# 기계학습 알고리즘을 이용한 침입탐지 성능향상 방안연구

정선욱*

*KAIST 소프트웨어 대학원

**요 약**

본 연구는 기존의 보안 분야에서 네트워크 침입탐지 방법으로 널리 사용되고 있는 Rule 기반의 침입탐지 기법과 비정상 탐지 기반의 침입탐지 기법의 오탐 및 미탐에 대한 단점을 보완하고 유해패킷 유입에 대한 좀 더 정확한 예측 모델을 구현함으로써 네트워크 공격에 대한 선제적 방어체계를 마련하는데 그 목적이 있다.

이를 위해 실제 공격상황에서의 네트워크 데이터를 pre-processing하고, 기계학습 알고리즘을 통해 유해패킷에 대한 예측모델을 구현함으로써 각 시나리오 별 모델에 대한 성능을 측정하였다.

구현결과 첫째, data set별 시나리오에서는 Filter방식과 CFS방식의 알고리즘 중 CFS data deduction 알고리즘이 좀 더 나은 성능을 보였고 둘째, 단일 알고리즘 중에서는 Decision Tree 알고리즘이 가장 뛰어난 성능을 보였으며 셋째, 혼합 알고리즘 시나리오에서는 Support Vector Machine 알고리즘과 Bayesian Classifier 의 혼합 알고리즘이 가장 높은 accuracy를 나타내었다.

그러나 accuracy 외 recall과 같은 다른 성능지표에서는 각기 다른 결과를 보여 현업에 적용 시에는 보안장비별 특성을 고려하여야 함을 나타내었다.

## I. 서론

최근에는 Big data 기반기술을 활용한 보안 로그 통합검색 및 분석방법이 널리 활용되고 있다. 그러나 이러한 플랫폼의 변화에도 불구하고 보안 데이터를 분석하는 기법은 signature DB나 SNORT를 이용한 Rule-based 기반 침입탐지방식이나 정상네트워크 상태와의 편차를 분석하여 정상 / 비정상 상태를 판단하는 Anomaly-based 기반 침입탐지방식을 통한 탐지 방식이 주를 이루고 있다. 이는 알려지지 않은 공격에 대한 선제적 대응이 어렵고, 공격 패턴에 대한 rule을 일일이 setting해야 하는 번거로움과 함께 보안인력 투입에 대한 효율성 또한 떨어지는 결과를 가져오게 된다.

이에, 공격패턴에 대한 데이터를 기계학습 알고리즘을 통해 스스로 학습하고, 유해여부를 판단할 수 있는 predict modeling 시스템을 빅데이타 플랫폼 기반에서 구현하고 활용할 수 있는 방안을 연구하였다.

### 1.1 Data Collection & Pre-processing

본 연구에서 기계학습을 위해 사용된 data set은 KDD'99의 refined version인 NSL-KDD Dataset으로서 실제 네트워크 공격상황에서 추출될 수 있는 41개의 네트워크 feature를 나타내고 있으며 각 패킷의 유해/정상여부에 대한 label을 가지고 있다.

이 Data set은 프로토콜 타입, 서비스, 송수신 bytes등 Transport Layer에서 수집될 수 있는 네트워크 basic feature들과 login 시도횟수, root access 횟수 등 Application Layer단계에서 추출할 수 있는 contents feature 들을 가지고 있으며, 알고리즘의 성능을 측정하기 위해 전체 약 125,000 row가량의 data set을 각 feature 별로 분리, csv file로 변환하고 cross-validation을 위해 10개의 subset으로 sampling하는 data pre-processing 작업을 하였다.

### 1.2 Data Dimension Deduction

다음 단계에서는 데이터의 차원축소 과정을 진행하였는데 이는 서로 연관성이 적은 feature(noise)를 제거하고 over-fitting을 피함으로써 분류에 따른 시간과 공간을 절약하여 결과적으로 데이터 마이닝에 대한 성능을 향상시키는 데에 그 목적이 있다.

데이터의 차원축소방법은 여러 데이터 들이 하나의 분포를 이룰 때 이 분포의 주성분을 분석해 주는 PCA(Principal Component Analysis)방식과 각 feature간의 연관성을 수치로 나타내는 방법인 CFS(Correlation based Feature Selection)방식이 있는데 본 연구에서는 CFS 방식을 사용하였으며 그 중에서도

카이스퀘어 통계치(Chi-square statistics) :

$$\chi^2(Z1;Z2) = \sum_{i=1}^{I}\sum_{j=1}^{J}\frac{(O_{z1_i,z2_j} - E_{z1_j,z2_j})^2}{E_{z1_i,z2_j}}$$

방식을 이용하였으며, data deduction 알고리즘별 성능을 비교하기 위해 Filter 방식도 같이 병행 실험하였다. 각 data deduction 방식별 feature set은 다음과 같다.

**Table1. Filter 방식 feature set**

| No | Feature | Sample data |
|---|---|---|
| 1 | Duration | 0 |
| 2 | Protocol | tcp |
| 3 | Services | ftp |
| 4 | Src_bytes | 420 |
| 5 | Dst_bytes | 0 |
| 6 | Num_failed_login | 0 |
| 7 | Root_shell | 3 |
| 8 | Num_root | 1 |
| 9 | Num_file_creation | 1 |
| 10 | Num_shells | 1 |

**Table2. Filter 방식 feature set**

| No | Feature | Correlation |
|---|---|---|
| 1 | Duration | 0.691082 |
| 2 | Dst_bytes | 0.490070 |
| 3 | Num_failed_logins | 0.438821 |

### 1.3 Machine Learning Algorithm scenario

이전 단계에서 전처리와 차원축소가 이루어진 data set을 기계학습 알고리즘 training을 위해 다음의 학습 순서 및 알고리즘을 적용하여 실험을 진행하였다.

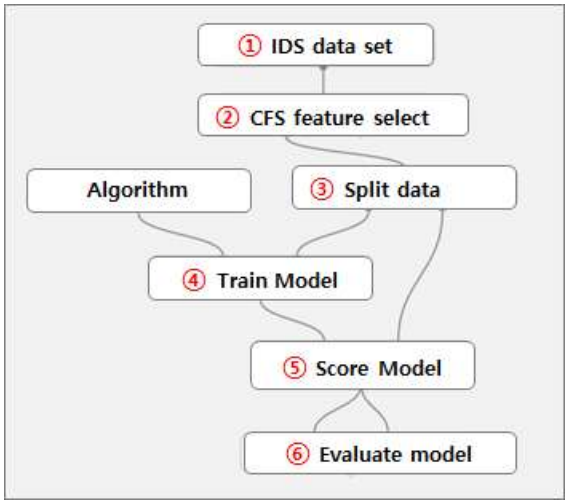


**Fig. 1. Learning Flow**

① CSV 형식의 학습 데이타를
② CFS방식으로 데이터 차원축소를 한 후
③ Training, Test를 위해 data를 분할하고
④ 알고리즘 별 모델을 학습한 후에
⑤ 학습 모델에 대한 scoring과
⑥ 학습 모델에 대한 성능 평가
의 순서로 기계학습 과정을 진행하였다.

**Table3. 기계학습 알고리즘 시나리오(단독)**

| No | Algorithm |
|---|---|
| 1 | Support Vector Machine |
| 2 | Decision Tree |
| 3 | Bayesian classifier |
| 4 | Neural Network |

**Table4. 기계학습 알고리즘 시나리오(혼합)**

| No | Algorithm | Combined |
|---|---|---|
| 1 | SVM, BC | SVM+BC |
| 2 | BC,SVM | BC+SVM |
| 3 | DT,BC | DT+BC |
| 4 | DT,SVM | DT+SVM |

1.4 Evaluation

기계학습 알고리즘 학습 및 평가를 위해선 통계/분석을 위한 오픈소프트웨어인 R Studio를 사용하였으며, 알고리즘의 성능평가는 다음의 지표를 사용하였다.

**Table5. 성능지표**

| Factor | Formula |
|---|---|
| Accuracy | (TP+TN)/(TP+FP+FN+TN) |
| Precision | TP/(TP+FP) |
| Recall | TN/(TP+FN) |
| F1 Score | $\frac{precision \cdot recall}{precision + recall}$ |

각 시나리오별 구현결과는 다음과 같다.

(지표별 비교를 위해 백분율 수치를 적용함)

**Table6. dataset 별 성능(Filter방식)**

| Algorithm | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| SVM | 99.1425 | 99.1268 | 99.9840 | 0.9956 |
| DT | **99.9920** | **99.9841** | **99.9920** | **0.9999** |
| BC | 99.1346 | 99.1188 | 99.9840 | 0.9956 |
| NN | 99.1346 | 99.1110 | 99.9760 | 0.9955 |

**Table7. dataset 별 성능(CFS방식)**

| Algorithm | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| SVM | 99.0738 | 99.0476 | 99.9733 | 0.9952 |
| DT | **99.9865** | **99.9061** | 99.9188 | **0.9995** |
| BC | 99.0738 | 99.0607 | **99.9866** | 0.9953 |
| NN | 99.0738 | 99.0345 | 99.9600 | 0.9951 |

**Table8. 알고리즘 별 성능(혼합)**

| Algorithm | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| SVM+BC | **99.7868** | **99.7751** | **99.9866** | **0.9989** |
| BC+SVM | 99.0738 | 99.0476 | 99.9733 | 0.9952 |
| DT+BC | 99.7735 | 99.7619 | **99.9866** | 0.9988 |
| DT+SVM | 99.0738 | 99.0345 | 99.9600 | 0.9951 |

**Table9. 학습 데이터 사이즈별 성능**

| Algorithm | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| **60%** | 99.4856 | 99.3433 | 99.9823 | 0.9941 |
| **70%** | 99.5728 | 99.3472 | 99.9840 | 0.9946 |
| **80%** | 99.6828 | 99.3512 | 99.9851 | 0.9952 |
| **90%** | 99.8155 | 99.3554 | 99.9864 | 0.9958 |

구현결과를 요약하면 다음과 같다

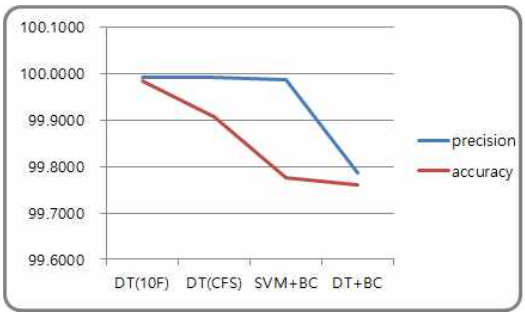


**Fig. 2. Precision & Accuracy**

Precision과 Accuracy모두 Filter 방식의 Decision Tree 모델이 높은 성능을 나타내었으나 CFS 방식과의 편차가 크지 않음(99.9919/99.9864)을 고려했을 때 feature 개수가 적은 CFS방식이 좀 더 효율적이라 볼 수 있다.

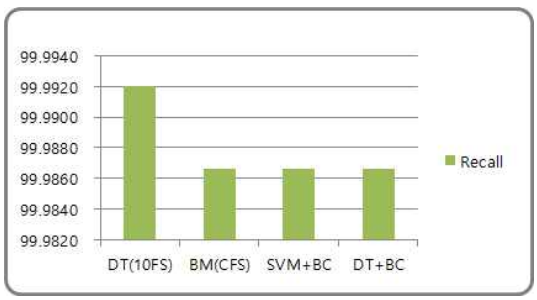


**Fig. 3. Recall**

Recall 지표에 있어서는 Filter방식의 Decision Tree모델이 가장 우수한 성능을 보여 미탐(1-Y인 것중 Y로 예측한 비율)확률이 가장 낮아 모델 중 가장 우수한 성능을 나타내었다.

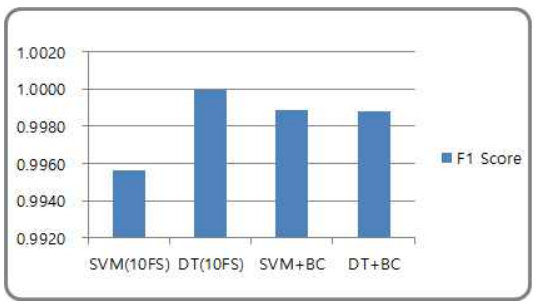


**Fig. 4. F1 Score**

F1 Score 지표에 있어서는 Filter방식의 Decision Tree모델과 SVM+BC의 혼합 알고리즘 모델이 가장 우수한 성능을 가짐으로서 Precision과 Recall의 조화평균이 골고루 높은 결과를 나타내었다.

## II. 결론

본 논문에서는 침입탐지 방식에 있어 기존에 널리 사용되었던 추론· 통계방식의 침입탐지 방식 대신 데이터마이닝 알고리즘을 적용한 기계학습 기반의 침입탐지 모델을 제안하여 구현하였으며 이 기법은 그 동안 유해공격의 패턴별로 rule이나 모델을 작성하였던 방식과는 달리 보안패킷데이타로부터 직접 그 연관성을 찾아 비정상상태의 패킷을 탐지해주는 방식으로 향후 빅데이타 기반의 통합보안로그 검색 시스템으로의 플랫폼 변환 시 좀 더 정확하고 자동화된 침입탐지 방법을 제공할 것으로 기대된다.

본 논문에서는 유해패킷을 예측하기 위해 실제 공격상황에서의 dataset으로 여러 가지 시나리오 및 알고리즘을 적용하여 성능을 측정하였으며 그중 Decision Tree모델이 실험 알고리즘 모델 중 가장 우수한 정확도(99.9920 %)를 보였다. 그러나 향후 현업에서 기계학습 알고리즘 적용시에는 미탐/오탐에 대한 민감도 및 학습데이터량 감소분에 대한 가중치 적용등 정확성과 성능, 시간측면에 대한 상호득실(trade-off)관계를 잘 고려하여 적용하여야 할 것이다.

향후에는 좀 더 정확한 예측모델의 구현을 위해 기계학습 방식과 기존의 signature 기반의 모델을 결합시키는 방법에 대한 연구와 더블어 기계학습 알고리즘 적용시 계산비용(computational cost)의 감소에 대한 추가연구가 필요할 것으로 예상된다.

## [참고문헌]


[1] L. J. Institute of Engineering and Technology, Ahmedabad, Gujarat, India, Effective Intrusion Detection System using Data Mining Techique , June, 2015.

[2] L Dhanabal, Dr.S.P.Shantharajah, A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithm, June 2015

[3] Omar Y.Al-Jarrarh, Omar Alhussein, Data Randomization and Cluster-Based Partitioning for Botnet Intrusion Detection, Oct, 2015

[4] Vipin Das 1, Vijaya Pathak2, Network Intrusion Detection System Based on Machine Learning Algorithms, Dec 2010

[5] Aleksandar Lazarevic, Levent Ertoz, A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection

[6] Shui Yu,Xiaodong Lin,Jelena, Networking for Big Data