J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

MLOps를 위한 효율적인 AI 모델 드리프트 탐지방안 연구☆

A Study on Efficient AI Model Drift Detection Methods for MLOps

이 예 은[1] 이 태 진[1*]
Ye-eun Lee Tae-jin Lee

요 약

오늘날 AI(Artificial Intelligence) 기술이 발전하면서 실용성이 증가함에 따라 실생활 속 다양한 응용 분야에서 널리 활용되고 있다. 이때 AI Model은 기본적으로 학습 데이터의 다양한 통계적 속성을 기반으로 학습된 후 시스템에 배포되지만, 급변하는 데이터의 상황 속 예상치 못한 데이터의 변화는 모델의 성능저하를 유발한다. 특히 보안 분야에서 끊임없이 생성되는 새로운 공격과 알려지지 않은 공격에 대응하기 위해서는 배포된 모델의 Drift Signal을 찾는 것이 중요해짐에 따라 모델 전체의 Lifecycle 관리 필요성이 점차 대두되고 있다. 일반적으로 모델의 정확도 및 오류율(Loss)의 성능변화를 통해 탐지할 수 있지만, 모델 예측 결과에 대한 실제 라벨이 필요한 점에서 사용 환경의 제약이 존재하며, 실제 드리프트가 발생한 지점의 탐지가 불확실한 단점이 있다. 그 이유는 모델의 오류율의 경우 다양한 외부 환경적 요인, 모델의 선택과 그에 따른 파라미터 설정, 그리고 새로운 입력데이터에 따라 크게 영향을 받기에 해당 값만을 기반으로 데이터의 실질적인 드리프트 발생 시점을 정밀하게 판단하는 것은 한계가 존재하게 된다. 따라서 본 논문에서는 XAI(eXplainable Artificial Intelligence) 기반 Anomaly 분석기법을 통해 실질적인 드리프트가 발생한 시점을 탐지하는 방안을 제안한다. DGA(Domain Generation Algorithm)를 탐지하는 분류모델을 대상으로 시험한 결과, 배포된 이후 데이터의 SHAP(Shapley Additive exPlanations) Value를 통해 Anomaly score를 추출하였고, 그 결과 효율적인 드리프트 시점탐지가 가능함을 확인하였다.

☞ 주제어 : 인공지능, 머신러닝 모델, 드리프트 탐지, 설명가능한 인공지능, 머신러닝 운영

ABSTRACT

Today, as AI (Artificial Intelligence) technology develops and its practicality increases, it is widely used in various application fields in real life. At this time, the AI model is basically learned based on various statistical properties of the learning data and then distributed to the system, but unexpected changes in the data in a rapidly changing data situation cause a decrease in the model's performance. In particular, as it becomes important to find drift signals of deployed models in order to respond to new and unknown attacks that are constantly created in the security field, the need for lifecycle management of the entire model is gradually emerging. In general, it can be detected through performance changes in the model's accuracy and error rate (loss), but there are limitations in the usage environment in that an actual label for the model prediction result is required, and the detection of the point where the actual drift occurs is uncertain. there is. This is because the model's error rate is greatly influenced by various external environmental factors, model selection and parameter settings, and new input data, so it is necessary to precisely determine when actual drift in the data occurs based only on the corresponding value. There are limits to this. Therefore, this paper proposes a method to detect when actual drift occurs through an Anomaly analysis technique based on XAI (eXplainable Artificial Intelligence). As a result of testing a classification model that detects DGA (Domain Generation Algorithm), anomaly scores were extracted through the SHAP(Shapley Additive exPlanations) Value of the data after distribution, and as a result, it was confirmed that efficient drift point detection was possible.

☞ keyword : Artificail Inteligence, Machine Learning Model, Drift Detection, XAI, MLOps

1. 서 론

최근 몇 년 동안 머신러닝 기술이 발전하면서 금융, 제조, 의료, 보안분야에 이르기까지 일상생활 속에서 다양하게 사용됨에 따라 생성되는 데이터의 양 또한 지속해서 증가하고 있다. 특히 보안 분야에서는 서비스거부공격, 스팸, 피싱 등 다양한 유형의 사이버공격은 몇 년간 기하급수적인 속도로 증가하게 되면서 더욱더 혁신적이

[1] Department of Information Security, Hoseo University., Chungnam, 31499, Korea.
* Corresponding author (kinjecs0@gmail.com)
[Received 4 May 2023, Reviewed 24 May 2013(R2 12 September 2023), Accepted 25 September 2013]
☆ 이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2022-0-000-89, 사이버공격 대응을 위한 Life-cycle 기반 공격그룹 식별 및 유형 분석 기술 개발)

고 효율적으로 처리하고자 머신러닝 기술이 도입되고 있다[1]. 이처럼 빅데이터 시대에서 원활하게 시스템을 운영하기 위해 고성능의 모델도 중요하지만, 모델이 생성되고 배포된 이후까지 전체적인 Lifecycle을 관리하는 것이 점점 중요해지는 추세이다. 이처럼 머신러닝은 수많은 데이터의 잠재력을 활용하는 기술로써 실제 환경에서 사용될 때 시간이 지나서도 신뢰할 수 있는 성능을 유지하는 능력이 중요해짐에 따라 MLOps 개념에 관한 관심이 증가하고 있다. MLOps는 생산적인 머신러닝을 설계하고 유지관리하는 것을 목적으로 기본적으로 머신러닝에 대한 통찰력을 제공하고 실제 환경에서 지속해서 좋은 성능을 유지할 수 있는 모델 관리 프로세스를 의미한다. 특히 네트워크 공격을 모니터링 및 분석하는 과정에서 알려지지 않은 예상치 못한 공격이 실시간으로 짧은 시간 동안 이뤄지는 상황에서 발생하는 수천 개의 데이터를 다루기 위해서는 모델 관리가 절대적으로 필수이다. 같은 네트워크 공격이라도 시간이 흐름에 따라 다양해지는 공격자의 공격 방법으로 공격 데이터의 분포도 변화하게 된다. 예를 들어 도메인 이름을 주기적으로 동적으로 생성하는 사이버공격인 DGA의 경우 수천 개의 도메인이 생성되어 시스템을 감염시키며 이후 방어의 추적을 회피하기 위해 지속해서 새로운 도메인을 생성하게 된다. 이런 상황 속에서 새로운 공격으로 인한 모델의 성능저하 문제는 사회에 있어 큰 위험을 초래하게 되므로 문제를 방지하기 위해 관리자는 데이터의 재구성 및 모델의 재학습 등의 후속 조치를 통해 모델을 지속해서 운영할 수 있어야 한다.

운영 중인 머신러닝 시스템의 성능이 저하되고 수용할 수 없게 되는 경우를 드리프트가 발생했다고 한다. 드리프트는 모델 드리프트, 데이터 드리프트, 개념 드리프트 등으로 다양한 원인으로 드리프트가 발생할 수 있다. 가장 흔한 이유로는 데이터 분포의 변화를 원인으로 기존 학습된 모델의 품질이 저하될 수 있다. 이를 막기 위해 지속해서 모델을 모니터링을 하는 과정을 통해 신속하고 정확한 시점에 조치를 할 수 있도록 Drift Signal을 빠르게 파악하는 것이 중요하다. 그러므로 배포된 모델의 상태를 모니터링하면서 장기간 좋은 성능을 유지할 수 있도록 효율적인 드리프트 탐지 방법이 필요하다.

일반적으로 모델의 정확도와 오류율이 초기성능과 비교해 일정 수준 이상의 차이가 발생하면 드리프트가 발생하였음을 의심할 수 있다. 오류율의 경우 모델의 예측값이 실제 정답과 얼마나 차이가 있는지 수치화한 값으로 드리프트 탐지 목적으로 많이 사용된다. 그 이유는 모델이 훈련되는 과정에서 데이터에 의존하게 되는 특징이 존재하기 때문이다. 따라서 데이터 분포가 변경될 때 모델 예측의 판단기능이 더는 올바르게 작동하지 않게 되고 이에 따라 발생하게 되는 오류율을 측정함으로써 드리프트가 발생했음을 확인할 수 있다. 하지만 성능에 의한 드리프트 탐지를 위해서는 기본적으로 정답지가 있어야 하므로 실제 환경에 적용하기 위해서는 많은 시간과 비용이 소요된다. 또한 많은 양의 전체 데이터에서 실제 드리프트가 발생한 구간을 정확히 탐지하는 부분에서도 부족한 모습을 보인다.

따라서 본 논문에서는 네트워크 환경에서 공격 분류모델이 운영된다고 가정하였을 때, 새로 유입되는 전체 시계열 데이터를 대상으로 XAI 지표를 활용하여 Anomaly 기반 드리프트가 발생한 시점을 탐지하는 방안을 제안한다.

본 논문 구성은 다음과 같다. 2장에서는 MLOps 및 모델 드리프트 탐지에 대한 관련연구 및 한계점을 제시하며, 3장에서는 본 논문에서 제안하는 모델의 Framework와 XAI 기반 Anomaly 분석을 통한 드리프트 탐지방안을 제시한다. 4장에서는 DGA 분석모델을 통해 실제 검증을 수행하였고, 5장에서는 결론을 맺는다.

2. 관련 연구

2.1 모델 관리

현재 머신러닝(Machine Learning)은 의료, 금융, 소셜미디어와 같은 일상생활 속 많은 측면에서 활발하게 사용되고 있다[2]. 그러므로 머신러닝 모델은 의사결정 프로세스에 상당한 영향을 미칠 수 있기에 구축과정에서 시스템의 운영목적을 제대로 수행할 수 있도록 설계된다. 하지만 모델 설계자의 기대와 다르게 시간이 지나면서 배포된 모델이 실제 환경에서 지속해서 좋은 성능을 유지하기는 쉽지 않다. 이러한 이유로 관리자가 ML 모델을 배포한 뒤 초기성능과 유사하도록 운영하기 위해서는 모델 분석, 데이터 검증, 테스트 및 디버깅, 모델 모니터링 등의 요소 기반 ML모델의 관리/감독을 통해 안정적인 운영을 해야한다[3].

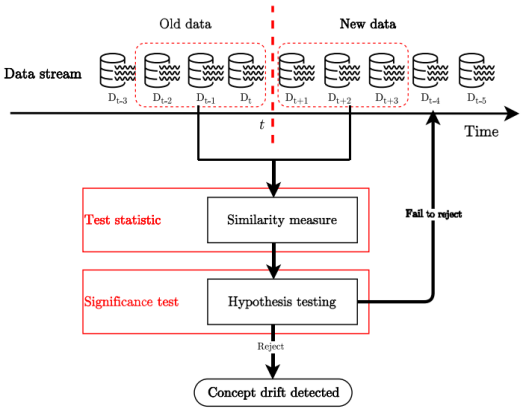
최근 이러한 문제를 인식하여 ML 모델을 관리하고자 DevOps의 기능과 합쳐진 MLOps는 비교적 새롭게 떠오르고 있는 연구 분야로 아직 초기 단계지만 점차 관심이 증가하고 있는 추세이다[4]. MLOps는 기존 학습모델에 새로운 데이터 입력 시 발생하는 문제를 신속하게 확인

하고 재학습을 통해 모델을 지속해서 관리하기 위한 전체적인 프로세스를 의미한다. 이때 ML의 개발과 운영 시스템을 통합하는 5가지 기능을 실현한다[5]. 순서대로 (1) 데이터 수집/전송, (2) 데이터 변환 (3) 지속적인 ML 재훈련 (4) 지속적인 ML 재배포 (5) 최종 사용자에게 생산/프레젠테이션을 출력한다. 즉, MLOps는 특정 시스템 환경에 모델이 배포된 이후 안정적이고 효율적으로 머신러닝 모델이 운영 및 유지되는 것을 목표로 한다. 따라서 이러한 일련의 과정 전체를 자동화 및 통합하는 과정에서 최종적으로는 사용자에게 효율적인 모니터링 파이프라인을 제공하고 모델링을 거쳐 릴리스 후 배포까지 되는 것이 전체적인 MLOps의 목적으로 인식할 수 있다.

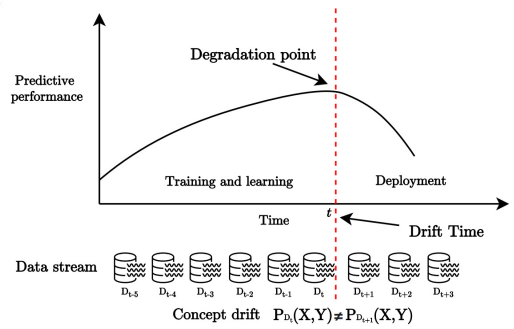
2.2 모델 드리프트 탐지

그림 1의 Concept drift detection framework는 드리프트 탐지의 일반적인 예시로 배포 이전 학습 데이터와 배포 이후 새로운 데이터의 유사성 비교 및 성능 측정을 통해 Drift Signal를 탐지하며, 그림 2의 Performance-based approach mechanism은 일반적인 성능기반 드리프트 측정 방식을 나타낸다[6]. 드리프트는 앞에서 언급한 ML의 개발과 운영 시스템을 통합하는 5가지 기능 중 지속적인 ML 재훈련을 위한 근거로 이러한 드리프트 탐지 방법은 크게 Supervised와 Unsupervised 2가지 방법으로 분류된다. Supervised 환경에서는 표본이 되는 학습 데이터와 새롭게 유입된 테스트데이터에 모두 label이 존재해야 드리프트 탐지가 가능하다.

유사성에는 정확도, 재현율, 민감도 오류율 등의 성능 변화에 따른 탐지방식이 있다. 그중 정확도는 드리프트 발생 시점에서 모델의 성능이 저하됨을 관찰하는 것이다. Abdulbasit A. Darem외 연구에서는 드리프트가 악성코드 분류 정확도에 미치는 영향을 입증하고자 6개의 모델을 대상으로 실험을 진행했다[7]. 실험을 위해 사전에 'DS1' 악성코드 데이터로 모델을 학습하였고 이후 'DS2-DS10'의 최신 악성코드 데이터를 입력으로 성능을 측정한 결과 6개 분류모델에서 특정 시점 이후부터 점차 분류 정확도가 저하되는 것을 확인할 수 있었다. 즉, 지속해서 변화하는 악성코드의 feature value와 label 관계의 변화로 인한 드리프트가 발생했음을 확인할 수 있었다. 하지만 해당 방법은 라벨링의 과정이 필요하므로 실시간 분석 및 신속한 탐지 측면에서 사용하기 쉽지 않다는 단점이 있다.



(그림 1) 개념 드리프트 탐지 프레임워크[6]
(Figure 1) Concept drift detection framwork[6]



(그림 2) 성능기반 접근 메커니즘[6]
(Figure 2) Performance-based approach mechanism[6]

모델 오류율(loss) 변화측정은 모델 모니터링 과정에서 드리프트를 탐지할 수 있는 주된 방법이다. 오류율은 기본적인 모델을 평가하는 지표로 최적의 머신러닝 모델을 생성하기 위해 오류율을 최소화하는 것이 목표이다. 이때 분류모델과 회귀모델에 따라 적절한 오류율 함수(loss function)를 선택할 수 있다. 먼저 회귀모델의 loss function 중 MSE(Mean Squared Error)는 실제값과 예측값의 차이를 제곱하여 평균화한 값으로 생성된 모델이 데이터와 차이가 어느 정도 있는지 확인할 수 있고 분류모델의 예측값은 실수형인 회귀모델과 다르게 0과 1로 분류되기 때문에 주로 entropy 방식을 통해서 loss값이 산출된다. 즉 산출과정에서 데이터의 분포가 반영되기 때문에 이를 활용해 데이터 내의 드리프트 발생 여부를 확인할 수 있다.

하지만 Daniel Vela외 연구에서는 모델 loss값은 선택한 모델과 사용자가 설정한 파라미터 그리고 새롭게 유

입되는 입력데이터 간의 관계와 여러 외부 환경요인들로 인해 드리프트 탐지에서 불확실한 결과를 보일 수 있다는 한계점을 언급하였다[8]. 실험을 위해 무작위로 선택된 배포 시간(t0)에서 새로운 데이터가 입력되는 기간(dt)에 따라 3가지 시나리오를 생성 후 각각의 MSE값을 산출하여 그래프를 통해 비교하였다. 그 결과 모델 드리프트 탐지에 오직 loss만을 활용하는 것은 정확한 발생 시점을 탐지하기에 불확실한 정보임을 보여주었다.

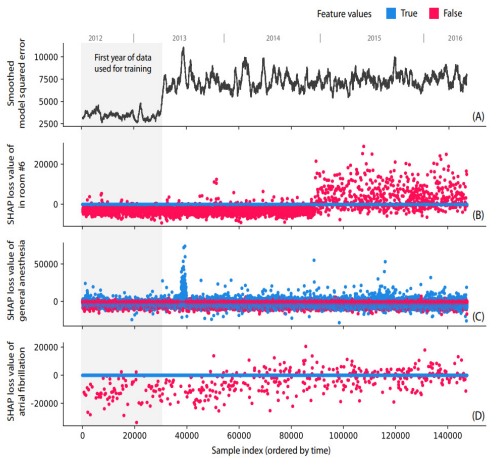
첫 번째는 사용된 환자 및 날씨 데이터의 상당한 변화에도 불구하고 장기간 점진적으로 유사하게 loss값이 산출되는 경우이다. 이러한 상황이 지속해서 이어지게 되면 전체 데이터 흐름에서 드리프트가 발생한 정확한 시점을 탐지 및 판단하기 어려워 최종적으로 분석가가 Drift Signal에 대한 확신을 가질 수 없으므로 모델 고장의 원인이나 성능검사 및 재학습의 원인으로 사용하기에 어렵다.

두번째는 시간에 따른 점진적인 오류율의 증가가 아니라 오랜 시간 우수한 성능을 유지하던 중 갑작스러운 loss값의 변화가 나타났을 경우이다. 단순히 과거와 다르게 갑작스럽게 변화한 구간이 드리프트라고 판단할 수도 있겠지만, 실험결과에서 NN모델은 약 1년 이후부터 성능이 저하되었지만, RV 모델은 1-2년까지는 좋은 성능을 보이고 이후 성능이 저하되는 모습을 보였다. 즉, 사용자가 선택한 모델과 특정 데이터에 대한 모델의 적합성에 따라 loss값의 차이가 존재할 수 있음을 나타낸다.

마지막으로 단순 시간적 증가가 오류율 증가의 원인이 아닐 수 있음을 보여준다. 만약 어떠한 데이터를 대상으로 분류하기 위해 NN 모델과 XGBoost 모델을 선택 및 학습한 이후 새로운 데이터와의 loss값을 측정하는 경우 설계자가 모델을 생성할 때 선택한 설정, 하이퍼 매개변수, 학습 데이터의 크기 등의 추가적인 외부 요인으로 인해 loss값에 변동성이 발생한다는 사실을 알 수 있다. 따라서 loss값은 데이터 관점이 아닌 모델 관점으로써 영향을 크게 받기에 실제 드리프트가 발생한 시점을 탐지하기에는 불확실한 결과를 제공함을 알 수 있다.

2.3 XAI 기반 모델 드리프트 탐지

기술발전에 따라 점점 더 많은 분야에서 머신러닝 모델이 사용되면서 AI 시장의 규모도 점차 확대되고 있으나 모델의 높은 성능으로 인한 복잡성과 해석성 및 투명성과의 trade-off 관계로 인하여 AI 시스템의 판단을 100% 신뢰할 수 없다는 점이 실질적인 활용에서 걸림돌로 작용하고 있다. 높은 예측성능을 보이더라도 모델의 블랙박스(Black Box) 문제는 분석가가 어떠한 근거를 통해서 AI가 판단하였는지 알 수 없게 만든다. 그래서 나온 개념이 XAI(eXplainable Artificial Intelligence)이다. AI 결과에 관한 판단 근거를 제공함으로써 AI기술의 신뢰도를 높이고 이로써 AI의 활용범위를 확장시킬 수 있는 기술로 의료/헬스케어 분야 서비스 외에도 제조/보안분야 등 여러 방면에서 현재 많은 연구가 진행되고 있다[9].



(그림 3) 개념 드리프트 탐지 프레임워크[11]
(Figure 3) Concept drift detection framwork[11]

Lundberg 등은 모델의 설명 가능성을 위해 XAI기술로써 Shapley value 기반 SHAP(SHapley Additive exPlanations)을 제안했다[10]. SHAP value는 각 feature 별 모델 예측에 대한 기여도를 측정하여 모델 판단의 근거를 제공하는 기술이다. Feature의 기여도는 단일 특징을 제거하였을 때 모델 판단이 변화하는 정도를 의미하며 모델 판단에 중요한 특징일수록 높은 값이 산출된다.

Lundberg, S.M.외 연구에서는 SHAP value를 통해 드리프트 탐지와 기여하는 feature를 발견할 수 있음을 보인다[11]. 그림 3은 SHAP value를 활용하여 모델 배포 이후 모니터링의 예시를 나타내는 결과 그래프이다. 특히 (a)와 (d)를 비교하여 드리프트 탐지 활용의 가능성을 주장한다. 실험을 위해 2015년 이후부터 room no.6 label과 room no 13 label을 변경한 테스트데이터를 사용하였다. 이후 기존 loss만을 사용하였을 때와 SHAP value를 사용하였을 때의 드리프트 탐지결과를 비교한다. 그 결과 (a)의 그래프는 드리프트 탐지에서 흔하게 사용하는 모델 예측의 전반적인 손실 그래프로 2012년에는 학습데이터의 loss값으로 낮은 수치를 보이지만, 테스트 데이터가 입력되는 시점에서는 불가피한 손실의 증가 폭을 보인다. 드리프트

탐지 관점에서 예상대로라면 2015년부터 데이터 라벨의 변경으로 loss 값이 증가해야 하지만, 해당 그래프를 통해서는 드리프트가 발생한 정확한 시작과 진행 상황을 추측하기 힘들다. 하지만 (b)는 정확히 드리프트가 발생한 지점에서 SHAP value를 통해 오류 증가에 대한 기여도 변화를 통합하여 검출될 수 있음을 보였다. (c)는 시스템 내부적 측정방식의 문제 발생으로 인한 특정 기능 오류를 감지할 수 있음을 보이며 (d)는 모델의 손실 함수에 대한 시간적 흐름에 따른 SHAP value이다. 점진적인 드리프트가 일어난 경우, 그래프에 나타난 것과 같이 SHAP Loss의 변화를 통해 드리프트 발생을 인지할 수 있음을 주장한다.

결국 드리프트 탐지는 모델이 배포된 이후 업데이트가 필요한 시점을 알리는 신호이므로, 다음 장에서는 이를 탐지하는 방안을 제시한다.

3. 제안 모델

이장에서는 XAI 중 모델 예측 과정에서 feature의 기여도를 나타내는 SHAP value를 활용하여 데이터 변화로 인한 드리프트를 세밀하게 탐지하고 변화가 발생한 시간대 추적을 위한 방법을 제안하고자 한다.

3.1 제안 방법

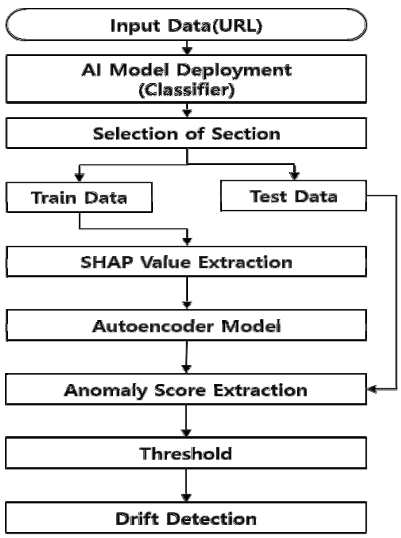
일반적인 드리프트 탐지는 모델에 입력되는 데이터를 공격과 정상으로 구분할 필요 없이 전체를 대상으로 Anomaly 측정을 통해 탐지할 수 있다. 하지만 본 실험에서는 드리프트 유발을 위한 변화하는 정상 데이터를 만들지 못했기에 공격일 때의 시계열과 정상일 때의 시계열을 나누어서 사전에 설정한 임계값에 따라 드리프트를 탐지하고자 한다. 본 논문에서 제안하는 Framework는 그림 4와 같으며 자세한 내용은 아래와 같다.

3.1.1 DGA 분류 모델 생성

먼저 DGA 공격과 정상 데이터를 대상으로 URL 문자열의 통계적인 특징을 반영한 Lexical 기반 Featuring을 적용한 후 분류를 위해 XGBoost 모델을 생성한다. XGBoost는 Gradient boosting 알고리즘 기반으로 여러 개의 분류기를 생성하는 앙상블 학습을 통해 높은 예측값 산출이 가능한 모델이다. 좀 더 세밀한 학습을 위해 세부 파라미터를 조정하였으며 DGA인 경우 0, 정상이면 1로 분류되는 이진 분류 탐지모델을 대상으로 언제 모델이 갱신되어야 하는지 판단하고자 한다.

3.1.2 구간 설정

먼저 데이터의 구간을 선정하는 과정은 드리프트를 추적하기 위한 가장 첫 번째 단계로써 그림 1에서 볼 수 있듯이 드리프트를 추정하기 위해서는 Old Data와 New Data가 존재한다. 이때 Old Data는 배포 이전의 학습 데이터이며 드리프트 탐지를 위한 참조 데이터를 의미한다. New Data는 모델이 배포된 이후 드리프트가 발생하게 될 탐지 대상의 테스트데이터로, 해당 데이터 내에서 드리프트를 탐지하고자 한다.



(그림 4) 제안 프레임워크
(Figure 4) Proposed framework

3.1.3 SHAP 추출

데이터의 통계적인 특성은 생성된 시기뿐만 아니라 전 기간에 걸쳐서 빠르게 변화한다. 그러므로 데이터 내의값들은 여러 가지 단위와 범위로 구성되기 때문에 데이터가 변화한 정도를 측정하는 과정에서 어려움이 존재한다. 따라서 이를 보완하기 위해 일반적으로는 일정값 범위로 스케일링하는 과정을 거치는 등의 모델학습에 적합한 값으로 변화를 주어야 한다. 이를 위해 앞선 Lundberg. S.M. 연구에서 모델 loss값에 기여한 SHAP value를 통해서 드리프트를 탐지할 수 있다라는 주장을 참고하여 통계적

변화량을 측정하는 과정에서 SHAP value를 활용하고자 한다. 따라서 전체 데이터를 대상으로 SHAP 값을 수식 1과 같이 추출하고 이후 추가적으로 입력데이터의 feature 중 모델의 결과와 성능에 중요한 Feature를 선별하여 향상된 드리프트 탐지를 하고자 한다.

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|(|F|-|S|-1)!}{|F|!} [f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)] \quad (1)$$

3.1.4 드리프트 탐지 모델 생성

이 과정에서는 드리프트를 실질적으로 탐지하는 모델을 생성한다. 배포 이전의 학습 데이터 전체를 대상으로 SHAP을 적용한 후 전체 SHAP value에서를 활용하여 Anomaly 모델인 Autoencoder을 생성하고자 한다. 이때 Autoencoder 는 입력과 출력의 편차를 기반으로 복원 오차 값을 통해 정상과 다른 패턴을 보이는 이상 패턴을 탐지할 수 있어서 주로 이상 탐지 목적의 연구에서 자주 사용되는 모델이다. 이에 따라 본 실험에서는 사용하는 데이터 세트에 적합한 탐지를 위해 공격과 정상의 시계열을 나누어서 진행하였으며, 각각의 SHAP value를 학습한 두 개의 Autoencder 모델을 생성하였다. 추가로 세밀한 학습을 위해 activation은 'relu', epoch는 100 batch_size는 256의 세부 파라미터를 사용하였다. 이때 Activation을 위해 'relu'를 사용한 이유는 다른 활성화 함수 대비 계산이 효율적이기에 학습 속도가 빠르고, sigmoid와 같은 활성화 함수의 경우 기울기 소멸 문제(Vanishing gradient problem)을 초래할 수 있지만, 해당 함수는 양수 값에서 1이라는 점에서 이러한 문제를 줄일 수 있기에 더 효율적인 모델을 만들 수 있다는 점에서 활용하였다.

3.2 모델 드리프트 탐지

배포된 모델의 드리프트를 탐지하고자 생성한 Autoencoder 탐지모델에 테스트데이터를 입력으로 Anomaly score을 추출하여 사전에 설정한 Threshold 이상이면 드리프트가 발생한 시점으로 판단하고자 한다.

제안 방법의 유용성을 검증하고자 배포된 분류모델의 loss값과 비교하고자 한다. 분류모델의 loss값을 산출은 XGBoost의 내장된 이진 분류의 오류함수인 error metric을 사용하였다. 이후 분류모델에 학습 데이터와 드리프트가 포함된 테스트데이터를 입력으로 loss값을 산출하고 사전에 설정한 Threshold 이상인 경우 드리프트라고 판단하고자 한다. 이를 통해 최종적으로 loss값을 통한 드리프트 탐지와 Anomaly score를 통한 드리프트 탐지 방법을 비교하였다. 마지막으로 드리프트가 포함되지 않은 경우와 포함되었을 때 발생하는 모델의 성능저하를 확인하고자 모델의 정확도를 측정하여 비교 및 검증하고자 한다.

4. 실험 결과

4.1 데이터 셋

모델 배포 이전과 이후의 드리프트 구간을 찾기 위해 사전에 데이터의 구간 선정이 필요하다. 따라서 본 논문에서는 NetLab에서 제공하는 DGA 데이터 세트를 활용하고 실험을 위해 새로운 DGA 기술이 도입되는 시점을 추가한 데이터를 사용한다. NetLab은 Oihoo 360의 보안팀으로 2014년에 창립되어 보안 데이터 중 특히 봇넷, 허니팟, 대규모 DNS 데이터 및 보안 데이터 관련 연구에 주력하는 팀으로 본 연구에서는 해당 팀에서 제공하는 DGA 데이터를 사용하고자 한다. 이때 드리프트의 정상 데이터는 배포 이전의 정상 데이터와 다른 성격의 변화한 데이터를 생성하지 못하였기 때문에 전체 시계열에서 공격과 정상 2개의 시계열로 나누어서 진행하고자 한다. 따라서 공격 데이터의 NO.1과 NO.2는 DGA 중 'pykspa_v1'에 해당하는 같은 유형이며 NO.3은 성격이 다른 DGA 중 'flubot'으로 구성된다. 배포 이전과 배포 이후 드리프트 데이터의 DGA 공격유형의 차이는 표 3과 같다. 정상과 공격의 전체 데이터 구성은 표 1과 표 2와 같다. 공격 시계열 데이터에서 배포 이전 학습 데이터는 5,031개, 배포 이후 테스트데이터 구간은 2,012개, 드리프트 유발 데이터 구간은 500개 데이터를 사용한다. 정상 시계열 데이터는 배포 이전 학습 데이터는 4,969, 배포 이후 테스트데이터 구간은 1,988, 드리프트 유발 데이터 구간은 500개를 사용하여 각각 시계열 데이터 대상 드리프트 탐지 결과를 산출한다.

추가로 본 실험의 목적은 드리프트 시간대를 찾는것이 목적이므로 임의로 데이터에 시계열 인덱스를 추가한다. 먼저 공격의 배포 이전 모델은 2002-01-01부터 2015-10-10까지의 데이터로 학습된다고 가정한다. 배포 이후 테스트데이터는 2015-10-11부터 입력되며 2021-04-13부터 드리프트가 발생한 시점이다. 다음 정상의 배포 이전 모델은 2002-01-01부터 2015-08-26까지의 데이터로 학습된다. 배포 이후 테스트데이터는 2015/08/27부터 입력되며 2021-02-04부터 드리프트가 발생한 시점이다. 최종적으로 본

실험은 드리프트가 발생한 시간대를 탐지하는 것을 목표로 한다.

(표 1) 공격 데이터 세트
(Table 1) Attack dataset

No.	Data Type	Number of Data
1	Train	5,031
2	Test	2,012
3	Drift	500

(표 2) 정상 데이터 세트
(Table 2) Normal dataset

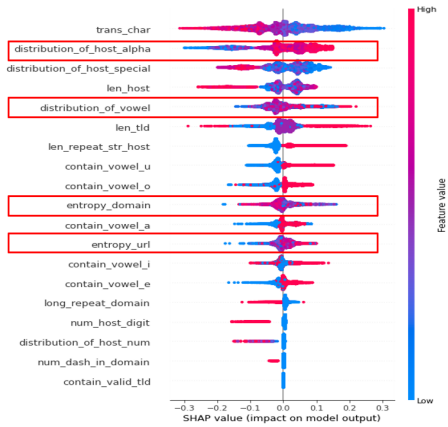
No.	Data Type	Number of Data
1	Train	4,969
2	Test	1,988
3	Drift	500

(표 3) 공격유형 비교
(Table 3) Attack type comparison

	pykspa_v1	flubot
TLD	biz, com, net	ru, com, cn, su
Length	6-15, a~z	15, a~y
Period	2 day	1 month
Count	5,000	5,000
Example	agadss.biz	bsgeijagbavgavk,cn

4.2 SHAP 추출

드리프트 구간 탐지를 위해 기본으로 전체 학습 데이터 대상 SHAP value를 추출한다. XAI 기술 중 하나인 SHAP의 예측 결과는 설명자(explainer)에 의해 설명된다. SHAP의 Explainer는 모델에서 가장 주요한 몇 가지 feature을 강조해주는 역할을 하며 파라미터로 학습 데이터와 생성한 모델을 사용한다. 본 실험에서는 Lundberg가 제안한 방법으로 Decision Tree, Random Forest, Gradient Boosted Tree와 같은 트리 기반 모델을 위한 Tree Explainer SHAP을 사용한다. 해당 방법은 빠르고 정확한 Shapley value를 계산한다. 추출된 SHAP value를 기반으로 특정 시간대의 드리프트가 발생한 구간에 대해서 드리프트 탐지모델을 통해 Anomaly score를 추출하여 그래프로 시각화하여 분석하고자 한다. 이를 활용하여 모델 예측에 주요하게 기여한 feature를 선정하고자 그림 5와 같이 Global 관점의 Plot인 Summary Plot을 분석하여 붉은색 구역로 표시한 특징값이 섞인 특징을 제외하고 총 19개 중 10개의 feature을 선정한 특징목록은 표 4와 같다.



(그림 5) SHAP 요약 플롯
(Figure 5) SHAP summary plot

(표 4) 중요 특징 리스트
(Table 4) Importance feature list

No.	Feature Name
1	trans_char
2	distribution_of_host_special
3	len_host
4	len_tld
5	len_repeat_str_host
6	contain_vowel_u
7	contain_vowel_o
8	contain_vowel_a
9	contain_vowel_i
10	contain_vowel_e

4.3 실험 결과

네트워크 환경에서 지속 유입되는 URL 데이터를 대상으로 DGA 여부를 탐지하는 배포된 모델을 대상으로 3가지 실험을 통해서 드리프트를 탐지하고 비교하였다. 첫 번째, 모델 정확도는 측정을 위해 데이터의 라벨링하는 과정을 거쳐 성능이 저하되었을 때 드리프트라고 판단할 수 있다. 두 번째, 모델 loss는 이전 loss 값과 드리프트라 의심되는 증가한 loss 값과의 비교계산을 통해서 판단할 수 있다. 마지막으로 제안 방법은 드리프트 탐지모델로 산출된 Anomaly score가 Threshold이상인 경우 기존방법 대비 드리프트가 발생한 시점의 시간대를 탐지할 수 있었다.

4.3.1 모델 오류율 기반 드리프트 탐지

제안 방법의 실험에 앞서 일반적인 드리프트 탐지 방

법 중 모델의 loss 값의 변화를 통해 드리프트를 탐지하고자 한다. 따라서 배포 이전 학습 데이터로 학습된 모델에 대해서 전체 시계열 데이터 중 공격과 정상의 시계열로 나눈 테스트데이터를 입력으로 드리프트가 포함된 테스트와 포함되지 않은 테스트에서 변화하는 모델의 loss 값을 비교하였다. 이때 사용한 loss function은 이진 분류데이터에서 주로 사용하는 error metric으로 전체 데이터 중 잘못 분류한 비율을 의미한다.

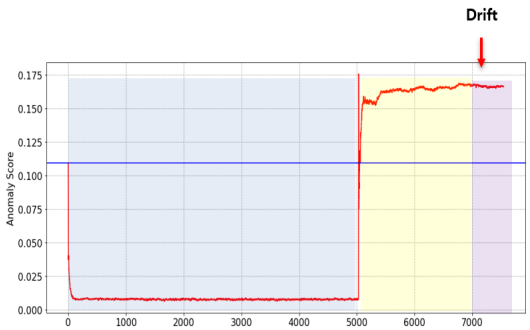
우선, 공격에 대해 loss 값을 산출한 결과는 그림 6과 같다. loss 산출을 위해서 XGBoost 분류모델에 입력으로 전체 학습 데이터 중 공격만 추려진 데이터와 드리프트가 포함된 공격 테스트데이터를 입력으로 추출을 진행하였다. 그 결과 테스트데이터가 입력되는 시점부터 Threshold 이상으로 드리프트 데이터로 의심할 수 있다. 하지만 앞서 설정한 데이터 구간에 따르면 6,900 이후 약 7000 index 지점 이상부터 실제 드리프트가 발생한 지점이지만, loss값의 결과만으로는 해당 지점에서 드리프트가 발생한 것을 확인할 수 없었다. 즉 모델에 입력된 데이터의 변화로 loss값이 증가하였지만 실제 데이터가 변화한 구간에서 증가한 것이 아니라 단순 입력된 테스트 전체 구간에서 loss값이 증가하여진 것으로 보이기에 실제 드리프트가 발생한 지점을 탐지하기에 어려움이 존재한다는 사실을 알 수 있었다.

정상인 경우에도 같은 과정을 통해서 진행하였으나 데이터를 구성하는 과정에서 드리프트를 위한 변화하는 데이터를 생성하지 못하였다. 따라서 그림 7과 같이 모델의 loss값이 Threshold 이하로 산출됨에 따라 정상 데이터는 드리프트가 아닌 것으로 판단할 수 있다. 이러한 결과를 통해 드리프트 발생 시 실시간으로 정확한 탐지가 어렵다는 사실을 알 수 있었다.

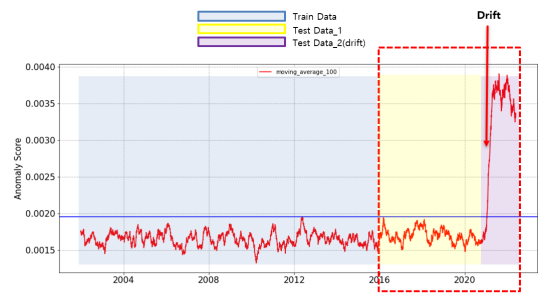
(그림 7) 정상 시계열에 대한 모델 손실
(Figure 7) Model loss for normal time series

4.3.2 XAI를 활용한 Anomaly 기반 드리프트 탐지

공격 구간 드리프트 탐지모델에 입력되는 학습 데이터는 테스트데이터 중 공격으로만 추려진 SHAP value 값을 사용하며, 이때 일부는 학습 데이터와 유사한 데이터이고 나머지는 드리프트가 포함된 데이터를 사용하여 드리프트 탐지를 진행한다. 그래서 전체 7,543개의 공격 데이터를 활용하여 Autoencoder 학습한 뒤 Anomaly Score를 추출하여 배포 이전 공격 학습 데이터의 Anomaly score와 드리프트가 포함된 배포 이후 공격 테스트데이터의 Anomaly score 흐름을 비교한다. 이때 Anomaly score는 1초씩 WindowSize 100으로 슬라이딩 윈도우를 진행한다. 탐지에 앞서 사전에 설정한 공격 데이터 구성에서 드리프트가 탐지되는 시점은 7,043 index로 2021-04-13에 발생하게 된다. 탐지 결과 전체 공격 시계열에서 Threshold 이상인 드리프트 구간은 100개 단위의 슬라이딩 윈도우로 인해 앞의 100개 size를 제외하여 6,944 index에 탐지되고 2020-10-10 이후부터 드리프트가 발생한 시점으로 판단할 수 있다. 그 결과는 그림 8과 같으며 드리프트가 탐지



(그림 6) 공격 시계열에 대한 모델 손실
(Figure 6) Model loss for attack time series



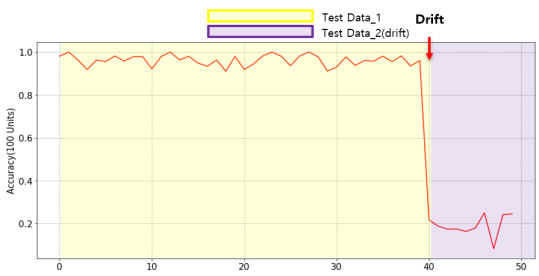
(그림 8) 이상탐지기반 드리프트 탐지
(Figure 8) Anomaly-based drift detection

된 테스트데이터를 확대한 결과는 그림 9와 같다. X축은 공격 데이터의 전체 시간대이며 Y축은 Anomaly score를 의미한다. 이는 단순히 입력된 데이터의 전체가 아니라 특정 구간에서 드리프트가 발생했음을 알 수 있으며, 이는 실시간 탐지에 있어서 loss값 기반 탐지와 차별점이 있음을 알 수 있다.

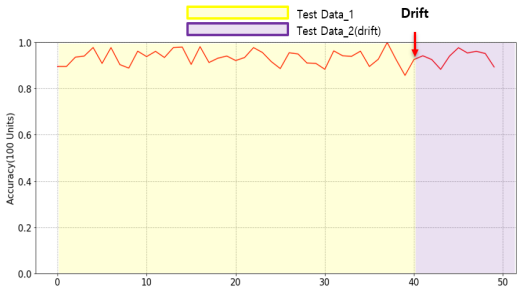
(그림 9) 테스트 데이터의 드리프트 탐지
(Figure 9) Drfit detectin in test data

4.3.3 모델 정확도 기반 드리프트 탐지

마지막으로 드리프트가 포함된 테스트데이터 대상 정상과 공격 각각에 대해서 변화하는 모델의 성능 비교를 통해 검증하고자 한다. 우선 드리프트가 포함되지 않은 테스트데이터의 경우 0.95의 높은 성능을 보였다. 그에 반해 드리프트가 포함된 테스트데이터의 경우 0.87로 드리프트가 포함되지 않았을 때 비해 전체적으로 성능이 0.08 이하로 저하됨을 알 수 있었다. 다음은 각 공격과 정상 시계열 대상 100개 단위씩 탐지율을 측정한 결과, 그림 10, 그림 11과 같다. 정상이면 배포 이전 학습 데이터와 다르지 않기에 높은 정확도가 유지됨을 보이지만, 공격이면 특정 시점에서 성능이 확연하게 저하되는 것을 보인다. 이를 통해 모델에 기존 학습 데이터와 다른 유형



(그림 10) 공격의 모델 정확도
(Figure 10) Model accuracy in attack



(그림 11) 정상의 모델 정확도
(Figure 11) Model accuracy in normal

데이터의 입력 시 모델의 품질이 저하된다는 사실을 검증할 수 있었다. 그러나 실험환경에서는 정답지가 존재함에 따라 성능을 비교하여 드리프트가 발생했음을 알 수 있지만, 실제 환경에서는 라벨이 없는 Unsuperevised 환경이기에 성능 비교를 통한 드리프트 탐지에는 어려움이 존재한다.

5. 결론 및 향후 연구과제

오늘날 머신러닝 모델의 역할이 확장됨에 따라 모델 관리의 중요성이 점차 중요하게 여겨지고 있다. 이때 모델은 시간이 지나면서 변화하는 데이터로 인해 성능이 저하되는 드리프트가 발생하게 되고, 이로 인해 기존 모델의 무력화와 전체 시스템의 문제를 발생시키게 된다. 이런 문제를 예방하기 위해 일반적으로 모델의 정확도와 loss값을 추적하여 드리프트를 탐지하고자 한다. 하지만 정확도는 모델 예측과 결과에 대한 정답이 존재하는 경우에만 사용할 수 있어, 정답지가 없는 환경에서는 적용하기 어렵고 사후적인 방식이기에 실시간 탐지를 할 수 없다는 단점이 있다. 따라서 주로 모델의 loss값을 통해 드리프트 탐지가 이뤄지게 되는데 Diniel Vela외 연구에서와같이 모델의 loss값을 통한 탐지는 데이터 변화에 따른 세밀한 탐지를 하지 못한다는 단점이 존재하였다. 드리프트가 포함된 테스트데이터에 대해서 전체적으로 Threshold 이상의 높은 loss값을 보였지만 실질적으로 드리프트가 발생한 위치를 알 수 없었기에 드리프트만을 원인으로 loss값이 증가한 것이 맞는지에 대한 불확실성으로 인해 사용에 어려움이 있었다.

본 논문은 이를 보완한 안정적인 모델 관리를 위해 SHAP value을 활용한 Anomaly 기반 드리프트 시점 탐지 방안을 제안하였다. 배포 이후의 데이터에 대해서 SHAP

value 기반 Anomaly score를 산출하였고 사전에 설정한 데이터의 드리프트 구간에서 정확히 Threshold 이상의 값이 산출됨에 따라 제안 방법의 유효성을 검증할 수 있었다. 이때 탐지하는 과정에서 슬라이딩 윈도우 사이즈를 줄인다면 신속한 탐지는 가능하겠지만, 너무 작은 값은 드리프트 판단에 있어 noise를 줄 수 있으므로 운영하는 상황에 따라 적절한 운용이 필요할 것으로 보인다.

또한 향후 모델 관리의 필요성과 중요성이 증가함에 따라 본 연구에서 제안한 방법론의 범용성을 검증하기 위해 다양한 유형의 데이터 셋 대상 드리프트 탐지의 효율성 및 SHAP value외의 다른 XAI 기법들과의 통합을 통해 더욱 정확한 드리프트 탐지방안의 연구가 필요할 것으로 보인다. 또한 드리프트가 탐지되었을 때 해당 드리프트의 원인을 분석하고 이를 해결하려는 방안을 추가로 진행한다면 데이터 패턴의 변화나 외부 환경적 요인들이 드리프트에 어떠한 영향을 주는지 분석함으로써 앞으로의 연구에 크게 기여할 수 있을 것이라 기대된다.

참고문헌(Reference)

[1] Sarker, I. H., Kayes, A. S. M., Badsha, S., Alqahtani, H., Watters, P., & Ng, A., “Cybersecurity data science: an overview from machine learning perspective,” Journal of Big data, 7, 1-29, 2020. https://doi.org/10.1186/s40537-020-00318-5

[2] Bhatt, U., Xiang, A., Sharma, S., Weller, A., Taly, A., Jia, Y., ... & Eckersley, P., “Explainable machine learning in deployment,” in Proceedings of the 2020 conference on fairness, accountability, and transparency, pp. 648-657, January 2020. https://doi.org/10.1145/3351095.3375624

[3] Spjuth, O., Frid, J., & Hellander, A., “The machine learning life cycle and the cloud: implications for drug discovery,” Expert opinion on drug discovery, 16(9), 1071-1079, 2021. https://doi.org/10.1080/17460441.2021.1932812

[4] John, M. M., Olsson, H. H., & Bosch, J., “Towards mlops: A framework and maturity model,” in 2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE, pp. 1-8, September 2021. https://doi.org/10.1109/SEAA53835.2021.00050

[5] Tamburri, D. A., “Sustainable mlops: Trends and challenges,” in 2020 22nd international symposium on symbolic and numeric algorithms for scientific computing (SYNASC), IEEE., pp. 17-23, September 2020. https://doi.org/10.1109/SYNASC51798.2020.00015

[6] Bayram, F., Ahmed, B. S., & Kassler, A., “From concept drift to model degradation: An overview on performance-aware drift detectors,” Knowledge-Based Systems, 108632, 2022. https://doi.org/10.1016/j.knosys.2022.108632

[7] Darem, A. A., Ghaleb, F. A., Al-Hashmi, A. A., Abawajy, J. H., Alanazi, S. M., & Al-Rezami, A. Y., “An adaptive behavioral-based incremental batch learning malware variants detection model using concept drift detection and sequential deep learning,“ IEEE Access, 9, 97180-97196, 2021. https://doi.org/10.1109/ACCESS.2021.3093366

[8] Vela, D., Sharp, A., Zhang, R., Nguyen, T., Hoang, A., & Pianykh, O. S., “Temporal quality degradation in AI models,” Scientific Reports, 12(1), 11654, 2022. https://doi.org/10.1038/s41598-022-15245-z

[9] Xu, F., Uszkoreit, H., Du, Y., Fan, W., Zhao, D., & Zhu, J., “Explainable AI: A brief survey on history, research areas, approaches and challenges,” in Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8, pp. 563-574, Springer International Publishing, 2019. https://doi.org/10.1007/978-3-030-32236-6_51

[10] Lundberg, S. M., & Lee, S. I., “A unified approach to interpreting model predictions,” Advances in neural information processing systems, 30, 2017. https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

[11] Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., ... & Lee, S. I., “From local explanations to global understanding with explainable AI for trees,” Nature machine intelligence, 2(1), 56-67, 2020. https://doi.org/10.1038/s42256-019-0138-9

◐ 저 자 소 개 ◑

이 예 은(Ye-eun Lee)
2019년 3월~현재 호서대학교 컴퓨터공학부 학석사과정
관심분야 : 악성코드 분석, 침입 탐지, 이상징후 탐지, 정보보호, AI
관심분야 : 데이터베이스, etc.
E-mail : judiaye4477@gmail.com

이 태 진(Tae-jin Lee)
2003년 2월 포항공과대학교 컴퓨터공학과
2008년 2월 연세대학교 컴퓨터공학과 석사
2017년 2월 아주대학교 컴퓨터공학과 박사
2013년 1월~2017년 2월 한국 인터넷진흥원 팀장
2017년 3월~현재 호서대학교 컴퓨터공학부 교수
관심분야 : 시스템 보안, 침해사고 대응, Trustworhty AI
E-mail : kinjecs0@gmail.com