Detecção de Malware em Aplicações Android

Diogo A. F. Pinto
Departamento de Engenharia, Telecomunicações e Informática
Universidade de Aveiro
Aveiro, Portugal
pintodiogo@ua.pt

***Abstract* - Este trabalho tem como objetivo replicar a metodologia apresentada no artigo *Malware Dataset Generation and Evaluation* para deteção de malware em aplicações Android, utilizando o dataset *TUANDROMD*. Foram testados cinco algoritmos de Machine Learning, avaliando o desempenho através de validação cruzada e métricas como *accuracy*, *precision*, *recall* e *F1-score*.**

I. Introdução

Nos últimos anos, a crescente utilização de dispositivos móveis tem tornado a segurança digital uma preocupação fundamental. O sistema Android, sendo um dos mais populares, também é um dos mais visados por ataques, onde aplicações maliciosas representam uma ameaça significativa para os utilizadores.

A deteção de *malware* é um desafio, pois cada vez mais existem técnicas avançadas para evitar a identificação. Por isso, abordagens baseadas em *Machine Learning* têm sido amplamente estudadas para melhorar a precisão da deteção.

Neste trabalho, o objetivo é replicar a metodologia apresentada no artigo original, *Malware Dataset Generation and Evaluation*, utilizando o dataset *TUANDROMD*, que contém permissões e chamadas de *API* de aplicações "benignas" (*goodware*) e "maliciosas" (*malware*). Para isso, foram testados 5 algoritmos de *Machine Learning*:

- *Random Forest*
- *Extra Trees*
- *AdaBoost*
- *XGBoost*
- *Gradient Boosting*

Os modelos foram treinados e avaliados com recurso a validação cruzada (*10-Fold Cross-Validation*), comparando os resultados obtidos com os apresentados no artigo original. O principal objetivo deste trabalho é analisar se a metodologia pode ser replicada com sucesso e discutir possíveis variações nos resultados.

II. Metodologia

O objetivo deste estudo foi testar diferentes modelos de *Machine Learning* replicando a metodologia do estudo original, e avaliar o seu desempenho na deteção de *malware* em aplicações *Android*.

O estudo seguiu as seguintes etapas principais:

1. Análise e preparação dos dados
2. Treino de diferentes algoritmos, com recurso a validação cruzada para avaliar a sua *"Accuracy"*
3. Ajuste de hiperparâmetros, testando diferentes configurações
4. Comparação com os resultados do artigo original, verificando a replicabilidade do estudo

A. Dataset

Para este estudo, utilizámos o *TUANDROMD*, um dataset que contém informações sobre aplicações Android, classificadas como "benignas" (*goodware*) ou "maliciosas" (*malware*).

O dataset contém permissões e chamadas de *API´s* associadas a cada aplicação, sendo que cada entrada é representada por um conjunto de *features* binárias, indicando a presença ou ausência de determinadas permissões/chamadas. A coluna "Label" indica a classificação de cada aplicação:

- *0: Goodware*
- *1: Malware*

Ao analisar os dados do dataset foi possível descobrir um desequilíbrio nas classes, onde há significativamente mais amostras de *malware* do que de *goodware*, tal como é possível ver na figura abaixo demonstrada. Este fator pode impactar os modelos e favorecer a classe maioritária. Para mitigar este efeito, utilizámos a técnica de *class_weight="balanced"* em alguns dos algoritmos, ajustando assim os pesos das classes. Para este efeito e de forma a evitar o desequilíbrio das classes também poderá ser usado o *SMOTE (Função do Scikit-Learn)*.

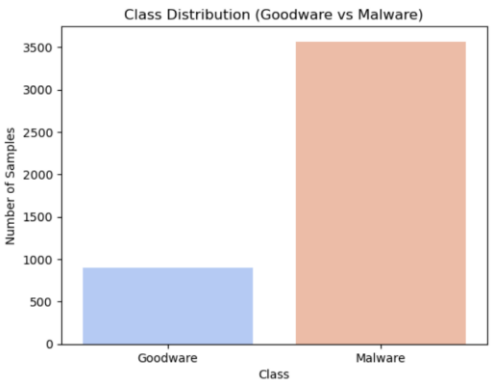

Figura 1 - Distribuição das classes no dataset TUANDROMD

Antes de avançar para o treino dos modelos, foram realizadas as seguintes operações:

- Análise da estrutura do dataset
- Verificação da existência de valores nulos, garantindo que os dados estavam completos
- Divisão das *features* (*X*) e do rótulo (*y*), separando os dados de entrada da classificação
- Análise gráfica da distribuição das classes, com recurso ao *seaborn* e ao *matplotlib* (*Figura 1*)

B. Algoritmos e Hiperparâmetros

Para replicar a metodologia do artigo, utilizámos 5 algoritmos de *Machine Learning*, tal como referido

anteriormente (*Random Forest, Extra Trees, AdaBoost, XGBoost e Gradient Boosting*).

Todos os modelos foram treinados com o dataset *TUANDROMD*, utilizando as permissões e chamadas de *API´s* das aplicações *Android* como features de entrada e a variável *"Label"* como rótulo.

Para otimizar o desempenho dos modelos e garantir resultados comparáveis ao artigo, foram testados diferentes hiperparâmetros para cada algoritmo.

Os principais hiperparâmetros utilizados foram:

- ***n_estimators***: Número de árvores utilizadas nos modelos em ensemble
- ***max_depth***: Profundidade máxima das árvores, que controla a complexidade do modelo
- ***learning_rate***: Taxa de aprendizagem para algoritmos baseados em *boosting*, ajustando assim a contribuição de cada árvore

Sendo que os valores testados foram os seguintes:

- ***n_estimator***s: {100, 200, 300}
- ***max_depth***: {5, 10, None}
- ***learning_rate***: {0.01, 0.1, 0.3}

Cada combinação de hiperparâmetros foi testada individualmente para cada modelo, permitindo identificar a configuração que melhor replicava os resultados apresentados no artigo abordado.

C. Validação dos Modelos

Para garantir a robustez dos resultados e evitar *overfitting*, foi utilizada a técnica de validação cruzada (*Stratified K-Fold Cross-Validation*) com 10 *folds*. Com a validação cruzada é possível ter um melhor aproveitamento dos dados, treinando e testando o modelo em diferentes subconjuntos, sendo que ajuda a reduzir variações estatísticas, produzindo resultados mais estáveis e assegura uma divisão equilibrada das classes em cada *fold*, crucial para datasets desequilibrados como o *TUANDROMD*, usado neste trabalho.

A métrica principal utilizada para a avaliação dos modelos foi a "*accuracy*", para garantir a comparabilidade direta com os resultados do artigo, no entanto, para além disso, foram usadas e analisadas outras métricas:

- ***Precision***: Proporção de deteções corretas de *malware* em relação ao total de previsões positivas
- ***Recall***: Capacidade de o modelo identificar corretamente todas as amostras de *malware*
- ***F1-Score***: Média entre *precison* e *recall*, útil para datasets desequilibrados (como é o caso).

Embora a "*accuracy*" seja a métrica principal e fundamental para comparação com os resultados apresentados no artigo, a mesma pode ser um pouco enganadora num dataset desequilibrado, daí a necessidade de abordar outras métricas.

III. Resultados e Conclusões

Após a implementação e execução dos modelos, foram comparados os resultados obtidos com aqueles reportados no artigo. O objetivo principal era replicar tanto a metodologia comos os resultados para avaliar a consistência e reprodutibilidade do estudo.

Tabela 1 - Comparação de resultados

Algoritmo	Accuracy (estudo)	Accuracy obtida
Random Forest	98.7%	98.7%
Extra Trees	98.8%	98.9%
AdaBoost	97.9%	97.9%
XGBoost	98.8%	97.8%
Gradient Boosting	97.4%	97.4%

Como é possível observar na *Tabela 1*, os valores obtidos são muito próximos dos apresentados no estudo original, indicando que foi conseguido replicar a maioria dos resultados. Por este motivo, é possível afirmar que a metodologia foi replicada com relativo sucesso.

No entanto, apesar do sucesso na replicação dos resultados, existiram alguns desafios durante a implementação. Alguns aspetos do artigo poderiam ter sido mais detalhados para tornar a replicação mais direta e para ter uma ideia mais concreta da metodologia adotada no estudo. Aspetos como, o artigo não especificar claramente se foi ou não realizada alguma normalização ou transformação adicional nas *features* do dataset, que podem influenciar negativamente os resultados finais. Para além deste ponto, o artigo também não menciona que hiperparâmetros foram utilizados, daí a necessidade de realizar múltiplas combinações de em todos os modelos, para encontrar as configurações mais próximas das usadas no estudo.

Para concluir, é sensato dizer que a replicação dos resultados do artigo foi bem-sucedida. Pequenas discrepâncias foram identificadas, mas podem ser justificadas por fatores como diferenças no pré-processamento de dados e variações nas versões das bibliotecas utilizadas. Para futuras replicações, seria benéfico que estudos semelhantes incluíssem informações mais detalhadas sobre:

- Tratamento dos dados antes da modelação
- Hiperparâmetros exatos usados para cada modelo

Com estas melhorias, seria possível garantir uma reprodutibilidade mais rigorosa, facilitando comparações diretas entre diferentes implementações.

References

[1] OpenAI. (2025). ChatGPT (Feb 15 version) [Large language model]. Disponível em: https://chat.openai.com/chat

[2] P. Borah, D. K. Bhattacharyya, and J. K. Kalita, "Malware Dataset Generation and Evaluation," *2020 IEEE 4th Conference on Information & Communication Technology (CICT)*, 2020, pp. 1-6. Disponível em: https://ieeexplore.ieee.org/document/9312053