SCA: 안드로이드 악성 앱 탐지의 개념 드리프트 대응을 위한 보안 기반 적응 모듈

권나희†, 노경민†, 조원빈†, 박수현‡, 조성제¶
단국대학교 사이버보안학과†, 단국대학교 컴퓨터학과‡, 단국대학교 소프트웨어학과¶
{alsrl030902, imsie1, wonbin0517, parksh, sjcho}@dankook.ac.kr

SCA: A Security-Calibrated Adapter for Concept Drift in Android Malware Detection

Nahee Kwon†, Kyoungmin Roh†, Wonbin Cho†, Suhyeon Park‡, Seong-je Cho¶
Department of Cybersecurity, Department of Computer Science and Engineering‡, Department of Software Science¶, Dankook University
{alsrl030902, imsie1, wonbin0517, parksh, sjcho}@dankook.ac.kr

요 약

본 논문은 개념 드리프트 환경에서 안드로이드 악성 앱 탐지 모델의 출력 신뢰도 저하를 완화하기 위해 SCA(Security-Calibrated Adapter)를 제안한다. SCA는 기존 Backbone 분류기를 고정한 상태에서 Security Descriptor, Latent Feature, 기본 Logit을 결합하여 샘플별 출력 Logit을 보정한다. 이를 통해 스케일, 바이어스, 잔차 보정항을 생성하고 분포 변화에 따른 판별 점수 왜곡을 완화한다. 실험 결과, SCA는 2019-2023년 평가 구간에서 평균 F1-Score 91.8%와 Accuracy 91.5%를 달성하였다. 이는 Additive Adapter 대비 각각 17.4%와 26.9%, Pure MLP 대비 각각 20.2%와 32.8% 높은 성능이다. 따라서 SCA는 전체 재학습 없이도 제한된 적응 샘플만으로 개념 드리프트 환경에서 안정적인 탐지 성능을 유지할 수 있다.

1. 서 론[1]

안드로이드 악성 앱 탐지는 모바일 환경의 보안을 위해 중요한 기술이다. 그러나 실제 운영 환경에서는 시간의 흐름에 따라 악성 앱과 정상 앱의 행위 분포가 변화하는 개념 드리프트 문제가 발생한다. 이로 인해 초기 데이터로 학습된 탐지 모델은 시간이 지날수록 성능이 저하될 수 있으며, 새로운 악성 행위 패턴이 등장할 경우 기존 분류기의 출력 신뢰도 역시 흔들릴 수 있다.

기존 연구에서는 이러한 문제를 완화하기 위해 새로운 샘플을 수집하여 재학습하거나, 탐지 Threshold를 조정하는 방식이 주로 사용되었다. 그러나 재학습 기반 방법은 반복적인 라벨 확보와 모델 업데이트가 필요하므로 운영 비용이 크고, Threshold 기반 방법은 전역적인 보정에 머무르는 경우가 많아 개별 입력의 특성을 충분히 반영하지 못한다. 실제 온라인 운영 환경에서는 성능뿐 아니라 적응에 필요한 샘플 수, 라벨링 비용, 적응 부담까지 함께 고려할 필요가 있다.

본 연구에서는 이러한 한계를 완화하기 위해 Backbone 분류기를 고정한 상태에서 출력만을 보정하는 SCA(Security-Calibrated Adapter)를 제안한다. SCA는 Security Descriptor, Backbone의 Latent Feature, 그리고 base logit을 함께 이용하여 샘플별 보정항을 생성하고, 이를 통해 최종 판별 점수를 조정한다. 또한 적응 비율을 조절할 수 있도록 설계하여, 온라인 환경에서 사용 가능한 비용 수준에 따라 적은 수의 샘플만으로도 경량 적응이 가능하도록 하였다. 이는 모든 샘플을 동일하게 반영하는 방식과 달리, 운영 상황에 따라 성능과 비용 사이의 균형을 선택할 수 있다는 점에서 실용적 의미가 있다.

본 연구의 기여는 다음과 같다.

- 안드로이드 악성 앱 탐지의 개념 드리프트 환경에서 Backbone 전체 재학습이 아닌 샘플별 출력 보정 기반 적응 구조를 제안하였다.
- 기존 Backbone을 고정한 채 부가적으로 결합할 수 있는 Adapter형 SCA 모듈을 설계하여, 특정 분류기 전체를 재구성하지 않고도 경량 적응이 가능하도록 하였다.
- 적응 비율을 조절할 수 있도록 하여 온라인 운영 환경에서 성능뿐 아니라 라벨링 및 적응 비용까지 고려한 유연한 적용 가능성을 보였다.

2. 관련 연구

안드로이드 악성 앱 탐지에서 개념 드리프트는 시간의 흐름에 따라 악성 및 정상 앱의 행위 분포가 변화하면서 기존 분류기의 성능을 저하시키는 핵심 문제로 알려져 있다. 이를 해결하기 위해 지속학습 기반 방법은 Active Learning과 Contrastive Learning을 결합하여 재학습 샘플을 선택하고 탐지 성능 저하를 줄이고자 하였다[1]. 그러나 이러한 방식은 새로운 샘플의 라벨 확보와 반복적인 재학습이 필요하므로 운영 비용과 배포 부담이 크다. 재학습 없는 접근에서는 API 공출현 그래프의 구조 변화를 이용해 드리프트를

* 본 연구는 2026년 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업 지원을 받아 수행되었음(2024-0-00035) 또한 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원-학석사연계 ICT핵심인재양성 지원을 받아 수행된 연구임(IITP-2026-RS-2023-00259867).
¶ 교신 저자임.

정량화하고, 이를 기반으로 탐지 임계값을 조정하는 방법이 제안되었다[2]. 이 방법은 재학습 없이도 드리프트에 대응할 수 있으나, 주로 전역적 구조 변화와 규칙 기반 임계값 보정에 초점을 두어 입력별 특성을 반영한 세밀한 출력 적응에는 한계가 있다. 한편, 최근에는 안드로이드 악성 앱 탐지에서 분류 Score를 확률적으로 보정하여 Calibrated Risk Score를 구성하거나[3], 예측 결과에 대한 Confidence Guarantee를 제공하여 불확실한 샘플을 보수적으로 처리하는 연구도 보고되었다[4]. 또한 보안 분야에서의 기계학습에서는 Threshold 및 Score Calibration이 실제 운영 성능에 큰 영향을 미친다는 점도 지적되었다[5]. 그러나 이러한 접근은 주로 후처리 확률 보정, 신뢰도 추정, 또는 규칙 기반 임계값 제어에 머무르므로 입력별 특성과 내부 표현 변화를 함께 반영하여 기존 분류기의 출력을 적응시키는 방식과는 차이가 있다.

3. 제안 모듈

본 연구에서는 기존 Backbone 분류기 전체를 재학습하는 대신, Backbone 을 고정한 상태에서 Adaptation Module 만 추가 학습하는 경량 적응 구조를 사용한다. 이를 위해 새로운 분포에서 Backbone 의 출력을 직접 대체하는 대신, Backbone 이 생성한 기본 Logit 을 샘플별로 보정하는 SCA(Security-Calibrated Adapter)를 설계하였다. SCA 의 전체 구조는 그림 1 과 같다.

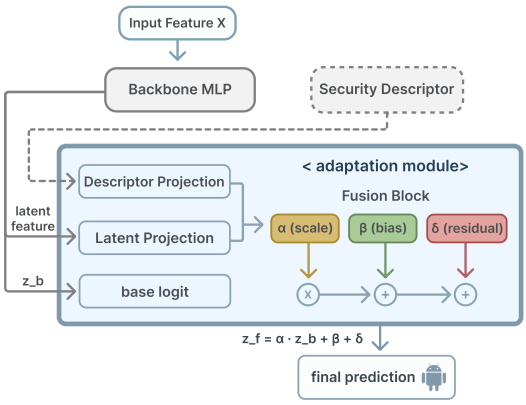

그림 1. 제안 모듈의 구조도

제안 모듈의 핵심 가정은 동일한 입력이라도 보안 의미론적 특성과 Backbone 의 내부 잠재 표현에 따라 기존 판별 점수의 신뢰도와 보정 방식이 달라질 수 있다는 점이다. 이에 따라 SCA 는 Backbone 이 생성한 기본 Logit 만을 사용하는 대신, Security Descriptor, Latent Feature, 그리고 기본 Logit 을 함께 입력으로 사용하여 샘플별 보정항을 생성한다.

여기서 Security Descriptor 는 원본 입력 특성들 중 Android 악성 행위와 관련된 Feature 들을 사전에 정의된 보안 위험 그룹으로 매핑한 뒤, 각 그룹의 존재 여부, 로그 빈도 기반 강도, 그룹별 위험도 가중치, 그리고 일부 그룹 간 상호작용 정보를 결합해 구성한 도메인 지식 기반 표현이다. 보안 위험 그룹은 Android 악성 앱에서 반복적으로 관찰되는 행위 의미를 기준으로 정의하였으며, 메시지 및 통화 기능 오용, 지속 실행, 개인정보 접근, 동적 코드 로딩, 네트워크 유출 등 주요 악성 행위 단위를 중심으로 구성하였다. 이러한 그룹화는 개별 API 수준의 희소성을 줄이면서도 단순 빈도 Feature 에서 손실될 수 있는 보안 의미를 보존하기 위한 것으로, SCA 에서는 독립 분류 Feature 가 아니라 Backbone Logit 의 과신 또는 과소평가를 보정하는 보조 의미 표현으로 사용된다.

한편 Latent Feature 은 Backbone MLP 의 마지막 은닉층에서 추출되는 내부 잠재 표현으로, 원본 입력으로부터 Backbone 이 학습한 분류 관련 정보를 압축적으로 담고 있다. 제안 모듈은 이 두 정보와 Backbone 의 기본 Logit 을 함께 이용하여 보정값을 산출한다.

구조적으로 SCA 는 Security Descriptor 와 Latent Feature 를 각각 별도의 Projection Layer 에 통과시킨 후, 이를 Backbone 의 기본 Logit 과 결합하여 Fusion Block 에 입력한다. 이후 Fusion Block 의 출력을 바탕으로 Scaling Term α, Bias Term β, Residual Term δ를 생성하며, 최종 로짓은 다음과 같이 계산된다.

$$z_f = \alpha \cdot z_b + \beta + \delta \quad (1)$$

여기서 z_b는 Backbone 이 산출한 기본 Logit 이고, z_f는 보정된 최종 Logit 이다. α는 기본 Logit 의 영향을 재조정하는 스케일 항으로 작동하며, β는 전체 Score 분포의 이동을 반영하는 보정항이고, δ는 α와 β만으로 충분히 설명되지 않는 추가적인 샘플별 편차를 보완하는 잔차 항이다. 따라서 SCA 는 단일 보정값이 아니라 스케일 조정, Score 이동, 잔차 보정을 함께 수행함으로써 분포 변화에 보다 유연하게 대응한다. 또한 SCA 학습 시에는 Adaptation 샘플에 대한 분류 손실과 함께, Anchor 샘플에 대해 Backbone 의 기존 출력 확률을 유지하도록 하는 Distillation Loss 및 보정항의 과도한 변화를 억제하기 위한 Regularization 을 함께 사용한다. 이를 통해 새로운 분포에 적응하면서도 Backbone 의 기존 판별 특성을 지나치게 훼손하지 않도록 설계하였다.

한편, 적응 비율은 새로운 분포의 샘플 중 적응 과정에 사용할 전체 예산을 결정하는 파라미터로 사용된다. 선택된 샘플은 Adapter 학습과 Threshold 조정에 활용되며, 이때 샘플 구성은 단순 무작위 추출이 아니라 Backbone의 예측 특성과 잠재 표현 정보를 일부 반영하여 이루어진다. 이를 통해 제한된 샘플 수에서도 보다 효율적인 적응이 가능하도록 하였으며, 실제 운영 환경에서는 적응 비율을 조절하여 성능과 라벨링 비용 사이의 균형을 선택할 수 있다.

4. 실험 평가

실험에서는 제안 모듈과 두 가지 비교 대상을 평가하였다. 비교 대상은 각각 (1) SCA, (2) Additive Adapter, (3) Pure MLP 로 구성하였다. SCA 는 본 연구에서 제안한 모듈이며, Additive Adapter 는 SCA 에서 Scaling Term α 를 제외한 Logit 을 사용하는 모듈이고, Pure MLP 는 별도의 적응 과정을 수행하지 않는 기본 Baseline MLP 모델이다.

그림 2 와 3 은 적응 비율에 따른 각 모델의 F1-Score 와 Accuracy 변화를 연도별로 나타낸다. 전체적으로 제안 모듈이 대부분의 연도에서 가장 우수하고 안정적인 성능을 보였으며, 특히 2019 년 이후와 같이 드리프트의 영향이 커지는 구간에서 성능 차이가 두드러졌다. 예를 들어 2019 년의 경우 Pure MLP 의 F1-Score 는 68.1%에 머물렀고, Additive Adapter 도 최대 72.7% 수준에 그친 반면, 제안 모듈은 적응 비율 0.2 기준으로 93.1%까지 향상되었다. Accuracy 역시 동일한 경향을 보여 Pure MLP 는 53.8%, Additive Adapter 는 최대 63.4%, 제안 모듈은 93.1%를 기록하였다.

이와 같은 경향은 2020 년부터 2023 년까지도 일관되게 나타났다. 제안 모듈은 대부분의 연도에서 적응 비율 0.05 이상부터 F1-Score 와 Accuracy 가 크게 상승하여 90% 이상의

성능을 안정적으로 유지하였다. 반면 Additive Adapter 은 일부 성능 개선을 보였지만 향상 폭이 제한적이었고, 기준 모델은 적응을 수행하지 않기 때문에 모든 적응 비율에서 동일한 성능에 머물렀다. 이러한 결과는 Bias(β) 및 Residual(δ)에 기반한 가산형 보정만으로는 분포 변화에 따른 출력 왜곡을 충분히 보정하기 어렵고 스케일 조정(α)과 전역 이동(β), 국소 잔차 보정(δ)을 함께 수행하는 구조가 드리프트 환경에서 더 효과적임을 보여준다.

그림 2. 적응 비율별 모델들의 F1-Score 비교

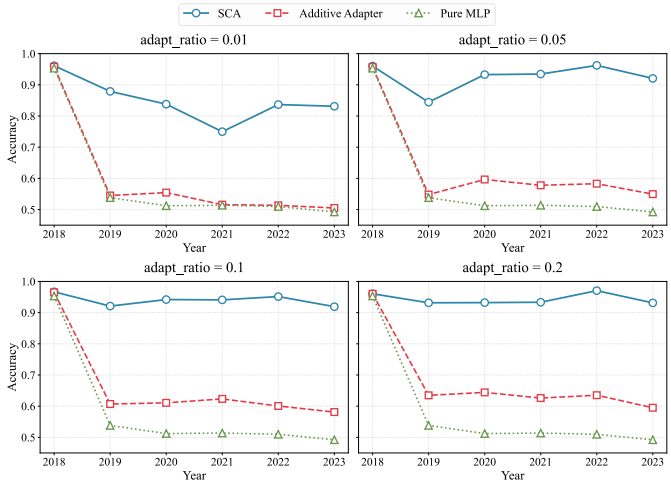

그림 3. 적응 비율별 모델들의 Accuracy 비교

5. 결론 및 논의

본 연구에서 제안한 Calibrated Adapter는 기존의 drift 대응 방식과 구조적으로 다른 접근을 취한다. 표 3은 제안 방법과 기존 연구들을 정성적 기반으로 간접 평가하여 나타낸 것이다.

제안 방법은 동일한 데이터셋을 사용하는 Chen의 방법[1]과 LDCDroid[6]가 각각 API Graph 기반의 재학습 방식으로 F1 85.8%와 68.3%를 달성한 반면, 본 모듈은 Security Descriptor의 도메인 지식과 Logit Calibration을 통한 적응적 학습을 통해 평균 91.8%의 높은 F1 성능을 달성하였다. 이는 제안 모듈이 강한 Concept Drift에도 더욱 잘 버틸 수 있음을 입증한다.

다만 본 연구는 Adaptation 샘플과 Threshold 조정용 샘플을 필요로 하므로 완전한 Unsupervised 적응 방식은 아니며, Security Descriptor 가 악성 행위 의미론에 기반해 사전 정의된 보안 위험 그룹에 의존한다는 한계가 있다. 또한 Backbone 을 고정하는 구조적 특성상 분포 변화가 매우 큰 경우에는 적응 성능이 제한될 수 있다. 향후 연구에서는 자동화된 Descriptor 학습, Unlabeled Target Data 활용, 그리고 다양한 Backbone 구조에 대한 확장을 통해 제안 방법의 일반성과 실용성을 더욱 높일 필요가 있다. 또한 추후 Backbone 구조 및 Security Descriptor 설계에 대한 성능 분석 및 Security Descriptor 의 보안 위험 그룹 정의 기준과 그룹 수에 대한 Ablation 실험을 진행할 예정이다.

표 3. 제안 방법과 기존 연구와의 비교

	Chen[1]	LDCDroid[6]	Proposed
데이터 출처	AndroZoo	AndroZoo	AndroZoo
특징정보	API Graph	API Graph	API Call Frequency
분류기	MLP + Encoder	MLP	MLP
드리프트 대응방식	Retraining	Retraining	Security Description + Logit Calibration
연구 특징점	Active + Contrastive Learning	Active Learning + Pseudo-labeling	Active + Adaptive Learning
F1-Score	85.8%	68.3%	평균 91.8%

참고 문헌

[1] Y. Chen, Z. Ding, and D. Wagner, “Continuous Learning for Android Malware Detection,” in *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 1127-1144, August, 2023.

[2] K. Roh, S. Park, and S.J. Cho, “Drift-Aware Security Module based on Louvain Communities for Retraining-Free Android Malware Detection,” in *2025 Korea Software Congress (KSC 2025)*, pp. 860-862, December, 2025. (in Korean)

[3] B. Raza, A. Maitlo, Z.H. Shar, and I. Hyder, “Operational Android Malware Filtering: Calibrated Probabilities and Distribution-Free Guarantees,” in *Kashf Journal of Multidisciplinary Research*, Vol. 2, No.12, December, 2025.

[4] H. Papadopoulos, N. Georgiou, C. Eliades, and A. Konstantinidis, “Android malware detection with unbiased confidence guarantees,” in *Neurocomputing*, Vol. 280, pp. 3-12, March, 2018.

[5] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, “Dos and Don’ts of Machine Learning in Computer Security,” *in 31st USENIX Security Symposium (USENIX Security 22)*, pp. 3971-3988, August, 2022.

[6] Z. Liu, R. Wang, B. Peng, L. Qiu, Q. Gan, C. Wang, and W. Zhang, “LDCDroid: Learning data drift characteristics for handling the model aging problem in Android malware detection,” in *Computers & Security,* Vol. 150, March, 2025.