	300	1.0	1.0	1.0	1.0
softsign	300	0.999	0.999	0.999	0.999
linear	300	0.999	0.999	0.999	0.999

An analysis of the data in the Layer Mode 2 design shows that the three most successful results are obtained with the 'tanh,' 'softsign,' and 'linear' activation methods.

The confusion matrix of these three methods is given in Table 28.

Table 28
Confusion Matrix for Layer Mode 2 with Test Dataset

	P	N		P	N		P	N
P	303	0	P	322	1	P	308	1
N	0	711	N	0	691	N	0	705
tanh			softsign			linear		

When we run our model with the binary data class, we obtained a 100% accuracy rate after nearly 75 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- AE Nodes in the layers : 129(Input)-64-32-64-129(Output)
- DNN Optimizer Algorithm : SGD
- DNN, Nodes in the layers : 129(Input)-400-650-400-150-2(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Tanh
- Epochs : 300
- Output Activation Method : sigmoid
- Output Loss Method : binary_crossentropy

With these parameters, the training duration of our model is 11:24 for 4055 different processes. The test duration is approximately 844 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.83 milliseconds.

The Layer Mode 3 design results of the binary data class are given in detail in Table 29.

Table 29
Binary Data Class Results for Layer Mode 3

	Epochs	Accuracy	Precision	Recall	F Score
tanh	500	1.0	1.0	1.0	1.0
relu	500	0.998	0.998	0.998	0.998
softsign	500	0.998	0.998	0.998	0.998

An analysis of the data in the Layer Mode 3 design shows that the three most successful results are obtained with the 'tanh,' 'relu,' and 'softsign' activation methods.

The confusion matrix of these three methods is given in Table 30.

Table 30
Confusion Matrix for Layer Mode 3 with Test Dataset

	P	N
P	316	0
N	0	698

tanh

	P	N
P	299	2
N	0	713

relu

	P	N
P	338	2
N	0	674

softsign

When we run our model with the binary data class, we obtained a 100% accuracy rate after nearly 120 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- AE Nodes in the layers : 129(Input)-64-32-64-129(Output)
- DNN Optimizer Algorithm : SGD
- DNN, Nodes in the layers : 129(Input)-400-650-900-650-400-150-2(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Tanh
- Epochs : 500
- Output Activation Method : sigmoid
- Output Loss Method : binary_crossentropy

With these parameters, the training duration of our model is 45:14 for 4055 different processes. The test duration is approximately 1.5 seconds for 1014 different processes, and the detection time of the new incoming attack is 1.4 milliseconds.

The accuracy and loss graphs of the model are shown in Figure 3.

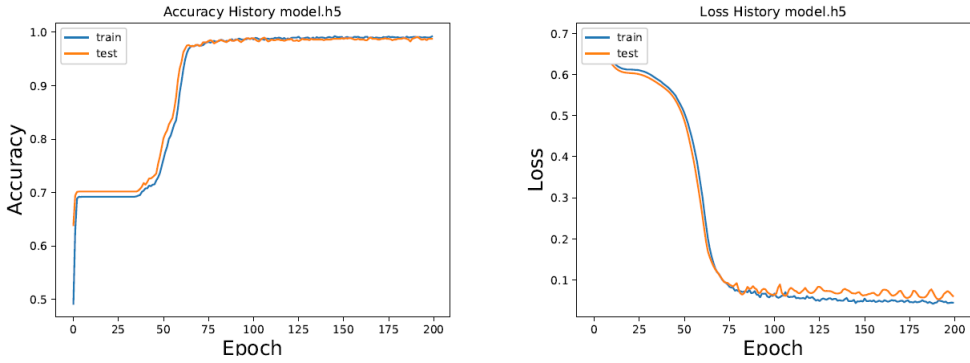


Figure 3. Binary classifier model accuracy and loss history.

3.5.2. Triple Data Class Results with AutoEncoder Model

The model is run separately for 15 datasets in the triple data class with triple tags with -1 Natural, 0-Normal, and 1-Attack.

The Layer Mode 1 design results of the triple data class are given in detail in Table 31.

Table 31
Triple Data Class Results for Layer Mode 1

	Epochs	Accuracy	Precision	Recall	F Score
tanh	200	0.999	0.999	0.999	0.999
softsign	200	0.999	0.999	0.999	0.999

An analysis of the data in the Layer Mode 1 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 32.

Table 32
Confusion Matrix for Layer Mode 1 with Test Dataset

	-1	0	1
-1	250	0	0
0	0	53	0
1	1	0	710

tanh

	-1	0	1
-1	233	0	0
0	0	69	0
1	0	1	711

softsign

When we run our model with the triple data class, we obtained a 99.9% accuracy rate after nearly 150 epochs.

We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-150-3(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Tanh
- Epochs : 200
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 2:52 for 4055 different processes. The test duration is approximately 413 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.4 milliseconds.

The Layer Mode 2 design results of the triple data class are given in detail in Table 33.

Table 33
Triple Data Class Results for Layer Mode 2

	Epochs	Accuracy	Precision	Recall	F Score
relu	300	0.999	0.999	0.999	0.999
softsign	300	0.997	0.997	0.997	0.997

An analysis of the data in the Layer Mode 2 design shows that the two most successful results are obtained with the 'relu' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 34.

Table 34
Confusion Matrix for Layer Mode 2 with Test Dataset

	-1	0	1
-1	244	0	1
0	0	71	0
1	0	0	698

relu

	-1	0	1
-1	241	1	0
0	0	73	0
1	1	1	697

softsign

When we run our model with the triple data class, we obtained a 99.7% accuracy rate after nearly 150 epochs.

We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-400-150-3(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 300
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 14:33 for 4055 different processes. The test duration is approximately 745 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.73 milliseconds.

The Layer Mode 3 design results of the triple data class are given in detail in Table 35.

Table 35
Triple Data Class Results for Layer Mode 3

	Epochs	Accuracy	Precision	Recall	F Score
relu	500	0.998	0.998	0.998	0.998
softsign	500	0.998	0.998	0.998	0.998
linear	500	1.0	1.0	1.0	1.0

An analysis of the data in the Layer Mode 3 design shows that the three most successful results are obtained with the 'relu,' 'softsign,' and 'linear' activation methods.

The confusion matrix of these three methods is given in Table 36.

Table 36
Confusion Matrix for Layer Mode 3 with Test Dataset

	-1	0	1		-1	0	1		-1	0	1
-1	229	0	1	-1	233	0	1	-1	250	0	0
0	0	65	0	0	0	58	0	0	0	53	0
1	1	0	718	1	1	0	721	1	0	0	711
	relu				softsign				linear		

When we run our model with the triple data class, we obtained a 100% accuracy rate after nearly 80 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-900-650-400-150-3(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Linear
- Epochs : 500
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 49:43 for 4055 different processes. The test duration is approximately 1.7 seconds for 1014 different processes, and the detection time of the new incoming attack is 1.6 milliseconds.

The accuracy and loss graphs of the model are shown in Figure 4.

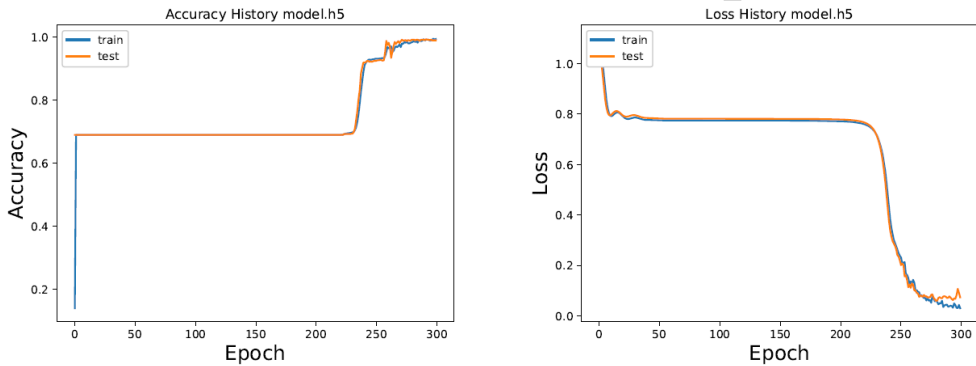


Figure 4. Triple classifier model accuracy and loss history.

3.5.3. Multi Data Class Results with AutoEncoder Model

The model is run separately for 15 datasets in the multi data class with multi tags with -2 Natural(Fault From Line), -1 Natural(Line maintenance), 0-Normal, 1-Attack (Data Injection), 2-Attack (Command Injection), 3-Attack (Command Injection), 4-Attack (Disabling relay function), 5-Attack (Disabling relay function), and 6-Attack (Disabling relay function).

The Layer Mode 1 design results of the multi data class are given in detail in Table 37.

Table 37
Multi Data Class Results for Layer Mode 1

	Epochs	Accuracy	Precision	Recall	F Score
tanh	200	0.975	0.975	0.975	0.975
softsign	200	0.983	0.983	0.983	0.983

An analysis of the data in the Layer Mode 1 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 38.

Table 38
Confusion Matrix for Layer Mode 1 with Test Dataset

	-2	-1	0	1	2	3	4	5	6
-2	53	2	0	0	0	0	0	0	0
-1	3	161	1	2	0	0	0	0	0
0	0	0	81	1	0	0	0	0	0
1	0	0	0	121	0	0	0	0	0
2	0	0	0	1	76	0	0	0	0
3	0	0	0	1	5	41	1	0	0
4	0	0	0	1	0	1	230	2	0
5	0	0	0	0	0	0	3	153	0
6	0	0	0	0	0	0	0	0	73

tanh

	-2	-1	0	1	2	3	4	5	6
-2	79	2	0	0	0	0	0	0	0
-1	2	176	0	0	0	0	0	0	0
0	0	0	52	1	0	0	0	0	0
1	0	0	1	112	0	0	0	0	0
2	0	0	0	1	73	1	0	0	0
3	0	0	0	0	4	36	2	0	0
4	0	0	0	1	0	0	252	0	0
5	0	0	0	0	0	0	1	141	1
6	0	0	0	0	0	0	0	0	76

softsign

When we run our model with the multi data class, we obtained a 98.3% accuracy rate after nearly 200 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-150-9(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 200
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 4:35 for 4055 different processes. The test duration is approximately 576 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.56 milliseconds.

The Layer Mode 2 design results of the multi data class are given in detail in Table 39.

Table 39
Multi Data Class Results for Layer Mode 2

	Epochs	Accuracy	Precision	Recall	F Score
tanh	300	0.987	0.987	0.987	0.987
softsign	300	0.977	0.977	0.977	0.977

An analysis of the data in the Layer Mode 2 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 40.

Table 40
Confusion Matrix for Layer Mode 2 with Test Dataset

	-2	-1	0	1	2	3	4	5	6
-2	63	0	0	0	0	0	0	0	0
-1	1	184	0	1	1	0	0	0	0
0	0	0	52	1	0	0	0	0	0
1	0	3	0	113	0	0	0	0	0
2	0	0	0	0	77	0	0	0	0
3	0	0	0	0	0	45	2	0	0
4	0	0	0	0	0	0	236	2	0
5	0	0	0	0	0	0	2	150	0
6	0	0	0	0	0	0	0	0	81

tanh

	-2	-1	0	1	2	3	4	5	6
-2	78	1	0	0	0	0	0	0	0
-1	1	173	1	2	0	0	0	0	0
0	0	0	67	0	0	0	0	0	0
1	0	0	0	103	3	0	0	0	0
2	0	0	0	1	66	2	0	0	0
3	0	0	0	0	2	32	1	0	0
4	0	0	0	0	0	1	229	1	0
5	0	0	0	0	0	0	7	169	0
6	0	0	0	0	0	0	0	0	74

softsign

When we run our model with the multi data class, we obtained a 98.7% accuracy rate after nearly 300 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-400-150-9(Output)
- Dropout Rate : 20%

- Kernel Initializer : Uniform
- Activation Method : Tanh
- Epochs : 300
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 17:54 for 4055 different processes. The test duration is approximately 894 milliseconds for 1014 different processes, and the detection time of the new incoming attack is 0.88 milliseconds.

The Layer Mode 3 design results of the multi data class are given in detail in Table 41.

Table 41
Multi Data Class Results for Layer Mode 3

	Epochs	Accuracy	Precision	Recall	F Score
tanh	500	0.981	0.981	0.981	0.981
softsign	500	0.990	0.990	0.990	0.990

An analysis of the data in the Layer Mode 3 design shows that the two most successful results are obtained with the 'tanh' and 'softsign' activation methods.

The confusion matrix of these two methods is given in Table 42.

Table 42
Confusion Matrix for Layer Mode 3 with Test Dataset

	-2	-1	0	1	2	3	4	5	6
-2	60	1	0	0	0	0	0	0	0
-1	2	185	0	0	0	0	0	0	0
0	0	0	69	0	0	0	0	0	0
1	0	0	1	111	1	0	0	0	0
2	0	0	0	0	86	1	0	0	0
3	0	0	0	1	0	41	3	0	0
4	0	0	0	1	1	1	218	4	0
5	0	0	0	0	0	1	0	148	2
6	0	0	0	0	0	0	0	0	77

tanh

	-2	-1	0	1	2	3	4	5	6
-2	60	0	0	0	0	0	0	0	0
-1	0	203	2	2	0	0	0	0	0
0	0	0	73	0	0	0	0	0	0
1	0	0	0	114	1	0	0	0	0
2	0	0	0	1	69	0	0	0	0
3	0	0	0	0	0	33	0	0	0
4	0	0	0	0	0	0	239	1	0
5	0	0	0	0	0	0	1	145	1
6	0	0	0	0	0	0	0	0	68

softsign

When we run our model with the multi data class, we obtained a 99% accuracy rate after nearly 500 epochs. We obtained this accuracy rate with the DNN model created with the following parameter sequence:

- AE Optimizer Algorithm : Adam
- DNN Optimizer Algorithm : SGD
- Nodes in the layers : 129(Input)-400-650-900-650-400-150-9(Output)
- Dropout Rate : 20%
- Kernel Initializer : Uniform
- Activation Method : Softsign
- Epochs : 500
- Output Activation Method : softmax
- Output Loss Method : categorical_crossentropy

With these parameters, the training duration of our model is 44:00 for 4055 different processes. The test duration is approximately 1.8 seconds for 1014 different processes, and the detection time of the new incoming attack is 1.8 milliseconds.

The accuracy and loss graphs of the model are shown in Figure 5.

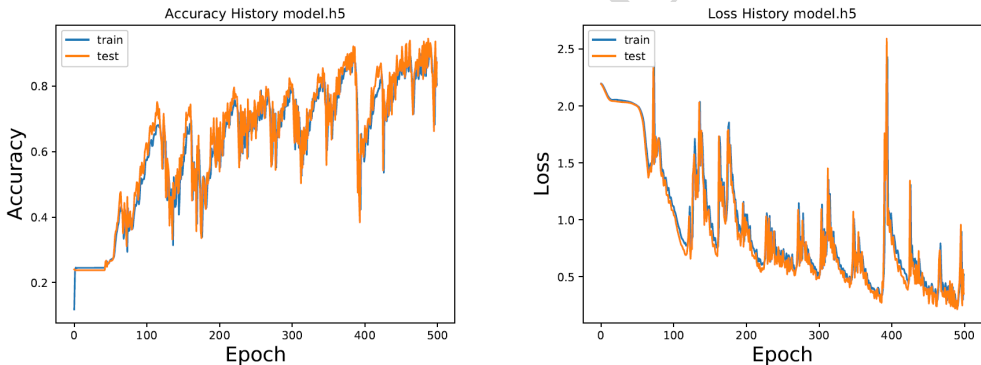


Figure 5. Multi classifier model accuracy and loss history.

3.6. Classification Model Results

We used three different algorithms in the classification models: Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Decision Trees (DTs). We used four different kernel types in the SVM algorithm; *rbf*, *linear*, *poly*, *sigmoid*.

3.6.1. Binary Data Class Results with Classification Models

The models are run in the binary data class with binary tags with 0-Normal and 1-Attack. The results of the three algorithms used in the classification models are given in detail in Table 43.

Table 43
Binary Data Class Results with Classification Models

	SVM	SVM	SVM	SVM	KNN	DTs
Kernel Types	rbf	linear	poly	sigmoid	-	-
Accuracy	0.9941	0.9901	0.9951	0.9941	0.9596	1.0
Precision	0.9941	0.9903	0.9951	0.9941	0.9598	1.0
Recall	0.9941	0.9901	0.9951	0.9941	0.9596	1.0
F Score	0.9941	0.9901	0.9951	0.9941	0.9591	1.0
Process Time (minute)	2:00	2:00	2:00	2:00	4:00	00:08

3.6.2. Multi Data Class Results with Classification Models

The models are run in the multi data class with multitags with -2 Natural (Fault From Line), -1 Natural (Line maintenance), 0-Normal, 1-Attack (Data Injection), 2-Attack (Command Injection), 3-Attack (Command Injection), 4-Attack (Disabling relay function), 5-Attack (Disabling relay function), and 6-Attack (Disabling relay function). The results of the three algorithms used in the classification models are given in detail in Table 44:

Table 44
Multi Data Class Results with Classification Models

	SVM	SVM	SVM	SVM	KNN	DTs
Kernel Types	rbf	linear	poly	sigmoid	-	-
Accuracy	0.6815	0.6588	0.6755	0.6805	0.5996	1.0
Precision	0.7074	0.6813	0.6974	0.6981	0.5972	1.0
Recall	0.6815	0.6588	0.6755	0.6805	0.5996	1.0
F Score	0.6445	0.6338	0.6489	0.6519	0.5947	1.0
Process Time (minute)	14:00	14:00	14:00	14:00	4:00	00:30

4. Conclusion

Industrial systems that perform crucial work to raise people's living standards are partially isolated environments; however, like any electronic system, they are vulnerable to cyberattacks. Research shows that attacks on these systems are increasing day by day. With the increase of cyberattacks, the methods of attack have also begun to differentiate. So, it has become increasingly difficult to detect these cyberattacks in a short time. The detection speed of the cyberattacks on industrial systems (including critical infrastructures) must be very high. In this study, a model was developed to quickly detect cyberattacks on industrial systems. The proposed model is based on deep learning methods. The reason for choosing deep learning methods in this study is the high maturity levels of the algorithms and technologies used. Hence, these technologies have a high level of robustness; they are also used in many commercial products. In this study, we used a new published dataset created by Mississippi State

University and Oak Ridge National Laboratory. Our proposed model's classification performance is better than the Morris et al. results given in the Related Work section. The original work results are 90.4%, 93%, and 99.1%, respectively. In our study; we obtained 100%, 100%, and 99.8% accuracy rates with binary, triple, and multi-labeled datasets, respectively. Our plan is to convert the proposed attack detection model with the transfer learning method. Applying the transfer learning model with an autoencoder algorithm, the new developed model will require less training time.

References

- [1] Baldi P.: Autoencoders, Unsupervised Learning, and Deep Architectures. In: I. Guyon, G. Dror, V. Lemaire, G. Taylor, D. Silver, eds., *Proceedings of ICML Workshop on Unsupervised and Transfer Learning, Proceedings of Machine Learning Research*, vol. 27, pp. 37–49. PMLR, Bellevue, Washington, USA, 2012. <http://proceedings.mlr.press/v27/baldi12a.html>.
- [2] Borges Hink R.C., Beaver J.M., Buckner M.A., Morris T., Adhikari U., Pan S.: Machine learning for power system disturbance and cyber-attack discrimination. In: *2014 7th International Symposium on Resilient Control Systems (ISRCS)*, pp. 1–8. 2014. <http://dx.doi.org/10.1109/ISRCS.2014.6900095>.
- [3] Deng L.: A tutorial survey of architectures, algorithms, and applications for deep learning. In: *APSIPA Transactions on Signal and Information Processing*, vol. 3, p. e2, 2014. <http://dx.doi.org/10.1017/atsip.2013.9>.
- [4] Dondossola G., Szanto J., Masera M., Nai Fovino I.: Effects of intentional threats to power substation control systems. In: *International Journal of Critical Infrastructures*, vol. 4(1-2), pp. 129–143, 2008. <http://dx.doi.org/10.1504/IJCIS.2008.016096>.
- [5] Falliere N., Murchu L.O., Chien E.: W32. stuxnet dossier. In: *White paper, Symantec Corp., Security Response*, vol. 5(6), p. 29, 2011.
- [6] Friedman N., Geiger D., Goldszmidt M.: Bayesian Network Classifiers. In: *Machine Learning*, vol. 29(2), pp. 131–163, 1997. ISSN 1573-0565. <http://dx.doi.org/10.1023/A:1007465528199>.
- [7] Han J., Pei J., Yin Y., Mao R.: Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. In: *Data Mining and Knowledge Discovery*, vol. 8(1), pp. 53–87, 2004. ISSN 1573-756X. <http://dx.doi.org/10.1023/B:DAMI.0000005258.31418.83>.
- [8] Kovacevic A., Nikolic D.: Cyber attacks on critical infrastructure: Review and challenges. In: *Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance*, pp. 1–18. IGI Global, 2015.
- [9] Makhzani A., Shlens J., Jaitly N., Goodfellow I.J.: Adversarial Autoencoders. In: *CoRR*, vol. abs/1511.05644, 2015. <http://arxiv.org/abs/1511.05644>.

- [10] Morris T.H., Gao W.: Industrial control system cyber attacks. In: *Proceedings of the 1st International Symposium on ICS & SCADA Cyber Security Research*, pp. 22–29. 2013.
- [11] Pan S., Morris T., Adhikari U.: Classification of Disturbances and Cyber-Attacks in Power Systems Using Heterogeneous Time-Synchronized Data. In: *IEEE Transactions on Industrial Informatics*, vol. 11(3), pp. 650–662, 2015. ISSN 1551-3203. <http://dx.doi.org/10.1109/TII.2015.2420951>.
- [12] Pan S., Morris T., Adhikari U.: Developing a Hybrid Intrusion Detection System Using Data Mining for Power Systems. In: *IEEE Transactions on Smart Grid*, vol. 6(6), pp. 3104–3113, 2015. ISSN 1949-3053. <http://dx.doi.org/10.1109/TSG.2015.2409775>.
- [13] Vincent P., Larochelle H., Lajoie I., Bengio Y., Manzagol P.A.: Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. In: *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010. ISSN 1532-4435. <http://dl.acm.org/citation.cfm?id=1756006.1953039>.

Affiliations

Murat Yilmaz

Istanbul Sehir University, Cyber Security Engineering, 34865, Istanbul, Turkey,
muratyilmaz85@std.sehir.edu.tr, ORCID ID: <https://orcid.org/0000-0002-5895-7839>

Ferhat Ozgur Catak

TUBITAK BILGEM Cyber Security Institute, Kocaeli, Turkey, ozgur.catak@tubitak.gov.tr

Ensar Gul

Istanbul Sehir University, Cyber Security Engineering, 34865, Istanbul, Turkey,
ensargul@sehir.edu.tr, ORCID ID: <https://orcid.org/0000-0001-8753-6075>

Received: 01.03.2019

Revised: 29.04.2019

Accepted: 29.04.2019