Kyushu Univ, Kyushu, Japan,
Jan. 30(Mon.), 2023.

AI for Intrusion Detection and Privacy-preserving

Prof. Kwangjo Kim
IACR Fellow

Emeritus Professor@KAIST / President@IRCS

KAIST

IRCS 국제사이버보안연구원

CV: Kwangjo Kim

- Contact Information
  - 1202 Sunji-heigtstel, 210-2, Seohyun-ro, Bundang, Korea
  - E-mail: kkj@kaist.ac.kr Home page: http://caislab.kaist.ac.kr/kkj
  - Google Scholar h-index : 46, 9,478 Citations

- Career
  - '79 ~ '97 : Section Head of Coding Tech. #1 in ETRI
  - '99 ~ '00 : Visiting Professor at Univ. of Tokyo, Japan
  - '99 ~ '05 : Director of IACR / Institute for IT-gifted Youth
  - '98 ~ '09 : Professor / Dean of School of Engineering in ICU
  - '02 : 1000 World Leaders of Scientific Influence by ABI
  - '05 ~ '06 : Visiting Scholar at MIT/UCSD
  - '09.1~12 : President of KIISC and Honorable President of KIISC (Current)
  - '09.3 ~ '21.8 : Professor in SOC@KAIST,
  - '12.1~8 : Visiting Professor at KUSTAR, UAE
  - '13.1(7)~2(8) : Visiting Professor at ITB, Indonesia
  - '14 : Who's who in the world (ABI) & 2000 Outstanding Intellectuals of the 21st Century (IBC)
  - '15 : Korean Representative to IFIP TC11
  - '17 : IACR Fellow
  - **'21.8 : Retired from KAIST**
  - **'21.9~ : President @ IRCS / Emeritus Professor @KAIST**

- Academic Activities
  - More than 100 Program Committee Members of Crypto and Security Conferences
  - General Chair of Asiacrypt2004, and CHES2014, Asiacrypt2020 and PQCrypto2021
  - More than 20 invited talks to international conferences
  - Editor-in-Chief, Cryptography, MDPI online Journal

- Course offered / Fluent Language
  - CS448, CS548 / English, Japanese, Korea

- **Awards**
  - Presidential Appreciation('02.),Presidential Citation ('09.9), Minister of NIS ('09.12)

- **Hobby**
  - Driving, Mountain Climbing, Cycling, Skiing, Rafting, etc.









KAIST

IRCS 국제사이버보안연구원

Abstract

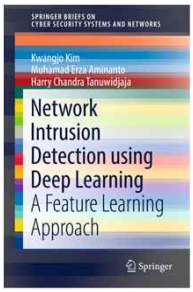
Based on his co-authored two books published by Springer in 2018 and 2021, his presentation consists of two parts:

The 1st half presents recent advances in Intrusion Detection Systems (IDS) using the state-of-the art deep learning methods, which have achieved great breakthrough recently, particularly in the field of computer vision, natural language processing and image processing from the point of detection performances. We discuss a systematic and methodical overview of the latest developments in deep learning and makes a comprehensive comparison between shallow machine learners and deep learning methods. A general overview of deep learning applications to IDS followed by a novel deep feature extraction and selection(D-FES) is suggested. Further challenges and research directions will be suggested.

The 2nd half provides fundamental insights into privacy-preserving and deep learning, offering a comprehensive overview of the state-of-the-art in PPDL methods. It discusses practical issues, and leveraging federated or split-learning-based PPDL. Covering the fundamental theory of PPDL, the pros and cons of current PPDL methods, and addressing the gap between theory and practice in the most recent approaches, it is a valuable reference resource for a general audience, undergraduate and graduate students, as well as practitioners interested learning about PPDL from the scratch, and researchers wanting to explore PPDL for their applications.

Xi'dian, China Oct. 1-2, 2018

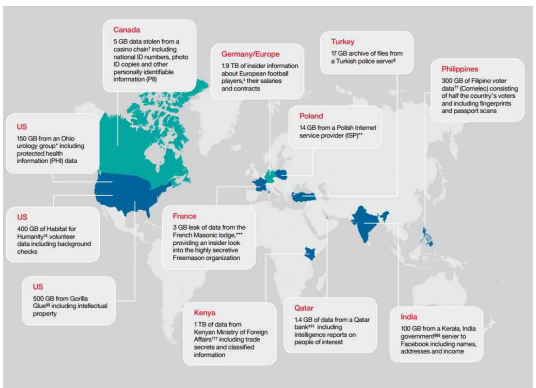
Part I: Intrusion Detection using Deep Learning



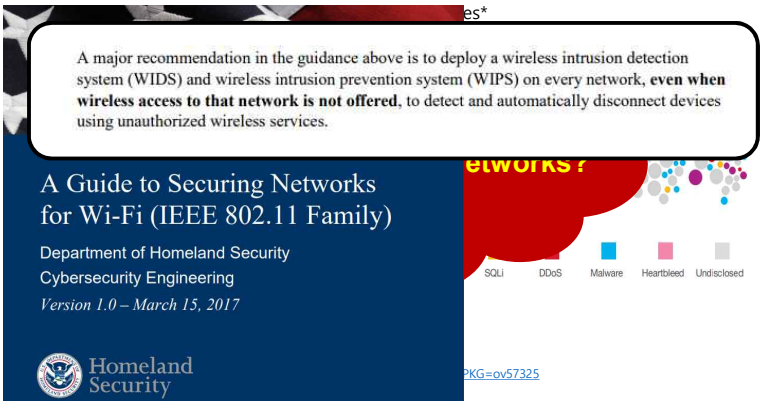
Kwangjo Kim, Muhamad Erza Aminanto, Harry Chandra Tanuwidjaja,"Network Intrusion Detection using Deep Learning",ISBN 978-981-13-1443-8,2018,Springer

Contents

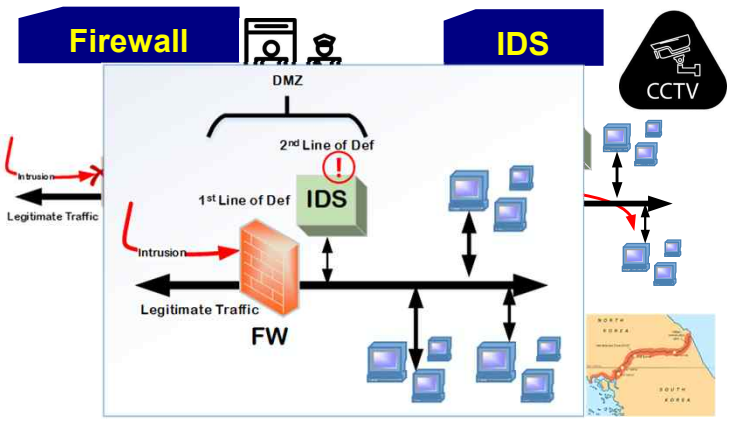
Global Attacks in 2016 (1/2)



Global Attack in 2016 (2/2)



Firewall vs IDS



Types of IDS (methodology) (2/2)

- *Misuse-based*: detects any attack by checking whether the attack characteristics match previously stored signatures or patterns. This also known as signature-based IDS.
- *Anomaly-based*: identifies malicious activities by profiling normal behavior and then measuring any deviation from it. It leverages statistical analysis or machine-learning.
- *Specification-based*: manually defines a set of rules and constraints to express the normal operations. Any violation of the rules and constraints during execution is flagged as an attack.

Comparion of IDS

	Misuse-based	Anomaly-based	Specification-based
Method	Identify known attack patterns	Identify unusual activity patterns	Identify violation of pre-defined rules
Detection Rate	High	Low	High
False Alarm Rate	Low	High	Low
Unknown Attack Detection	Incapable	Capable	Incapable
Drawbacks	Updating signatures is burdensome	Computing any statistical or machine-learning is heavy	Relying on expert knowledge to define rules is undesirable

Learning : Supervised vs Unsupervised

- Unknown attack detection: Detects new attacks without prior knowledge

	Supervised	Unsupervised
Definition	The data are labeled with pre-defined classes.	The data are labeled **without** pre-defined classes
Method	Classification	Clustering
Example	• Support Vector Machine (SVM) • Decision Tree (DT) • Fuzzy Inference System (FIS)	• *k*-means Clustering, • Density-based Spatial Clustering of Applications with Noise (DBSCAN) • Ant Clustering Algorithm (ACA)
Known Attack DR	High	Low
Unknown Attack DR	Low	High

Tree of Deep Learning

- ANN, SAE, RBM, DBN, CNN, *etc*

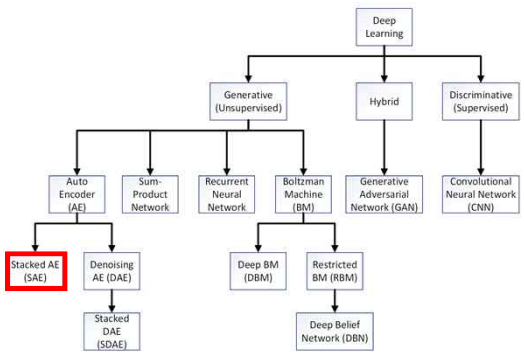


Figure from Aminanto, M.E. and Kim, K.J., "*Deep Learning in Intrusion Detection System: An Overview*", International Research Conference on Engineering and Technology-IRCET 2016, Jun. 28-30, 2016, Bali, Indonesia.

Deep Learning-Based IDSs (1/6)

• DNN (Deep Neural Network)

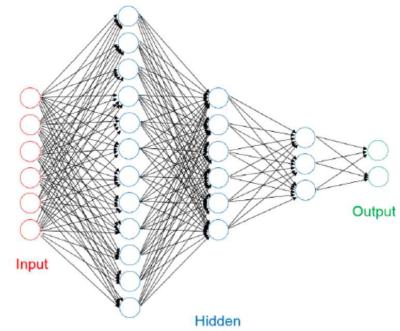


Figure from T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in Wireless Networks and Mobile Communications (WINCOM), 2016 International Conference on. IEEE, 2016, pp. 258–263.

1. T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in Wireless Networks and Mobile Communications (WINCOM), 2016 International Conference on. IEEE, 2016, pp. 258–263.
2. S.S. Roy, A. Mallik, R. Gulati, M.S. Obaidat, and P. Krish-na, "A deep learning based artificial neural network approach for intrusion detection," in International Conference on Mathematics and Computing. Springer, 2017, pp. 44–53.
3. S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," in Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on. IEEE, 2016, pp. 1–8.

Deep Learning-Based IDSs (2/6)

• LSTM-RNN (Recurrent NN)

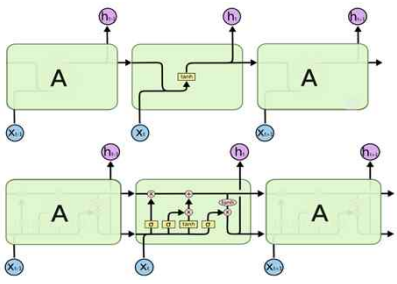


Figure from C. Olah, "Understanding LSTM networks," http://colah.github.io/posts/2015-08-Understanding-LSTMs/, 2015, [Online; accessed 20-February-2018].

1. J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in Platform Technology and Service (PlatCon), 2016 International Conference on. IEEE, 2016, pp. 1–5.
2. Y. LIU, S. LIU, and Y. WANG, "Route intrusion detection based on long short term memory recurrent neural network," DEStech Transactions on Computer Science and Engineering, no.cii, 2017.
3. C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," IEEE Access, vol. 5, pp. 21 954–21 961, 2017.
4. R. C. Staudemeyer, "Applying long short-term memory recurrent neural networks to intrusion detection," South African Computer Journal, vol. 56, no. 1, pp. 136–154, 2015.
5. L. Bontemps, J. McDermott, N.-A. Le-Khac et al., "Collective anomaly detection based on long short-term memory recurrent neural networks," in International Conference on Future Data and Security Engineering. Springer, 2016, pp. 141–152.
6. M. K. Putchala, "Deep learning approach for intrusion detection system (ids) in the internet of things (iot) network using gated recurrent neural networks (gru)," Ph.D. dissertation, Wright State University, 2017.
7. P. K. Bediako, "Long short-term memory recurrent neural network for detecting ddos flooding attacks within tensorflow implementation framework." 2017.

Deep Learning-Based IDSs (3/6)

• CNN (Convolutional Neural Network)

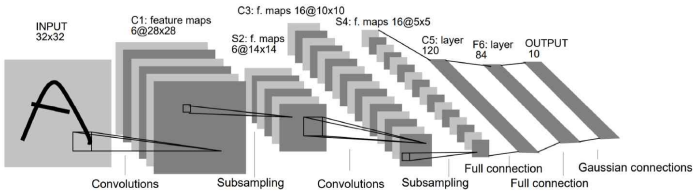
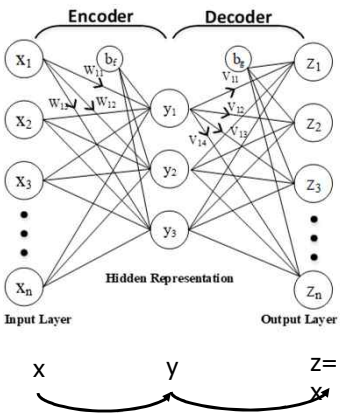


Figure from Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

1. Z. Li, Z. Qin, K. Huang, X. Yang, and S. Ye, "Intrusion detection using convolutional neural networks for representation learning," in International Conference on Neural Information Processing. Springer, 2017, pp. 858–866.

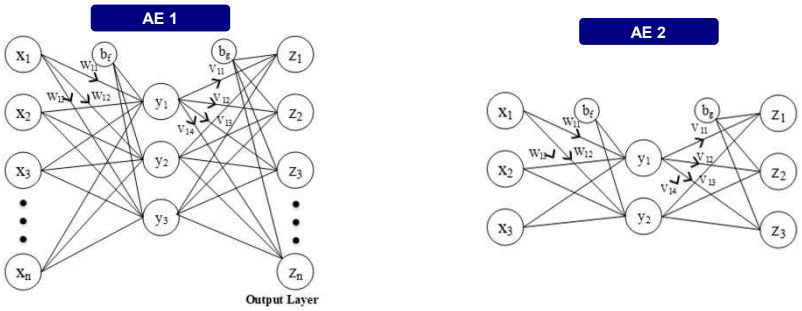
Deep Learning-Based IDSs (4/6)

• AE (Auto-Encoder)



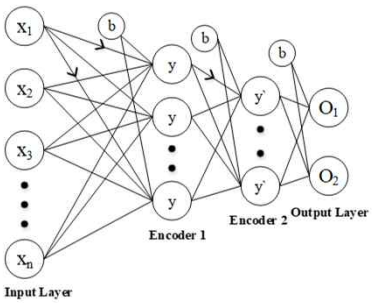
Deep Learning-Based IDSs (5/6)

- SAE (Stacked Auto-Encoder)



Deep Learning-Based IDSs (6/6)

- SAE (Stacked Auto-Encoder)



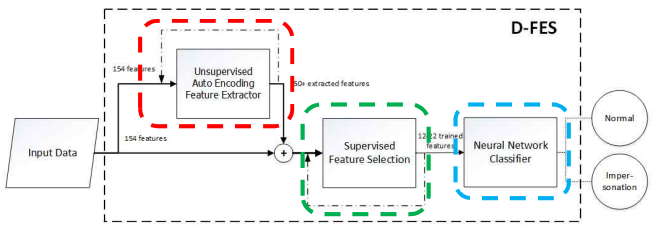
1. A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016, pp. 21–26.
2. Y. Yu, J. Long, and Z. Cai, "Session-based network intrusion detection using a deep learning architecture," in Modeling Decisions for Artificial Intelligence. Springer, 2017, pp. 144–155.
3. M. E. Aminanto, R. Choi, H. C. Tanuwidjaja, P. D. Yoo, and K. Kim, "Deep abstraction and weighted feature selection for Wi-Fi impersonation detection," IEEE Transactions on Information Forensics and Security, vol. 13, no. 3, pp. 621–636, 2018.
4. M. E. Aminanto and K. Kim, "Detecting impersonation attack in Wi-Fi networks using deep learning approach," Information Security Applications: 17th International Workshop, WISA 2016, 2016.
5. M. E. Aminanto and K. Kim, "Improving detection of Wi-Fi impersonation by fully unsupervised deep learning," Information Security Applications: 18th International Workshop, WISA 2017, 2017.

Our SAE Applications

- SAE as a classifier [1]
- Combination of feature extraction and selection [2]
- SAE as a clustering method [3]

1. M. E. Aminanto and K. Kim, "Detecting impersonation attack in Wi-Fi networks using deep learning approach," Information Security Applications: 17th International Workshop, WISA 2016, 2016.
2. M. E. Aminanto, R. Choi, H. C. Tanuwidjaja, P. D. Yoo, and K. Kim, "Deep abstraction and weighted feature selection for Wi-Fi impersonation detection," IEEE Transactions on Information Forensics and Security, vol. 13, no. 3, pp. 621–636, 2018.
3. M. E. Aminanto and K. Kim, "Improving detection of Wi-Fi impersonation by fully unsupervised deep learning," Information Security Applications: 18th International Workshop, WISA 2017, 2017.

Deep-Feature Extraction and Selection(D-FES)

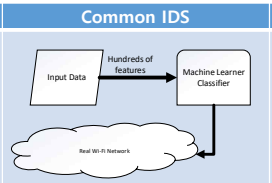
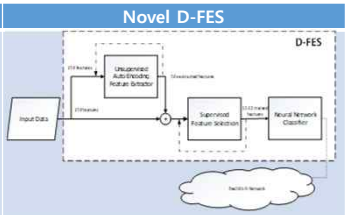


Extraction – Selection – Classification

- A novel Deep-Feature Extraction and Selection (D-FES) for designing feature-focused learning IDS by combining stacked feature extraction and weighted-feature selection techniques
- Consists of 3 main tasks:
  - ✓ Task 1: Unsupervised Auto Encoding Feature Extractor using Stacked Auto Encoder (SAE)
  - ✓ Task 2: Supervised Feature Selection using Artificial Neural Network (ANN) with combined inputs (original and extracted features)
  - ✓ Task 3: Neural Network Classifier using ANN

M. E. Aminanto, R. Choi, H. C. Tanuwidjaja, P. D. Yoo, and K. Kim, "Deep abstraction and weighted feature selection for Wi-Fi impersonation detection," IEEE Transactions on Information Forensics and Security, vol. 13, no. 3, pp. 621–636, 2018.

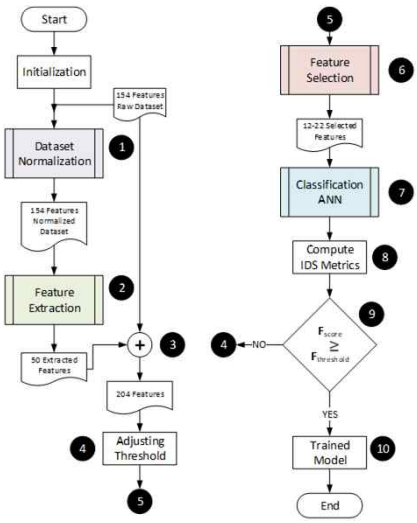
Common IDS vs Novel D-FES

	Common IDS	Novel D-FES
Scheme		
Pros	• Faster for training task	• More meaningful features are observed from extracted features • Lightweight input for the classifier • Faster for classification task • Higher accuracy • Higher detection
Cons	• Slower for classification task • Lower accuracy • Lower detection	• Slower for training task

KAIST

IRCS 국제사이버보안연구원

Flowchart of D-FES Scheme



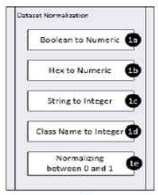


Figure 7.2: Flowchart of D-FES: sub-process 1

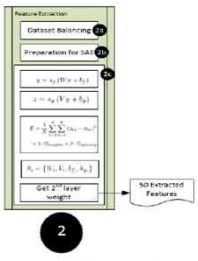


Figure 7.3: Flowchart of D-FES: sub-process 2

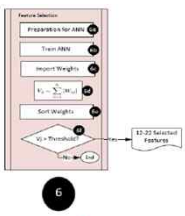


Figure 7.4: Flowchart of D-FES: sub-process 6

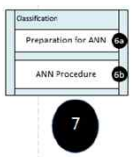


Figure 7.5: Flowchart of D-FES: sub-process 7

KAIST

IRCS 국제사이버보안연구원

D-FES Source Code (1/3)

D-FES Source Code (2/3)

D-FES Source-Code (3/3)

```
%-------------------------------------------------%
%------------ Beginning of Flowchart #3 ----------%
%-------------------------------------------------%
d_IA_train_WoT = cat(1,d_IA_train_WoT, feat2_saya); %concatenation -> 204 Features
%-------------------------------------------------%
%---------------- End of Flowchart #3 ------------%
%-------------------------------------------------%

Xtrain=d_IA_train_WoT;%assign Xtrain
Xtrain=Xtrain';%transpose input
Ytrain=d_IA_train_T_dummy;%assign Ytrain
Ytrain=Ytrain';%transpose input target
%%%%%%%%Data for test%%%%%%%%
d_IA_test_WoT=d_IA_test(:,1:154);%get fld target class att
d_IA_test_T=d_IA_test(:,155);%get target class att only
d_IA_test_T_dummy=dummyvar(d_IA_test_T);%get the hot 1 vector of target class
Xtest=d_IA_test_WoT;%assign Xtest
Xtest=Xtest';%transpose input test
Ytest=d_IA_test_T;%assign target class for testing
Ytest=Ytest';%transpose target class for testing
Ytest=d_IA_test_T_dummy;%assign Ytest with dummy
Ytest=Ytest';%transpose target with dummy
%---------------------- End of Flowchart 4a Preparation for ANN ----------------%

%---------------------- Beginning of Flowchart 4b ANN Procedure ----------------%
% >> nnstart
% >> Select Pattern Recognition App
% >> Select Xtrain for input and Ytrain for target data
% >> Set Hidden Neuron as 100
% >> TRAIN
% >> Save the output model
%---------------------- End of Flowchart 4b ANN Procedure ----------------------%

%------------------------ Beginning of Flowchart 4c Import Weight ------------%
berat=net.IW{1,1};%Import output weight from the hidden layer
indeks=net.inputs{1}.range;%range yg tidak penting bernilai 0, kolom ke 1 masing
for i=1:154 %check range (min and max) on each input attribute
    range_indeks=indeks(1,2)-indeks(1,1);
    if range_indeks==0
        tmp(1,1)=0;%If range=0, it means this att is not important
    else tmp(1,1)=1;
    end %end if
end %end for
tmp=logical(tmp);
```

87

```
indeks(:,1)=[1:154]; %fill first column with increment 1-41
tmp=indeks(:,2)'=0;%Get the index that != 0
indeks2=indeks(tmp,:);%Select row != 0
indeks_aja=indeks2(:,1);%Get the index only
indeks_aja=indeks_aja';
berat=abs(berat);%Absolute value of weight
%------------------------- End of Flowchart 4c Import Weight ----------------%

%---------------------- Beginning of Flowchart 4d Summation of Weight ----------%
sizeya = size(berat);%size of weight
col=sizeya(1,2);%column index
row=sizeya(1,1);%row index
berat_per_input=zeros(1,col);
berat_per_input(1,:)=indeks_aja;
for i=1:col %iteration column
    berat_per_input(2,i)=0;
    for j=1:row %iteration rows
        berat_per_input(2,i)=berat_per_input(2,i)+berat(j,i);
    end %end for
end %end for
%---------------------- End of Flowchart 4d Summation of Weight ----------------%

%---------------------- Beginning of Flowchart 4e Sorting Weight ---------------%
berat_per_input=berat_per_input'; %transpose
berat_sort=sortrows(berat_per_input, 2); %sort descending
%------------------------ End of Flowchart 4e Sorting Weight -------------------%

%-------------------------------------------------%
%------------ Beginning of Flowchart #4 ----------%
%-------------------------------------------------%
%------------------ Beginning of Flowchart 4f Compare Weight ---------------%
tmp2=berat_sort(:,2)>=16;%adjust the treshold value
%-------------------------------------------------%
%-------------------- Beginning of Flowchart 4f Compare Weight ---------------%
%---------------- End of Flowchart #4 ------------%
%-------------------------------------------------%

selected_features=berat_sort(tmp2,1); %%% 12-22 Selected Features %%%%

%-------------------------------------------------%
%---------------- End of Flowchart #5 ------------%
%---------------- Feature Selection process ------%
%-------------------------------------------------%
```

88

```
%-------------------------------------------------%
%------------ Beginning of Flowchart #7 ----------%
%---------------- Classification process ---------%
%-------------------------------------------------%

%---------------------- Beginning of Flowchart 7a Preparation for Classification --------%
d_IA_train_trans=d_IA_train';
d_IA_test_trans=d_IA_test';
nkurang_mat=size(selected_features);%get # features
jumlah_fitur=nkurang_mat(1,1);%get # row (# features)
indeks_fitur=[1:154];
for j=1:154%increment at feature index
    for i=1:jumlah_fitur%increment at selected features
        if indeks_fitur(1,j)==selected_features(1,i)%if equal, assign 1
            indeks_fitur(2,j)=1;
            break;%get out of if
        else indeks_fitur(2,j)=0;
        end %end if
    end %end for
end %end for
%recover selected features within all features
indeks_fitur_aja=indeks_fitur(2,:);%get the boolean only
indeks_fitur_aja=indeks_fitur_aja';
indeks_fitur_aja_kelas=indeks_fitur_aja;%add 155th index as class
indeks_fitur_aja_kelas(156,1)=1;%add 155th
indeks_fitur_aja_logical=logical(indeks_fitur_aja_kelas);%convert into logical
%training dataset with selected features
train_data=d_IA_train_trans;
train_data=train_data(indeks_fitur_aja_logical,:);
input_fitur=train_data(1:jumlah_fitur,:);%training without target class
target_train=train_data([jumlah_fitur+1],:);%training target class
target_train_final=dummyvar(target_train);
target_train_final=target_train_final';
%test dataset with selected features
test_data=d_IA_test_trans;
test_data=test_data(indeks_fitur_aja_logical,:);
input_fitur_test=test_data(1:jumlah_fitur,:);%test without target class
target_test=test_data([jumlah_fitur+1),:);%test target class
target_test_final=dummyvar(target_test);
target_test_final=target_test_final';
%final assignment
x_train = input_fitur;   % the input for training
y_train = target_train_final;  % the target for training
x_test = input_fitur_test;  % the input for testing
```

89

```
y_test = target_test_final;   % the target for testing
%---------------------- End of Flowchart 7a Preparation for Classification --------%

%---------------------- Beginning of Flowchart 7b ANN Procedure ----------%
% >> nnstart
% >> Select Pattern Recognition App
% >> Select Xtrain for input and Ytrain for target data
% >> Set Hidden Neuron as 50
% >> TRAIN
% >> Get the confusion matrix ==> Flowchart #8 Compute ISS Metrics
% >> Save the output model
%---------------------- End of Flowchart 7b ANN Procedure ----------------%

%-------------------------------------------------%
%---------------- End of Flowchart #7 ------------%
%---------------- Classification process ---------%
%-------------------------------------------------%

%---------------------- Beginning of Flowchart #8 Comparison -------------%
% Compare F score and F treshold
%---------------------- End of Flowchart #8 Comparison ----------------%

%---------------------- Beginning of Flowchart #10 Model -------------%
% Trained model complete
%---------------------- End of Flowchart #10 Model -------------------%

%------------ END of File --------------------%
```


Demo Video

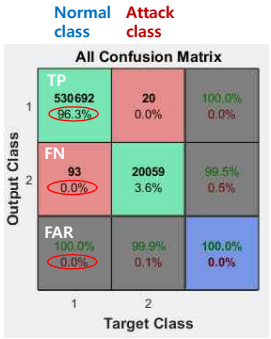
PC Windows 10 with Intel Xeon E-3-1230v3 CPU @3.30 GHz, 32 GB RAM
MATLAB 2016b (student license) with Neural Network Toolbox

Results

1: Normal Class
2: Impersonation Attack Class

Normal Class		
Class	Dataset	
	Training	Test
Normal (unbalanced)	1,633,190	530,785
Normal (balanced)	163,319	53,078

Attack Class		
Class	Dataset	
	Training	Test
Impersonation	48,522	20,079
Flooding	48,484	8,097
Injection	65,379	16,662
Total	162,385	44,838



- True-Positive (TP): normal instances classified correctly as normal class
- False-Negative (FN): normal instances classified incorrectly as attack class
- FAR: the ratio between the normal instances classified incorrectly as attack class and the entire normal instances

KAIST

국제사이버보안연구원

Comparison

1. SAE as a classifier (WISA16)
2. Combination of feature extraction and selection (IEEE IF&S18)
3. SAE as clustering method (WISA17)

Method	DR (%)	FAR (%)
1	65.178	0.143
2	99.918	0.012
3	92.180	4.400
Kolias *et al.**	22.008	0.021

AWID Dataset

	Normal	Impersonation	Flooding	Injection
	Balanced			
Train	163,319	48,522	48,484	65,379
Test	53,078	20,079	8,097	16,682
	Unbalanced			
Train	1,633,190	48,522	48,484	65,379
Test	530,785	20,079	8,097	16,682

*) Kolias, Constantinos, *et al.*, "Intrusion detection in 802.11 networks: empiri cal evaluation of threats and a public dataset," IEEE Communications Survey s & Tutorials, vol:18.1, pp: 184-208, 2015.

KAIST

국제사이버보안연구원

Summary

✓The principle of DL is to process hierarchical features of the provided input data, where the higher-level features are composed by lower-level features.

✓DL can discover sophisticated underlying structure and feature from abstract aspects.

✓The goal of DL is to learn and output feature representation which makes more suitable for feature engineering.

Future Challenges

✓Huge training load in the learning step.
✓How to apply DL in constrained-computation devices.
✓Incorporating DL models as a real-time classifier.
✓IDS detecting zero-day attacks with high detection rate and low false alarm rate.
✓Comprehensive measure not only detection but also prevention, *etc.*

For details, refer to Prof. Muhamad Erza Aminanto's Ph.D Thesis entitled "*Utilizing Deep Abstraction for Higher Intrusion and Detection in Wireless Networks*", 2018, KAIST
https://caislab.kaist.ac.kr/publication/thesis_files/2018/AMPhD.pdf