https://doi.org/10.7840/kics.2022.47.11.1804

무선 센서 네트워크에서 침입 탐지 머신 러닝 기법의 성능 비교 분석

임 성 욱•, 유 명 식°

Comparison on Machince Learning Techniques for Intrusion Detection in Wireless Sensor Network

SungWook Lim•, Myungsik Yoo°

요 약

무선 센서 네트워크를 이루는 센서들은 개방적이고 분산된 특성을 가지고 있으며, 센서 노드의 제한된 리소스 및 실시간 데이터 수집 등의 요인들로 인해 공격에 매우 취약하다. 공격자는 이러한 취약점을 이용해 가짜 메시지 주입 및 네트워크 리소스 낭비 등의 공격을 시도하며, 이러한 공격 중 DoS 공격에 대해 패턴을 분석하여 분류하기 위해 트리 기반 머신 러닝을 사용하였다. 머신러닝의 성능 비교분석을 훈련 방식의 차이점, Feature Importance 그리고 각 공격 유형의 특징을 고려하였다. 실험 결과에 대한 비교 분석을 위하여 머신 러닝 성능 평가 지표 중 F1-Score를 사용하였다. 분석 결과를 통하여 단일 트리 구조를 가지는 Decision Tree가 가장 낮은 성능을 보이며, 가중치를 이용해 모델을 훈련하는 Boosting 기반 모델이 높은 성능을 보임을 증명하였다.

키워드 : WSN, 침입 탐지, 머신 러닝, DoS 공격, 성능 비교

Key Words : Wireless Sensor Network, Itrusion Detection, Machine Learning, Dos Attack, Performance Comparison

ABSTRACT

Sensors constituting a wireless sensor network have open and distributed characteristics, and are very vulnerable to attacks due to factors such as limited resources of sensor nodes and real-time data collection. Attackers use these vulnerabilities to attempt attacks such as injecting fake messages and wasting network resources. Among these attacks, tree-based machine learning is used to analyze and classify patterns for DoS attacks. In the performance comparison analysis of machine learning, differences in training methods, feature importance, and characteristics of each attack type were considered. For comparative analysis of the experimental results, F1-Score was used among the machine learning performance evaluation indicators. Through the analysis results, it was proved that the decision tree with a single tree structure showed the lowest performance, and the boosting-based model, which trains the model using weights, showed high performance.

※ 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 정보통신 · 방송 연구개발사업의 일환으로 수행하였음. [2019-0-00135-004, 5G 기반 스마트센서 검증 플랫폼 구축]

• First Author : Department of Information Communication Convergence, Soongsil University, Seoul, South Korea, ernasty@soongsil.ac.kr, 학생회원

° Corresponding Author : School of Electronic Engineering, Soongsil University, Seoul, South Korea, myoo@ssu.ac.kr, 종신회원

논문번호 : 202206-117-B-RN, Received June 16, 2022; Revised July 29, 2022; Accepted August 25, 2022

Ⅰ. 서 론

무선 센서 네트워크(이하 : WSN)는 물리적 또는 환경적 조건을 조사하기 위해 무선 통신이 가능한 센서로 구성된 무선 네트워크의 한 형태로 그림 1은 그 구성을 보여준다. WSN은 실시간으로 광범위한 영역에서 데이터 수집을 필요로 하는 군사 감시, 건물 보안 및 의료시설 등이 증가함에 따라 중요한 연구 분야가 되었다.

WSN을 이루는 센서들은 개방적이고 분산된 특성 그리고 센서 노드의 제한된 리소스 및 실시간 데이터 수집 등 여러 가지 요인들로 인해 공격에 매우 취약하고 이에 따라 공격자의 표적이 될 확률이 높다.

이러한 WSN의 취약점을 이용하여 공격자가 WSN에 주는 영향으로는 센서 노드의 손상, 메시지 가로채기, 가짜 메시지 주입 및 네트워크의 리소스 낭비 등이 있다. WSN에 가해지는 DoS (Denial-of-Service) 공격은 WSN의 보안을 위협하는 가장 일반적이고 위험한 공격 중 하나로 간주되며, 주요 목적은 WSN이 제공하는 서비스를 중단하거나 일시적으로 정지시키는 것이다.

이로 인하여 다양한 네트워크 보안 위협으로부터 WSN을 지키는 것은 주요한 과제가 되었다. 위에서 언급한 보안 위협을 피하거나 방지하는 프로세스가 언제나 성공하는 것은 불가능하기에 알려지지 않은 공격을 탐지하고 이를 경고하기 위한 침입 탐지 시스템(이하 : IDS)이 필요하다. IDS(Intrusion Detection System)는 그림 2와 같은 구조를 가지며, 의심스럽거나 비정상적인 활동을 감지하고 침입이 발생하면 시스템 관리자에게 침입 사실을 보고하는 역할을 한다. 그림 2에 표시된 정보 분석기는 정보를 분석하고 가공하여 침입 탐지를 실행하는 역할을 수행하며, 해당 영역에서 머신 러닝이 사용된다.

본 논문에서는 WSN에 알려지지 않은 공격이 탐지되었을 경우 해당 공격을 분류 및 경고하기 위하여 공격이 없을 때 상태와 4가지 유형의 DoS 공격을 발생

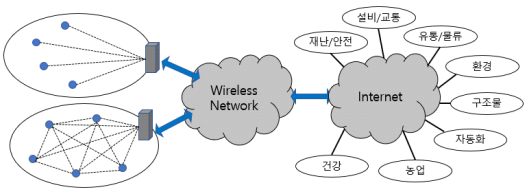

그림 1. WSN 구조
Fig. 1. WSN Architecture

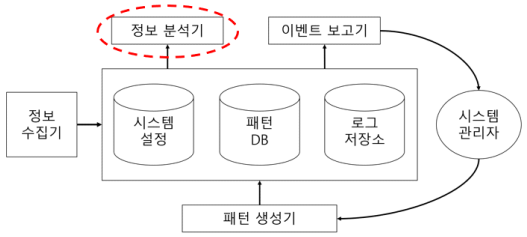

그림 2. IDS 구조
Fig. 2. IDS Architecture

시켜 WSN 시뮬레이션을 통하여 생성된 WSN-DS 데이터 셋을 이용해 WSN 침입 탐지에서 Tree 구조를 기반으로 하는 다섯 가지의 머신 러닝 알고리즘의 성능을 비교하는 것을 목적으로 한다.

본 논문은 다음과 같이 구성된다. 섹션 Ⅱ는 본 논문에서 사용한 WSN-DS 데이터 셋에 대한 설명과 다른 데이터 셋을 이용한 WSN에서 머신 러닝을 적용하여 침입 탐지 성능 비교를 진행한 연구에 대해 소개한다. 섹션 Ⅲ에서는 본 논문에서 고려하는 머신 러닝의 특징에 대해 설명한다. 섹션 Ⅳ에서는 데이터 셋의 구성 요소에 대한 설명 및 각 머신 러닝의 하이퍼 파라미터 설정값을 설명한다. 섹션 Ⅴ에서는 Feature Importance와 과적합 방지에 대하여 설명한 후 Feature Importance 측정 결과를 표를 이용해 정리한다. 또한 머신 러닝을 사용하여 훈련 및 실험 결과를 표와 그래프를 이용하여 표현한 후 각 머신 러닝의 특징과 Attack Type의 특징에 연관 지어 성능을 비교하고 분석한다. 마지막으로 섹션 Ⅵ에서는 결론을 맺는다.

Ⅱ. 관련 연구

WSN의 취약점을 보완하기 위한 IDS의 정보 분석기는 다수의 공격이 발생하였을 경우 공격들의 특징을 분석하여 최대한 빠르고 정확하게 정보를 이벤트 보고기에 전달할 필요성이 높아졌다. 이에 따라 정보 분석기에 사용되는 머신 러닝의 성능을 비교하기 위한 연구가 진행되었다.

Gaurav Meena 연구팀[1]이 연구한 NSL-KDD와 KDDcup99 데이터 셋을 이용한 IDS 분류에서 연구팀은 J48 Decision Tree와 Naive Bayes 알고리즘을 사용하여 IDS에 가해지는 비정상적 행동과 사전에 정의된 비정상적인 패턴을 인식하여 침입 탐지를 수행한다. 이 연구에서 주요 초점은 WEKA 도구와 다양한 분류 알고리즘을 이용해 침입 탐지 정확도를 판단하는 것이다.

Thi-Thu-Huong Le 연구팀[2]이 연구한 WSN에서 DoS 공격을 분류하기 위한 효율적인 방법 연구에서는 WSN-DS 데이터 셋을 이용하여 RNN과 Random Forest의 정확도를 비교하였다.

Ravipati Rama Devi 연구팀[3]은 KDD-99 와 NSL-KDD 데이터 셋을 사용하여 침입 탐지 분류 연구를 진행하였다. 이들은 지도 학습, 비지도 학습 그리고 준지도 학습에 속한 머신 러닝을 이용하여 침입 탐지 시스템을 구현하였을 경우의 성능을 평가하기 위한 실험을 수행하였다. 이들의 연구 결과에서는 각 머신 러닝의 정확성, 탐지율, 합리적인 오경보율 면에서 어떤 접근 방식이 더 효과적이었지는 보여준다.

LAHEEB M. IBRAHIM의 연구팀[4]은 네트워크에서 비정상적인 트래픽 감지에서 처리 능력 및 스토리지 측면에서 컴퓨팅 성능 영역의 발전으로 리소스 집약적인 지능형 알고리즘의 호스팅과 침입 탐지 활동 감지 능력이 향상된 것을 기반으로 하여 침입 탐지 시스템의 일부로 구현될 Self Organization Map(SOM) 인공 신경망의 성능을 KDD-99와 NSL-KDD 데이터 셋을 이용해 다양한 성능 평가 지표를 기반으로 비교 및 분석한다.

Ashwini B. Abhalea의 연구팀[5]은 WSN의 노드가 낮은 배터리 전원 공급, 다른 노드에 대한 종속성 및 자연에 분산되어 있어 다양한 보안 관련 공격에 노출되어 있으며, 공격은 특정 계층만이 아닌 OSI 모델의 모든 계층에서 발생하며 이러한 이유로 인해 발생하는 다양한 문제 해결을 위해 침입 탐지 시스템 구축을 주장한다. 이 연구에서는 침입 탐지 시스템을 구축하기 위한 적합한 머신 러닝의 선택을 위해 KDD-99 데이터 셋의 오류를 수정하여 만들어진 NSL-KDD 데이터 셋을 이용해 SVM, Decision Tree, LightGBM, Gradient Boosting, KNN 등의 머신 러닝의 성능을 비교하였다.

Alexander Zien의 연구팀[6]은 트리 기반 모델에 대한 분석을 통해 우수한 예측 또는 분류 성능을 달성할 수 있는 FIRM(Feature Importance Ranking Measure)에 대하여 연구하였다. FIRM은 기존 Feature Importance 측정 방식과는 달리 Feature 간의 상관관계를 고려한다. 이로 인해 훈련 데이터에서 나타나는 Feature가 노이즈에 의해 차단되어도 가장 관련성이 높은 Feautre를 찾아낼 수 있다.

Ⅲ. 머신 러닝 기법

본 논문에서는 트리 기반 머신 러닝 알고리즘 중 기본적인 단일 트리 구조로 이루어진 Decision Tree 그리고 다중 트리 구조로 이루어진 Random Forest, Extra Tree, XGBoost 그리고 LightGBM을 사용한다. 다중 트리 구조는 크게 두 가지로 분류할 수 있다. 투표를 통해 분류 결과를 정하는 Majority 방식과 가중치를 이용해 과거의 모델을 수정해 나가면서 결과를 결정하는 Boosting 방식으로 나뉜다. Majority 방식은 그림 3과 같이 각각의 트리가 병렬 구조로 되어 있어 모든 트리가 독립적으로 훈련되어 서로에게 영향을 주지 않고 결과를 생성한다. 그에 반해 Boosting 방식은 그림 4와 같이 트리가 직렬로 연결되어 있으며 이전에 생성된 트리가 나중에 생성된 트리에 영향을 주어 분류 결과를 정하게 된다. 본 논문에서 사용하는 Majority 기반 머신 러닝에는 Random Forest와 Extra Tree가 있으며, Boosting 기반 머신 러닝에는 XGBoost와 LightGBM이 있다. 3.1에서는 트리 기반 머신 러닝이 클래스를 분류하는 방법에 대해 설명하고 3.2부터 3.6에서는 트리 기반 머신 러닝의 클래스 결정 방법과 각 머신 러닝의 특징 및 훈련 방식 그리고 주요 하이퍼 파라미터에 대해 설명한다.

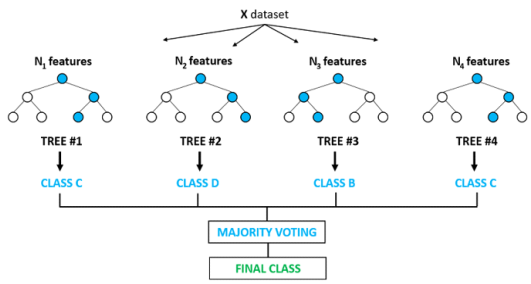

그림 3. Majority 방식
Fig. 3. Majority Method

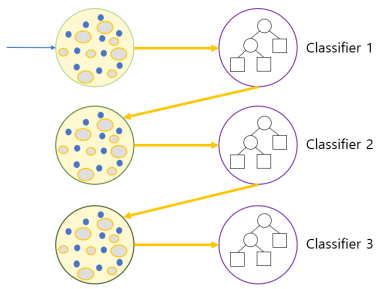

그림 4. Boosting 방식
Fig. 4. Boosting Method

3.1 클래스 분류 방법

지도 학습의 경우 클래스는 Feature의 특정한 값

하나로 정해지는 것이 아닌 범주에 의해 정해진다. 예를 들어 설명을 하면 WSN-DS 데이터 셋에 ADV_S라는 CH가 전송한 광고 메시지의 수를 뜻하는 Feature가 존재한다. 해당 Feature의 경우 값이 16 이상인 경우 해당 노드가 Flooding 공격을 받은 것으로 분류된다. 그리고 나머지의 경우 16미만(0~15)인 경우 Flooding을 포함한 다른 공격 유형이 모두 존재한다. 이를 통해 특정 클래스로 분류하는 것은 특정한 하나의 값이 아닌 특정 값의 범주에 의해 정해지는 것을 알 수 있다. 또한 트리 기반 머신 러닝에서 다중 클래스의 경우 훈련시킬 때 입력한 max_depth까지 트리를 확장한 후 각 노드마다 존재하는 클래스의 Information Gain을 모두 계산하여 해당 값이 가장 큰 것을 해당 노드의 클래스로 확정하게 된다. 위의 예시를 통해 설명을 하면 ADV_S가 16미만인 경우 모든 공격 유형을 가지고 있다고 하였다. 여기서 ADV_S가 2 이상(2~15)인 경우 해당 노드에 속한 공격 유형은 Flooding과 Normal 두 가지가 존재하게 된다. 이때 해당 노드가 트리의 마지막 노드의 경우 각각의 공격 유형에 대해 Information Gain을 계산하여 큰 값을 가지는 공격 유형이 해당 노드의 클래스가 된다.

3.2 Decision Tree

Decision Tree는 단일 트리 구조의 머신 러닝으로 매 분기마다 데이터를 분류하기 위해 하나의 영역을 특정 기준을 사용하여 두 개의 영역으로 분리한다. 이때 선택하는 분리 기준은 불순도를 기반으로 하며 불순도는 해당 영역 내에 서로 다른 데이터가 얼마나 섞여 있는지를 뜻한다. Decision Tree는 이러한 불순도를 최소화하는 방향으로 학습을 진행한다. Decision Tree는 오버 피팅 방지를 위해 가지치기(Pruning)라는 기법을 적용하는데 이는 Decision Tree의 최대 깊이를 제한하거나 터미널 노드의 최대 개수 또는 한 노드가 분할하기 위한 최소 데이터 수를 제한한다. 이러한 제한 사항을 설정하는 것이 하이퍼 파라미터로 Decision Tree에는 max_depth, min_samples_split 그리고 min_sample_leaf 등이 있다.

3.3 Random Forest

Random Forest[8]는 Majority 방식을 사용하는 다중 트리 구조의 머신 러닝으로 단순한 Tree를 다수 생성하고 각각의 Tree를 개별적으로 훈련시키고 결합하여 가장 많은 투표를 받은 Class를 선택하는 약한 학습자 모델로 분산을 최소화하고 과적합을 방지하는 것을 목적으로 한다. 그림 5의 좌측은 각각의 Tree에

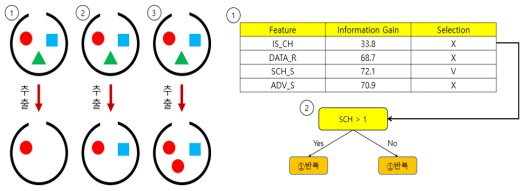

그림 5. RF의 Bootstrap(좌측)과 Feature 선택 방법(우측)
Fig. 5. RF Bootstrap (left) and Feature Selection Method (right)

할당되는 Original 데이터의 부분집합을 생성할 때 데이터 추출 방식에 대해 나타내었다. 해당 방식은 Bootstrap이라고 하며, 데이터 추출 시 중복을 허용하여 각 트리마다 Unique한 데이터 셋을 할당하는 것을 목적으로 한다. 그림 5의 우측은 자식 노드를 생성할 시 그 기준인 Split-Point를 정하기 위한 Feature 선택 방법을 나타내었다. Random Forest는 Feature 선택 시 랜덤으로 일부의 Feature를 선정하여 Feature 집합을 생성하고 해당 집합 내에 있는 Feature들의 Information gain을 각각 계산하여 가장 수치가 높은 Feature를 Split-Point로 지정한다. 이를 위한 주요 하이퍼 파라미터로는 n_estimators, max_depth 및 max_features 등이 있다.

3.4 Extra Tree

Extra Tree[9]는 Majority 방식을 사용하는 다중 트리 구조의 머신 러닝으로 단순한 Tree를 다수 생성하고 각각의 Tree를 개별적으로 훈련시키고 결합하여 가장 많은 투표를 받은 Class를 선택하는 약한 학습자 모델로 전체적인 구조는 Random Forest와 유사하다. Extra Tree와 Random Forest의 훈련 방식의 차이가 두드러지게 나타나는 부분은 각 트리에 할당하는 데이터 셋의 부분집합을 생성하는 과정과 Split-Point 생성 시 Feature를 선택하는 과정이다. Extra Tree의 경우 그림 6의 좌측에 보이는 것처럼 데이터 셋의 부분

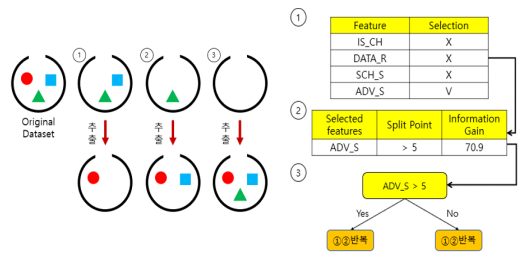

그림 6. ET의 데이터 추출(좌측)과 Feature 선택 방식(우측)
Fig. 6. ET data extraction (left) and feature selection method (right)

집합을 생성할 때 중복을 허용하지 않아 모든 Tree가 유사한 데이터 셋을 할당받는다. Split-Point를 결정하는 과정에서도 그림 6의 우측과 같이 모든 Feature 중 랜덤으로 하나의 Feature를 선택한 뒤 해당 Feature를 사용하였을 때의 최적 분할 기준을 찾아 모델을 훈련한다. Extra Tree의 주요 하이퍼 파라미터로는 max_depth, n_estimators 및 min_samples_leaf 등이 있다.

3.5 XGBoost

XGBoost[10]는 Extreme Gradient Boosting의 약자로 Gradient Boost 알고리즘을 병렬 학습이 지원되도록 구현한 다중 트리 기반 머신 러닝 알고리즘으로 성능이 낮은 예측 모형들의 학습 에러에 가중치를 두고, 순차적으로 다음 학습 모델에 가중치를 반영하여 강한 예측 모형을 생성한다. Gradient Boost 모델과는 달리 과적합 규제 기능과 Early Stopping 기능을 지원하며, 결측치에 대해서 내부적으로 처리해 주는 과정이 존재한다. 그림 7은 XGBoost가 트리 생성 시 노드를 분할하는 방식에 대해 나타낸 그림으로 균형 트리 분할이라고 한다. 균형 트리 분할은 트리의 깊이를 최소화할 수 있으며 Overfitting에 보다 더 강한 구조를 가질 수 있다. 하지만 균형을 맞추기 위한 시간이 필요하다는 단점이 있다. XGBoost의 주요 하이퍼 파라미터로는 n_estimators, learning_rate 및 max_depth 등이 있다.

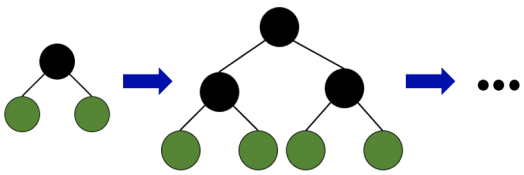

그림 7. 균형 트리 분할 방식
Fig. 7. Balanced tree split method

3.6 LightGBM

LightGBM[11]은 XGBoost와 동일한 Boosting 계열의 알고리즘으로 학습 에러에 가중치를 두고 순차적으로 다음 학습 모델에 가중치를 부여한다. LightGBM은 이름에서도 알 수 있다시피 Gradient Boost 모델을 좀 더 가볍게 하여 상대적으로 더 빠른 학습 시간, 적은 메모리 사용량 그리고 Feature의 자동 변환 및 최적 분할을 지원하는 것을 목적으로 한다. 이를 위해 LightGBM은 다른 Boosting 모델과는 다른 트리 분할 방식을 선택하게 되었으며, 해당 방식

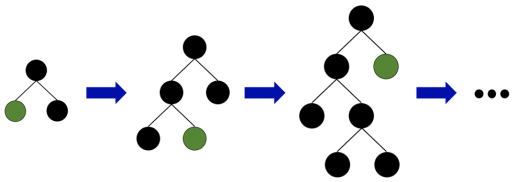

그림 8. 리프 중심 트리 분할 방식
Fig. 8. Leaf-Wise Tree Growth

은 리프 중심 트리 분할 방식이라고 한다. 그림 8는 리프 중심 트리 분할 방식을 도식화한 것이며, 트리가 수직적으로 확장되는 것을 볼 수 있다. 이러한 트리 분할 방식은 동일한 노드를 확장할 때 더 적은 손실로 트리 분할이 가능하다. LightGBM의 경우 적은 데이터를 가진 데이터 셋을 이용하여 훈련할 경우 다른 머신 러닝에 비해 과적합의 가능성이 크다는 단점을 가지고 있다. 주요 하이퍼 파라미터로는 num_leaves, min_date_in_leaf 및 max_depth 등이 있다.

Ⅳ. 데이터 분석 및 실험 환경

4.1 WSN-DS 데이터 셋

WSN-DS[7] 데이터 셋은 NS-2 네트워크 시뮬레이션을 이용하여 구성된 WSN 시뮬레이션을 통하여 생성된 데이터 셋으로 19가지의 Feature와 네가지의 공격 유형으로 이루어져 있다. 네가지의 공격 유형으로는 Blackhole, Grayhole, TDMA 그리고 Flooding 공격이 있으며, 그림 9는 각 공격 유형의 데이터 수와 비율을 도식화했으며, 정확한 수치상 비율은 다음과 같다. Normal 20,067개(약 38.46%), Blackhole 8,000개(약 15.38%), Grayhole 8,0000개(약 15.38%), TDMA 8,000개(약 15.38%) 마지막으로 Flooding 8,000개(약 15.38%)로 데이터간 불균형이 크지 않도록 비율을 맞춘 데이터 셋으로 머신 러닝을 진행하였다.. 해당 데이터 셋을 구성하는 19개의 Feature에 대한 개념과 각 공격 유형의 특징은 아래에서 서술한다.

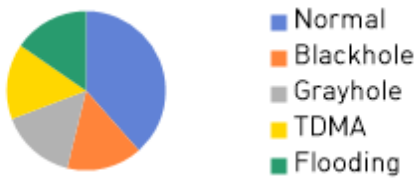

그림 9. WSN-DS 데이터 셋에서 각 공격 유형의 비율
Fig. 9. Proportion of each attack type in the WSN-DS data set

4.1.1 공격 유형

Blackhole : LEACH 프로토콜을 사용하는 네트워크 구조에서 Blackhole 공격은 공격자가 CH로 지정되는 노드 중 하나를 가로챈 후 본인을 CH로 광고하여 네트워크에 영향을 미치는 DoS 공격 유형 중 하나이다. 해당 CH에 합류한 모든 노드는 BS로 전송해야 하는 패킷을 공격자에게 전송하게 된다. 멤버 노드들로부터 패킷을 전달받은 공격자는 해당 패킷들이 BS로 도달하지 못하도록 전달받은 모든 패킷을 드롭하게 된다.

Grayhole : Grayhole 공격은 공격자가 CH를 탈취하고 본인을 CH로 광고하여 영향을 미치는 DoS 공격으로 Blackhole 공격과 유사하게 동작한다. Blackhole 공격과의 차이점은 Blackhole 공격은 전달받은 모드 패킷을 드롭하지만 Grayhole 공격의 경우 전달받은 패킷 중 일부를 선택적 또는 랜덤으로 드롭하여 패킷이 BS로 도달하는 것을 방해한다.

Flooding : Flooding 공격은 공격자가 CH를 탈취하여 CH가 가진 높은 전송력을 기반으로 하여 네트워크에 영향을 미치는 DoS 공격 유형 중 하나로 공격자는 다수의 CH 광고 메시지를 전송한다. 네트워크 내에 속한 센서 노드가 많을수록 많은 수의 광고 메시지를 송신하게 된다. CH가 아닌 일반 노드에 경우 메시지를 수신하는데 많은 에너지를 소모하며, CH를 선택하는데 더 많은 시간을 소비하게 된다. 공격자는 피해를 받은 노드를 속여 다음 CH 선택 시 멀리 떨어진 CH를 선택하도록 유도한다.

TDMA : TDMA 공격은 LEACH 프로토콜에서 CH 선택 후 Scheduling을 진행할 때 발생하는 DoS 공격 유형으로 클러스터 내에 속한 노드들에게 패킷 전송의 순서를 정해주기 위해 각 패킷마다 타임 슬롯을 할당해 줄 때 발생한다. CH 역할을 하는 공격자는 모든 멤버 노드에게 동일한 타임 슬롯을 할당하여 Scheduling 메시지를 전송한다. 동일한 타임 슬롯을 할당받은 멤버 노드들은 동일한 시간에 패킷을 전송하게 되고 결국에는 패킷 간의 충돌이 발생하여 데이터 손실로 이어진다.

4.1.2 Feature

- Node ID : 모든 라운드 및 모든 단계에서 센서 노드를 구별하는 고유 ID (101000~3401088)
- Time : 시뮬레이션의 수행 시간 (50~3600)
- IS_CH : 해당 노드의 클러스터 헤더 여부 (0, 1)
- Who_CH : 해당 노드와 연결된 클러스터 헤더 ID (101000~3401088)
- Dist_To_CH : 노드와 CH 사이의 거리 (0~214)
- ADV_S : CH가 노드로 보낸 광고 메시지의 수 (0~97)
- ADV_R : 노드가 CH에서 수신한 광고 메시지의 수 (0~29)
- JOIN_S : 노드가 CH로 보낸 참여 요청 메시지의 수 (0, 1)
- JOIN_R : CH가 노드에서 수신한 참여 메시지의 수 (0~99)
- SCH_S : 노드로 전송된 TDMA 브로드 캐스트 메시지의 수 (0~99)
- SCH_R : CH가 수신한 예약 메시지의 수 (0, 1)
- Rank : TDMA 스케줄링 순서 (0~99)
- DATA_S : 해당 노드에서 전송한 데이터 패킷의 수 (0~241)
- DATA_R : 해당 노드가 수신한 데이터 패킷의 수 (0~1496)
- Data_Sent_To_BS : BS으로 전송된 메시지의 수 (0~241)
- Dist_CH_To_BS ; CH와 BS 사이의 거리 (0~202)
- Send_Code : 클러스터의 전송 코드 (0~15)
- Consumed_Energy : 소비된 에너지 (0.001~46.1)
- Attack_Type : 공격 유형 또는 Normal

4.2 머신 러닝 하이퍼 파라미터 설정

머신 러닝의 성능을 정확하게 비교하고 분석하기 위해서 각 머신 러닝의 성능을 최대치로 얻을 수 있도록 필요하다. 본 연구에서는 최적의 하이퍼 파라미터를 찾기 위해 Scikit-learn 라이브러리에서 제공하는 GridSearch를 사용하여 각 머신 러닝에 적합한 하이퍼 파라미터를 설정하였다. 여기서 사용한 GridSearch에 경우 설정할 하이퍼 파라미터와 해당 하이퍼 파라미터에 할당할 수치들을 입력하면 모든 하이퍼 파라미터 조합의 점수를 계산하여 결과로 보여준다. 이렇게 생성된 결과에서 가장 높은 점수를 가지는 하이퍼 파라미터 조합을 머신 러닝을 훈련하고 테스트할 때 사용하게 된다.

V. 실험 및 결과

5.1 Feature Importance

트리 기반 알고리즘은 각 노드가 2개의 Child 노드를 가지는 이진 트리를 적절한 불순도를 지표로 하여 모델을 훈련시키는 알고리즘으로 분류 목적일 때는

불순도 지표로 Gini 계수 또는 엔트로피를 이용한다. 이때 불순도를 가장 크게 감소시키는 Feature가 가장 높은 중요도를 가지게 된다.

본 연구에서 이용한 Scikit-learn 라이브러리는 Gini Importance를 이용하여 Feature의 중요도를 측정하게 되는데 이때 필요한 개념이 Gini 불순도이다. Gini 불순도는 해당 노드에서 샘플들이 각각의 클래스에 골고루 분포되어 있을수록 높아지게 된다. 트리 기반 알고리즘은 이러한 불순도를 감소시키는 방향으로 노드를 생성하고 분류를 진행한다.

표 1은 각 머신 러닝에 대해 Scikit-learn 라이브러리에서 제공하는 Feature Importance를 추출할 수 있는 함수인 'tree.feature_importances_'를 이용하여 얻은 결과이다. 표 1을 보게 되면 머신 러닝에 따라서 중요 Feature로 판단하는 순서가 모두 다르다는 것을 알 수 있다. Extra Tree에 경우 Split- point를 지정할 때 무작위로 Feature를 선택하여 훈련하기 때문에 Feature Importance가 계속 변화되어 Feature Importance를 사용하는 의미가 적다.

5.2 Feature Importance

머신 러닝에서 Overfitting(과적합)이란 학습 데이터에 대해 과하게 학습하여 실제 데이터에 대한 오차가 증가하는 현상이다. 이러한 과적합이 발생하는 이유는 다음과 같이 크게 세 가지가 있다. 첫 번째로 일반적인 학습 데이터는 실제 데이터의 부분집합으로 실제 데이터를 모두 수집하는 것이 불가능하다는 점이다. 두 번째는 실제 데이터를 모두 수집하는 것이 가능하여도 모든 데이터를 학습시키기 위한 시간이 측정 불가능한 수준으로 증가할 가능성이 있다. 마지막으로 학습 데이터만 가지고 실제 데이터의 오차가 증가하는 지점을 예측하는 것은 거의 불가능하기 때문이다. 이러한 이유로 과적합을 방지 또는 완화하기 위한 방법으로 최적화와 Cross Validation 등의 방안이 제시되었다.

본 연구에서 훈련 모델의 과적합 방지를 위해 훈련용 데이터 셋 중 일부를 Validation 데이터로 분리하여 훈련용 데이터 셋으로 모델을 훈련하고 Validation 데이터 셋을 이용하여 모델을 검증하였다.

이를 위해 데이터 셋을 생성된 시간 순으로 정렬하였으며, 전체 데이터 셋의 50%는 훈련, 20%는 Validation 그리고 30%는 Test 순으로 머신 러닝의 훈련을 진행하였다.

5.3 Feature Importance

모델의 성능 평가란 실제 값과 모델에 의해 예측 또는 분류된 값을 비교하여 두 값의 오차를 구하는 것으로 모델 평가의 목적은 과적합을 방지하고 최적의 모델을 찾기 위해 실시된다. 모델링의 목적과 목표 변수 유형에 따라 다른 평가 지표를 사용해야 하며, 분류가 목적인 경우 Accuracy, Precision, Recall 그리고 F1-score 등의 평가 지표가 존재하며, 본 논문에서는 그중 Precision, Recall 그리고 F1-score를 사용하며 각 성능 평가 지표의 특징은 다음과 같다;

Precision : 모델이 Positive로 판단한 데이터 중, 실제 Positive로 판단된 데이터의 비율로 실제 Negative인 데이터를 Positive로 잘못 판단하게 되었을 때 큰 문제가 발생될 경우 중요한 지표로 간주된다.

$$(\mathrm{Precision}) = \frac{TP}{TP+FP}$$

Recall : 실제 Positive인 데이터 중 올바르게 Positive로 판단한 데이터의 비율로 실제로 Positive인 데이터의 예측을 Negative로 잘못 판단하게 될 때 큰 영향이 발생되는 경우에 중요한 지표로 간주된다.

$$(Recall) = \frac{TP}{TP+FN}$$

표 1. 머신 러닝 별 Feature Importance
Table 1. Feature Importance by Machine Learning

	DT	RF	ET	XGBoost	LightGBM
1st	SCH_S	ADV_S	Feature Importance 의미 없음	ADV_S	SCH_S
2nd	IS_CH	SCH_S		SCH_S	ADV_S
3rd	ADV_S	Consumed_Energy		Data_sent_To_BS	IS_CH
4th	Consumed_Energy	IS_CH		DATA_R	Data_Sent_To_BS
5th	Dist_CH_To_BS	Data_sent_To_BS		IS_CH	DATA_R

F1-score : Precision와 Recall을 결합하여 생성한 지표로 Precision와 Recall이 어느 한쪽으로 치우치지 않는 수치를 나타낼 때 높은 수치를 가지게 된다.

$$(F1-Score) = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

5.4 Feature Importance

표 2, 표 3, 표 4는 다섯 가지의 머신 러닝을 이용하여 모델을 훈련시키고 Test 데이터 셋을 이용해 훈련된 모델을 테스트한 결과를 각각 Precision, Recall 및 F1-Score를 이용해 표로 정리한 내용이다. 전체적으로 단일 트리 구조의 훈련 방식을 가진 Decision Tree의 성능이 가장 낮게 평가되었으며, 가중치 및 병렬 학습을 통해 훈련하는 XGBoost의 성능이 가장 높게 평가되었다. 공격 유형을 기준으로 보았을 때, 대부분이 공격자의 영향을 받지 않는 멤버 노드로 구성된 Normal의 분류 정확도가 가장 높게 평가되었다. 공격 방식이 유사한 Blackhole과 Grayhole의 경우 서로 분류 정확도에 영향을 주어 다른 공격 유형에 비해 분류 정확도가 낮게 평가되었다. TDMA와 Flooding의 경우 특정 Feature의 중요도가 높을 경우 성능이 높게 측정되었다. 이에 관한 자세한 비교 분석은 섹션 5.5에서 설명한다.

표 2. Precision 결과
Table 2. Result of Precision

	Normal	Blackhole	Grayhole	TDMA	Flooding
DT	0.98	0.84	0.85	0.96	0.93
RF	0.99	0.88	0.92	0.97	0.97
ET	0.99	0.89	0.90	0.95	0.97
XGBoost	0.99	0.93	0.95	0.99	0.98
LightGBM	0.99	0.93	0.92	0.99	0.96

표 3. Recall 결과
Table 3. Result of Recall

	Normal	Blackhole	Grayhole	TDMA	Flooding
DT	0.98	0.88	0.84	0.94	0.97
RF	0.99	0.92	0.90	0.97	0.98
ET	0.99	0.89	0.91	0.97	0.97
XGBoost	0.99	0.94	0.92	0.99	0.99
LightGBM	0.99	0.91	0.92	0.98	0.97

표 4. F1-Score 결과
Table 4. Result of F1-Score

	Normal	Blackhole	Grayhole	TDMA	Flooding
DT	0.98	0.86	0.85	0.95	0.95
RF	0.99	0.90	0.91	0.97	0.98
ET	0.99	0.89	0.90	0.96	0.97
XGBoost	0.99	0.94	0.94	0.98	0.99
LightGBM	0.99	0.92	0.92	0.98	0.97

5.5 머신 러닝 성능 비교

각 머신 러닝 알고리즘의 Feature Importance와 머신 러닝 테스트 결과 중 F1-Score를 그래프로 표현한 그림 10과 Feature Importance 결과를 정리한 표 1을 이용하여 머신 러닝의 성능을 비교하고 결과를 분석하고자 한다.

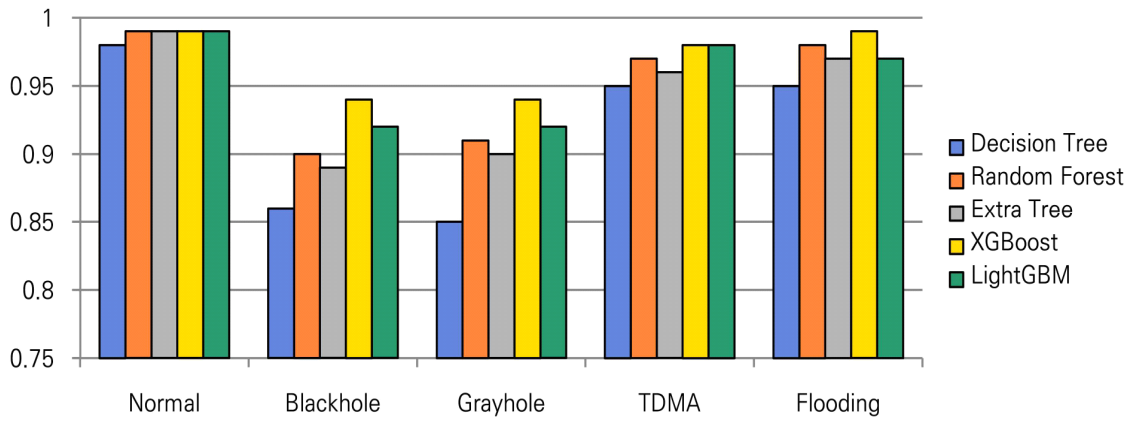

그림 10. F-1 Score 그래프
Fig. 10. F-1 Score Graph

5.5.1 Normal

그림 10의 그래프에 나와있는 것처럼 Decision Tree가 다른 머신 러닝 알고리즘에 비해 정확도가 가장 낮은 것을 볼 수 있다. 이는 Decision Tree가 단일 Tree 구조로 되어 있어 다른 머신 러닝 알고리즘보다 정확도가 낮은 편이기 때문이다. 또한 Normal의 분류 결과는 다른 Class에 비해 평균적인 정확도가 높은 편이다. 이는 공격이 클러스터 헤더에 한정되어 가해지기 때문에 데이터 셋의 대부분을 이루고 있는 일반 노드의 경우 모두 Normal로 분류되기 때문이다.

5.5.2 Blackhole

Blackhole 공격은 공격자가 가로챈 CH에 속한 멤버 노드들로부터 전송받은 데이터를 모두 드롭하는 공격으로 DATA_R과 Data_Sent_To_BS의 값이 모두 0이 되는 특징을 가지고 있다. 또한 베이스 스테이션으로 전송하는 데이터가 없기 때문에 Consumed_Energy의 값도 작아지게 된다. 따라서 위에서 언급한 세 가지 Feature 들은 Blackhole 공격 분류에 있어 매우 중요하다. 이를 통해 결과를 분석하면 Decision Tree의 경우 단일 Tree 구조의 한계와 중요 Feature 중 Consumed_Energy만을 중요하게 판단하여 가장 성능이 떨어진다. Extra Tree의 경우 다중 Tree 구조로 인해 Decision Tree보다는 성능이 좋지만 랜덤하게 Feature를 선택하여 훈련을 진행하기 때문에 다른 머신 러닝에 비해 성능이 낮다. Random Forest의 경우 세 개의 중요 Feature 중 두 개의 Feature를 높게 평가하였지만 Consumed_Energy가 다른 두 Feature에 비해 중요도가 낮고 Boosting에 비해 훈련 방식의 성능이 떨어져 Boosting 계열의 머신 러닝보다 성능이 낮은 결과를 보였다. LightGBM과 XGBoost 모두 같은 Feature를 중요하게 판단하였지만 XGBoost가 더 중요도를 높게 판단하였으며, 훈련 방식의 차이로 인해 XGBoost의 성능이 가장 높게 측정되었다.

5.5.3 Grayhole

Grayhole의 경우 Blackhole과 동일한 Feature를 분류하는데 중요하게 보고 있어 Blackhole의 분류 결과와 유사한 양상을 보여준다.

5.5.4 TDMA

TDMA 공격은 공격자가 가로챈 CH에 속한 모든 멤버 노드에게 같은 타임 슬롯을 배분하여 충돌을 유발하는 공격으로 SCH_S의 값이 1보다 크거나 같으며 수신한 참여 메시지의 수를 뜻하는 JOIN_R의 값과 동일하다는 특징을 가진다. 또한 데이터 간의 충돌이 발생하여 데이터의 손실이 발생할 가능성이 있어 DATA_R과 Data_Sent_To_BS의 값이 동일하지 않을 수도 있다는 특징을 가진다. 이를 통해 결과를 분석하면 Decision Tree와 Extra Tree는 다른 공격 유형과 같은 이유로 낮은 성능을 보인다. Random Forest와 XGBoost의 경우 SCH_S의 중요도를 2순위로 판단했지만 XGBoost의 경우 DATA_R과 Data_Sent_To_BS를 중요 Feature로 판단하여 더 높은 성능을 보여준다. 마지막으로 LightGBM의 경우 SCH_S의 중요도를 1순위로 판단하였지만 XGBoost에 비해 훈련 중 사용되는 Split 알고리즘의 성능이 떨어져 XGBoost와 동일한 성능을 보여주었다.

5.5.5 Flooding

Flooding 공격은 공격자가 CH를 가로챈 후 다량의 광고 메시지를 브로드 캐스팅하는 공격으로 ADV_S의 값이 1보다 크거나 같으며 다량의 광고 메시지를 전송하므로 Consumed_Energy의 값이 커지는 특징을 가진다. 이를 통해 결과를 분석하면 Decision Tree와 Extra Tree의 경우 다른 공격 유형들과 동일하게 낮은 성능을 보여준다. LightGBM의 경우 중요 Feature인 ADV_S의 중요도를 다른 두 머신 러닝에 비해 낮게 판단하고 Consumed_Energy의 경우는 중요 Feature로 선정하지 않아 다른 두 머신 러닝에 비해 성능이 낮게 측정되었다. Random Forest의 경우 XGBoost와 동일하게 ADV_S의 중요도를 1순위로 측정하였으나 Boosting 방식으로 모델을 훈련하는 XGBoost에 비해 낮은 성능을 보여주었다.

Ⅵ. 결 론

본 논문은 논문은 WSN에 가해지는 DoS 공격을 탐지하는 침입 탐지 시스템에서 머신 러닝을 활용하는데 있어 적합한 머신 러닝 알고리즘에 대한 성능 비교 및 분석을 수행하였다. 단일 Tree 알고리즘을 사용하는 Decision Tree는 노드를 분할할 때 2진 분할을 사용하여 데이터 셋에서 다루는 Feature의 수와 분류를 해야하는 Class가 많을 수록 Tree의 크기가 커지고 이에 따라 과적합 가능성이 높아진다. 이로 인해 Tree는 일정 크기 이상으로 커지지 않는며, 한 번 잘못 분류된 데이터를 재분류할 수 없어 가장 낮은 성능을 보여준다. Random Forest와 Extra Tree의 경우 Decision Tree에서 만드는 Tree보다 작은 크기의

Tree를 다수 생성하여 각각의 Tree의 훈련 결과들을 토대로 다수결 방식을 이용해 전체적인 훈련을 진행한다. Random Forest와 Extra Tree는 데이터 셋 분할 방식, 노드 분할 방식 그리고 Feature Importance에 의해 정확도의 차이가 발생한다. 원본 데이터를 그대로 다수의 Tree의 훈련데이터로 사용하는 Extra Tree와 달리 중복을 허용하여 부분 데이터 집합을 만드는 Random Forest의 경우 각 Tree가 유니크한 데이터 셋을 가지게 된다. 또한 노드 분할 시에 Extra Tree는 Feature Importance와 상관 없이 무작위로 Feature를 선정하여 노드를 분할하지만 Random Forest는 최적의 노드 분할 방식을 사용해 Extra Tree보다 정확도가 높다. 마지막으로 가중치를 사용하는 XGBoost와 LightGBM의 경우 Random Forest와 달리 첫 번째 Tree에서 잘 못 분류된 데이터에 가중치를 두어 두 번째 Tree에서 개선되는 방식을 사용하여 Random Forest보다 정확도가 높다. LightGBM과 XGBoost의 가장 큰 차이는 노드 분할 방식으로 균형 Tree 분할을 하는 XGBoost는 한 번 분류가 잘못되어도 괜찮지만 리프 중심 Tree 분할을 하는 LightGBM의 경우 한 번 잘 못 분류된 데이터는 재분류가 되지 않기 때문에 XGBoost가 가장 좋은 성능을 보여주었다.

향후 연구에서는 본 논문에서 사용한 WSN-DS 데이터 셋을 기반으로 하여 Feature Selection 알고리즘에 대한 연구를 수행할 예정이다. 또한 Tree 기반의 머신 러닝 방식 이외의 머신 러닝 방식에 대한 비교 분석을 수행할 예정이다.

References

[1] G. Meena, et al., “A review paper on IDS classification using KDD 99 and NSL KDD dataset in WEKA,” *2017 Comptelix*, Jaipur, India, Jul. 2017. (https://doi.org/10.1109/COMPTELIX.2017.8004032)

[2] T.-T.-H. Le, et al., “An effective classification for DoS attacks in wireless sensor networks,” *2018 ICUFN*, Prague, Czech Republic, Jul. 2018. (https://doi.org/10.1109/ICUFN.2018.8436999)

[3] R. D. Ravipati, et al., “Intrusion detection system classification using different machine learning algorithms on KDD-99 and NSL-KDD datasets - A Review Paper,” *IJCSIT*, vol. 11, no. 3, 2019. (http://dx.doi.org/10.2139/ssrn.3428211)

[4] L. M. Ibrahim, et al., “A comparison study for intrusion database (KDD99, NSL-KDD) based on self organization map (SOM) artificial neural network,” *J. Eng. Sci. and Technol.*, vol. 8, no. 1, 2013.

[5] A. B. Abhalea, et al., “Supervised machine learning classification algorithmic approach for finding anomaly type of intrusion detection in wireless sensor network,” *Optical Memory and Neural Networks,* vol. 29, 2020. (https://doi.org/10.3103/s1060992x20030029)

[6] A. Zien, et al., “The feature importance ranking measure,” *Mach. Learn. and Knowledge Discovery in Databases,* vol. 5782, 2009. (https://doi.org/10.1007/978-3-642-04174-7_45)

[7] I. Almomani, et al., “WSN-DS: A dataset for intrusion detection systems in wireless sensor networks,” *Recent Advances in Security and Privacy for Wireless Sensor Networks 2016,* vol. 2016, 2016. (https://doi.org/10.1155/2016/4731953)

[8] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, 2001. (https://doi.org/10.1023/A:1010933404324)

[9] P. Geurts, et al., “Extremely randomized trees,” *Mach. Learn.*, vol. 63, 2006. (https://doi.org/10.1007/s10994-006-6226-1)

[10] T. Chen, et al., “XGBoost: A scalable tree boosting system,” *KDD ’16: Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Aug. 2016. (https://doi.org/10.1145/2939672.2939785)

[11] M. R. Machado, et al., “LightGBM: An effective decision tree gradient boosting method to predict customer loyalty in the finance industry,” *ICCSE 2019,* 2019. (https://doi.org/10.1109/iccse.2019.8845529)

임 성 욱 (SungWook Lim)

2019년 8월 : 숭실대학교 컴퓨터공학과 학사

2019년 9월~현재 : 숭실대학교 정보통신융합학과 석사과정

[ORCID:0000-0002-8741-8105]

유 명 식 (Myungsik Yoo)

1989년 2월 : 고려대학교 전자공학과 학사

1991년 2월 : 고려대학교 전자공학과 석사

2000년 6월 : SUNY at Buffalo Dept. of EE 박사

2000년 9월~현재 숭실대학교 전자정보공학부 교수

[ORCID:0000-0002-5578-6931]