Análise Espaço-Temporal de Binários Android Usando Representação Byte-to-Video

Caio Kloppel, Jhonatan Geremias, Altair O. Santin e Eduardo K. Viegas

[1]Programa de Pós-Graduação em Informática (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR)
80.215-901 – Curitiba – PR

{caio.kloppel, jhonatan.geremias, santin, eduardo.viegas}@ppgia.pucpr.br

Resumo. *Este artigo apresenta um framework de detecção de malware Android baseado em uma representação espaço-temporal dos arquivos binários das aplicações. A abordagem proposta utiliza uma codificação Byte-to-Video, na qual o arquivo analisado é convertido em uma sequência de frames de alta resolução, preservando sua estrutura original. Essa representação é processada por uma Convolutional Neural Network 3D (3D-CNN), permitindo a extração de características espaciais e de dependências entre diferentes regiões do bytecode. Ao tratar o binário como uma sequência de vídeo, o modelo identifica padrões maliciosos distribuídos ao longo do código sem as perdas introduzidas pelo redimensionamento tradicional de imagens 2D. Os experimentos em um dataset próprio, com mais de 22 mil amostras, mostram uma melhoria de 0,02 no F1-Score em relação às abordagens convencionais. Além disso, a avaliação no dataset CICMalDroid alcançou uma taxa de verdadeiro-positivo média de 0,97 entre diferentes famílias de malware, superando métodos existentes na literatura.*

1. Introdução

Em 2025, o ecossistema Android registrou a identificação de mais de 815 mil novos pacotes de instalação maliciosos distintos por pesquisadores de segurança. Como consequência, ao longo desse mesmo período, foram bloqueadas mais de 14 milhões de tentativas de ataque direcionadas à plataforma [List 2026]. Diante da rápida evolução desse cenário de ameaças, a literatura recente tem direcionado esforços ao desenvolvimento de abordagens mais eficazes para a detecção de malware em dispositivos Android, com destaque para métodos baseados em Machine Learning (ML) e na análise de comportamento das aplicações [Haidros Rahima Manzil and Manohar Naik 2024, de Oliveira et al. 2025].

O emprego de Deep Neural Networks (DNNs) na detecção de malware Android tem se tornado cada vez mais frequente, principalmente por meio de abordagens baseadas em análise estática e dinâmica [Shu et al. 2023]. Enquanto a análise estática examina artefatos da aplicação, como o bytecode, o arquivo de `manifest` e outras características estruturais, sem a necessidade de executar o aplicativo, a análise dinâmica fundamenta-se na observação do comportamento da aplicação durante sua execução em ambientes controlados, monitorando informações como chamadas de sistema, comunicação em rede e demais atividades em tempo de execução [Geremias et al. 2023]. Embora ambas as estratégias apresentem vantagens, a análise estática permanece como a alternativa predominante na literatura devido ao seu menor custo computacional e à eliminação da necessidade de ambientes específicos para execução das aplicações. Nesse contexto, uma

prática amplamente adotada consiste em transformar o malware em uma representação visual, permitindo explorar a elevada capacidade de extração automática de características oferecida por arquiteturas de visão computacional baseadas em DNNs [Geremias et al. 2022]. Nesse processo, o arquivo Dalvik Executable (`classes.dex`), responsável por armazenar o código compilado da aplicação Android, é interpretado como uma sequência de bytes que posteriormente é reorganizada em uma matriz bidimensional, originando imagens em escala de cinza ou no formato RGB. A partir dessa representação, as DNNs tornam-se capazes de identificar padrões espaciais e características estruturais associadas a comportamentos maliciosos, viabilizando a detecção automatizada de malware sem depender de etapas de engenharia manual de *features*.

Um dos principais desafios das abordagens de detecção de malware baseadas em imagens está relacionado à grande variação no tamanho dos arquivos `classes.dex`, o que naturalmente resulta na geração de imagens com dimensões distintas [Vasan et al. 2020]. Como as arquiteturas convencionais de DNN exigem entradas de tamanho fixo, a estratégia mais comum consiste em redimensionar essas imagens para uma resolução previamente definida [Dong et al. 2026]. Embora essa padronização simplifique o processamento dos dados, ela pode comprometer informações relevantes presentes na representação original. Em especial, padrões estruturais e sequências de bytes de alta granularidade tendem a ser distorcidos ou eliminados durante o redimensionamento, sobretudo em aplicações cujo bytecode é extenso ou apresenta elevada complexidade [Espindola et al. 2026b]. Logo, a capacidade do modelo de aprender características discriminativas é reduzida, prejudicando a generalização e diminuindo a eficácia na identificação de variantes de malware não observadas durante o treinamento [Espindola et al. 2026a]. Uma alternativa promissora para contornar essa limitação consiste no uso de arquiteturas *Convolutional Neural Networks* tridimensionais (3D-CNNs), capazes de processar múltiplas imagens de forma conjunta e de explorar dependências espaciais entre elas [Yapici 2025]. Ao preservar relações estruturais que normalmente se perdem em representações bidimensionais, essas redes têm potencial para melhorar a qualidade da representação do malware. Apesar dessa vantagem, o emprego de 3D-CNNs ainda é pouco investigado na literatura dedicada à detecção de malware em dispositivos Android.

Entretanto, os principais datasets utilizados na literatura para detecção de malware Android normalmente disponibilizam os arquivos Android Application Package (APK) apenas na forma de imagens previamente processadas e padronizadas a uma resolução fixa [Vasan et al. 2020]. Como consequência, essas bases preservam apenas versões redimensionadas das aplicações, eliminando as informações presentes na estrutura original dos binários e removendo a variabilidade dimensional inerente aos arquivos DEX [Geremias et al. 2026]. Essa limitação imposta pelos próprios conjuntos de dados limita o desenvolvimento de novas estratégias de detecção capazes de explorar entradas de dimensões variáveis. Na prática, a ausência das representações espaciais originais dificulta a investigação e a avaliação de arquiteturas projetadas para acomodar a heterogeneidade dos binários Android, uma vez que os pesquisadores deixam de ter acesso às informações necessárias para explorar integralmente as características estruturais dos arquivos `classes.dex`.

Contribuição. Com o objetivo de superar essas limitações, este trabalho apresenta um novo método de detecção de malware Android com consciência espacial, composto por

três etapas principais. Inicialmente, o arquivo APK é transformado em um conjunto de imagens obtidas diretamente a partir do conteúdo binário do arquivo `classes.dex`, preservando a organização original dos bytes, sem necessidade de redimensionamento da representação. Na etapa seguinte, as imagens produzidas são organizadas sequencialmente para formar uma representação em formato de vídeo, permitindo modelar de forma mais completa a estrutura espacial do código presente no `classes.dex`. Por fim, essa sequência é utilizada como entrada de uma arquitetura 3D-CNN, responsável por aprender simultaneamente relações espaciais e temporais e, a partir delas, classificar a aplicação como benigna ou maliciosa. A principal contribuição desta proposta consiste em reinterpretar o binário de aplicações Android como uma sequência temporal de informações, em vez de uma única imagem estática. Essa representação baseada em vídeo possibilita explorar a dimensão adicional oferecida pelas 3D-CNNs, reduzindo as perdas de informação associadas ao redimensionamento bidimensional tradicional e favorecendo uma maior capacidade de generalização na detecção de amostras de malware com diferentes características estruturais.

Em resumo, a principal contribuição deste trabalho é um novo modelo de detecção de malware Android com consciência espacial implementado por meio de um esquema 3D-CNN. Nosso modelo proposto melhora o F1-Score em até $0{,}03$ em nosso dataset e aumenta o true-positive médio da detecção de família de malware em $0{,}04$ em um dataset de benchmark em comparação à literatura.

2. Trabalhos Relacionados

Nos últimos anos, a detecção de malware Android tem sido amplamente impulsionada pela adoção de modelos de *deep learning* e de arquiteturas híbridas, com o objetivo de aumentar tanto a precisão quanto a robustez dos sistemas de classificação [Shu et al. 2023]. Nesse cenário, diferentes estratégias têm sido propostas para incorporar informações complementares às DNNs, explorando representações derivadas de grafos de chamadas de API, permissões do sistema, padrões de comportamento das aplicações e características extraídas de tráfego de rede criptografado [Dong et al. 2024, Zhang et al. 2023, Haq et al. 2021].

Entre as abordagens mais representativas encontram-se arquiteturas baseadas em CNNs multiescala, desenvolvidas para integrar múltiplos tipos de representação em um único processo de aprendizado [Dong et al. 2024]. Paralelamente, diversos estudos combinam redes neurais convolucionais com modelos recorrentes, como BiLSTM, LSTM e GRU, buscando capturar relações temporais e sequenciais presentes na execução de aplicações Android [Haq et al. 2021, Usama Tanveer et al. 2025, Misalkar and Harshavardhanan 2025]. Outra direção de pesquisa envolve o emprego de técnicas de *ensemble learning* associadas à estimativa de incerteza, o que permite reduzir erros de classificação, identificar amostras inconsistentes e detectar anomalias em bases de malware [Li et al. 2026, Li et al. 2025, Musikawan et al. 2023].

Além disso, modelos fundamentados em Recurrent Neural Networks (RNNs) têm sido utilizados para acomodar a natureza variável dos arquivos APK, principalmente por meio da modelagem de sequências de chamadas de API e de informações comportamentais coletadas durante a execução das aplicações [Usama Tanveer et al. 2025, Misalkar and Harshavardhanan 2025]. Embora essas soluções apresentem resultados expressivos,

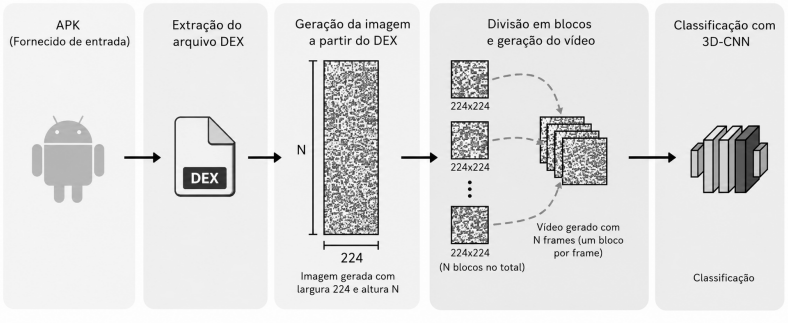

Figura 1. Arquitetura de alto nível do modelo proposto.

observa-se que a maioria ainda depende de representações de entrada com dimensões fixas, de etapas complexas de pré-processamento e de conjuntos de dados previamente padronizados. Essas restrições podem eliminar informações estruturais importantes dos binários, comprometendo a capacidade dos modelos de generalizar para novas variantes e famílias de malware que surgem continuamente [Dong et al. 2024, Li et al. 2026, Zhang et al. 2023].

3. Modelo Proposto

Com o objetivo de superar as limitações discutidas anteriormente, este trabalho propõe uma abordagem de detecção de malware para Android baseada em representações de vídeo. A arquitetura do método, ilustrada na Figura 1, é composta por três módulos principais que atuam de forma sequencial.

Na primeira etapa, denominada *Geração de Imagem*, o arquivo `classes.dex` é extraído do APK e convertido em um conjunto de imagens de dimensões variáveis por meio de um mapeamento direto entre bytes e pixels. Em seguida, o módulo *Geração de Vídeo* organiza essas imagens em uma sequência de quadros, formando uma representação em vídeo capaz de preservar a organização espacial do código da aplicação, sem recorrer a técnicas convencionais de redimensionamento. Por fim, o módulo *Classificação* recebe essa sequência como entrada de uma arquitetura 3D-CNN, responsável por aprender simultaneamente relações espaciais e temporais para determinar se o APK corresponde a um *malware* ou a um *goodware*.

A principal inovação da proposta reside na interpretação do conteúdo binário das aplicações Android como uma sequência temporal de informações, em vez de uma representação bidimensional isolada. Essa mudança de perspectiva permite explorar o poder de modelagem das convoluções tridimensionais, preservando detalhes estruturais que normalmente são perdidos durante o redimensionamento de imagens. Como resultado, o modelo mantém informações de granularidade fina presentes no binário original, favorecendo uma melhor capacidade de generalização diante da elevada diversidade estrutural observada entre diferentes famílias de malware Android.

3.1. Geração de Imagem

Nas abordagens convencionais de detecção de malware Android baseadas em imagens, a análise do arquivo `classes.dex` geralmente depende do redimensionamento das representações visuais para compatibilizá-las com as dimensões de entrada exigidas pelas arquiteturas tradicionais de DNNs. Embora esse procedimento facilite o processamento

dos dados, ele também introduz perdas de informação, uma vez que padrões estruturais e detalhes presentes no nível dos bytes podem ser distorcidos ou eliminados. Como consequência, a capacidade de generalização dos modelos tende a ser reduzida, sobretudo diante da elevada diversidade entre as famílias de malware. Para contornar essa limitação, o módulo *Geração de Imagem* (Figura 1) gera múltiplas imagens diretamente a partir do conteúdo binário do arquivo `classes.dex`. Dessa forma, a proposta preserva a organização espacial original dos dados e mantém a integridade dimensional do código, evitando a perda de informações decorrente das técnicas tradicionais de redimensionamento.

Considere um arquivo `classes.dex` representado por $\mathcal{D}$, definido como uma sequência finita de bytes, $\mathcal{D} = \{b_1, b_2, \ldots, b_n\}$, em que n é o tamanho total do arquivo. O objetivo desta etapa é converter essa sequência binária em um conjunto de imagens $\mathcal{I} = \{I_1, I_2, \ldots, I_k\}$, de modo que cada imagem represente uma porção específica do bytecode, preservando as informações originais sem recorrer ao redimensionamento da representação completa. Para isso, a sequência $\mathcal{D}$ é dividida em k segmentos contíguos. Cada segmento é então transformado em uma imagem $I_j \in \mathbb{R}^{H \times W}$ por meio de um mapeamento direto entre bytes e pixels, em que H e W são a altura e a largura da imagem, respectivamente. Essa estratégia permite representar o conteúdo binário em múltiplas imagens de tamanho fixo, preservando a distribuição espacial de cada segmento. Como resultado, as dependências estruturais locais e os padrões espaciais presentes no arquivo original são preservados com maior fidelidade, fornecendo uma representação mais rica para as etapas posteriores de modelagem espaço-temporal.

3.2. Geração do Vídeo

Embora cada imagem de $\mathcal{I}$ represente apenas uma fração do arquivo `classes.dex`, a decisão de classificação deve considerar todo o conteúdo do APK. Para isso, as imagens produzidas na etapa anterior são reunidas em uma única representação sequencial por meio do módulo *Geração do Vídeo* (Figura 1), o que permite a análise integrada de toda a estrutura binária.

A estratégia proposta consiste em organizar as imagens na ordem em que seus respectivos segmentos aparecem no arquivo original, formando uma sequência que preserva a continuidade do bytecode. Dessa maneira, a informação espacial passa a ser representada como um volume espaço-temporal, no qual cada imagem corresponde a um quadro (*frame*) da evolução do conteúdo do `classes.dex`. Essa representação possibilita o uso de arquiteturas 3D-CNN, capazes de aprender simultaneamente padrões espaciais presentes em cada quadro e as relações existentes entre quadros consecutivos, produzindo uma única decisão de classificação para toda a aplicação.

Formalmente, o conjunto de imagens $\mathcal{I} = \{I_1, I_2, \ldots, I_k\}$ é convertido em um tensor tridimensional $\mathcal{V} \in \mathbb{R}^{k \times H \times W}$, em que k corresponde ao número de quadros da sequência, enquanto H e W representam, respectivamente, a altura e a largura de cada imagem.

3.3. Classificação

O módulo de classificação foi desenvolvido para analisar a estrutura completa de uma aplicação Android, explorando as relações espaciais presentes em todo o bytecode, representado por uma sequência de imagens. Em vez de avaliar cada imagem de forma

independente, a proposta considera o conjunto $\mathcal{I}$ como uma única entidade, preservando a continuidade entre os segmentos que compõem o arquivo `classes.dex`.

Com esse objetivo, o conjunto de imagens é representado pelo volume de vídeo $\mathcal{V}$, que constitui a entrada do módulo *Classification* (Figura 1). Com base nessa representação, uma arquitetura 3D-CNN é empregada para produzir uma única decisão de classificação para todo o APK. Ao processar simultaneamente as dimensões espaciais e temporais da sequência, o modelo é capaz de aprender representações hierárquicas que capturam padrões estruturais complexos distribuídos em múltiplos quadros, o que favorece que características associadas a famílias sofisticadas de malware sejam identificadas.

Em cada camada convolucional da 3D-CNN, filtros tridimensionais são aplicados ao volume de entrada para extrair simultaneamente informações ao longo das dimensões espaciais e temporais. Durante esse processo, cada ativação é obtida pela combinação ponderada dos valores da vizinhança local do volume de entrada, acrescida de um termo de viés e seguida da aplicação de uma função de ativação não linear. Dessa forma, a rede aprende automaticamente padrões distribuídos entre quadros consecutivos, capturando relações que não seriam observadas por meio de convoluções bidimensionais convencionais.

À medida que os dados percorrem as sucessivas camadas de convolução e as operações de *pooling*, a representação inicial é progressivamente condensada em um vetor de características de alto nível. Esse vetor é então processado por camadas totalmente conectadas, responsáveis por produzir a classificação final $y \in \{0, 1\}$, indicando se o APK corresponde a um *malware* ou a um *goodware*. Essa estratégia permite que a decisão seja fundamentada na organização estrutural do arquivo `classes.dex` como um todo, em vez de depender exclusivamente de padrões locais ou de fragmentos isolados do código.

O treinamento da 3D-CNN é supervisionado, utilizando um conjunto de aplicações benignas e maliciosas para ajustar os parâmetros da rede. Durante esse processo, o modelo busca reduzir a discrepância entre as probabilidades previstas e os rótulos reais de cada APK, refinando iterativamente sua capacidade de discriminar padrões associados a diferentes classes.

A otimização dos parâmetros é conduzida por meio do algoritmo de *backpropagation*, com a função de perda *Binary Cross-Entropy* (BCE). Essa função quantifica o erro entre as probabilidades produzidas pelo modelo e as respectivas classes verdadeiras, fornecendo o sinal necessário para atualizar os pesos da rede a cada iteração do treinamento. À medida que esse erro é minimizado, a arquitetura aprende representações cada vez mais discriminativas da organização espaço-temporal do bytecode, tornando-se capaz de identificar assinaturas estruturais complexas características de aplicações maliciosas.

3.4. Discussão

A proposta apresentada introduz uma nova perspectiva para a detecção de malware Android ao tratar o arquivo `classes.dex` como uma sequência estruturada de informações, em vez de uma representação bidimensional isolada. Essa mudança permite preservar características do binário que normalmente são perdidas nos processos convencionais de conversão e redimensionamento de imagens. Ao utilizar uma representação baseada em vídeo, a abordagem explora a capacidade das 3D-CNNs de modelar dependências entre diferentes regiões do bytecode, combinando informações espaciais e

temporais em uma única etapa de classificação. Dessa forma, o método reduz a dependência de técnicas de pré-processamento, que podem introduzir distorções nos dados, e fornece uma representação mais próxima da estrutura original das aplicações Android. Além disso, a estratégia proposta apresenta potencial de aplicação em cenários em que a diversidade dimensional dos arquivos e a evolução contínua das famílias de malware representam desafios relevantes para os métodos tradicionais baseados em imagens.

4. Avaliação

Nosso conjunto conduzido de experimentos visa responder às seguintes questões de pesquisa (QP):

- **QP1.** *Qual é o desempenho de detecção de malware Android das estratégias 2D-CNN tradicionais?*
- **QP2.** *Como nosso esquema proposto melhora a acurácia de detecção de malware Android?*

4.1. Um Dataset Realista de Malware Android

Uma limitação recorrente dos datasets disponíveis para detecção de malware Android é a disponibilização dos APKs apenas como imagens previamente redimensionadas, o que inviabiliza a avaliação de métodos que preservam a estrutura original dos arquivos `classes.dex`. Para contornar essa restrição, construímos um novo dataset no qual cada aplicação é convertida diretamente do binário em um conjunto de imagens de resolução 224×224. O número de imagens geradas varia conforme o tamanho do arquivo, preservando integralmente a organização estrutural do bytecode e fornecendo uma representação adequada para o processamento espaço-temporal. O dataset contém $22{,}058$ aplicações, sendo $11{,}107$ *goodwares* e $10{,}951$ *malwares*, coletadas do AndroZoo ao longo de cinco anos. A rotulagem foi validada com o VirusTotal, considerando como *malware* as amostras detectadas por pelo menos dois mecanismos antivírus distintos.

A avaliação experimental foi realizada utilizando as arquiteturas ResNet-18 e DenseNet121 em versões 2D e 3D, implementadas em PyTorch v2.1.0. Os modelos foram treinados por 100 épocas com o otimizador Adam, empregando *learning rate* inicial de 1×10^{-5}, *batch size* de 32, *weight decay* de 1×10^{-5} e um escalonador para ajuste automático da taxa de aprendizado. O dataset foi dividido de forma estratificada em treinamento (60%), validação (20%) e teste (20%), preservando o balanceamento entre aplicações benignas e maliciosas. O desempenho dos modelos foi analisado por meio das métricas True Positive Rate (TPR), True Negative Rate (TNR) e F1-Score.

4.2. Detecção de Malware

O primeiro experimento foi conduzido para responder à *QP1*, avaliando o desempenho de abordagens convencionais de detecção de malware baseadas em 2D-CNN. Para isso, implementamos as arquiteturas ResNet e DenseNet utilizando o procedimento tradicional adotado na literatura, no qual o conteúdo do arquivo `classes.dex` é convertido em uma única imagem posteriormente redimensionada para a resolução de 224×224. O objetivo desse experimento é estabelecer uma baseline de desempenho para métodos baseados em representações bidimensionais, permitindo uma comparação direta com a abordagem proposta neste trabalho, que preserva a estrutura original do binário por meio de uma representação espaço-temporal.

Tabela 1. Comparação de Desempenho das Arquiteturas 2D e 3D em nosso dataset construído.

Abordagem	Arquitetura	AUC	F1	TNR	TPR
Tradicional 2D-CNN	DenseNet	0,96	0,93	0,93	0,95
	ResNet	0,96	0,93	0,93	0,95
Nossa 3D-CNN	DenseNet	0,98	0,95	0,96	0,95
	ResNet	0,98	0,95	0,95	0,95

Os resultados obtidos pelas abordagens baseadas em 2D-CNN são apresentados na Tabela 1. Observa-se que tanto a ResNet quanto a DenseNet alcançaram um F1-Score de $0{,}93$, estabelecendo uma baseline sólida para a tarefa de detecção de malware Android. Apesar do bom desempenho, esse resultado evidencia uma limitação das estratégias tradicionais baseadas em imagens. Como essas arquiteturas exigem entradas de dimensões fixas, a representação do arquivo `classes.dex` precisa ser redimensionada antes do processamento. Esse procedimento pode distorcer ou eliminar padrões estruturais presentes no bytecode, reduzindo a quantidade de informação disponível para o modelo e limitando seu potencial de classificação.

O segundo experimento foi desenvolvido para responder à *QP2*, avaliando o desempenho da abordagem proposta baseada em 3D-CNN. Para isso, utilizamos as mesmas arquiteturas empregadas no experimento anterior (ResNet e DenseNet), adaptadas para processar a representação em vídeo gerada pelo método proposto, conforme ilustrado na Figura 1. Os resultados apresentados na Tabela 1 demonstram que a estratégia proposta supera consistentemente as versões baseadas em 2D-CNN. Em ambas as arquiteturas, observou-se um incremento de $0{,}02$ no F1-Score, indicando que a representação espaço-temporal preserva informações relevantes que são perdidas nas abordagens convencionais. Esse ganho decorre principalmente da eliminação da etapa de redimensionamento para uma única imagem, permitindo que o modelo analise o bytecode mantendo sua organização estrutural original. Além disso, o uso de convoluções tridimensionais possibilita a extração de padrões distribuídos ao longo de múltiplos quadros da sequência, capturando relações espaciais e temporais que não podem ser modeladas por arquiteturas bidimensionais. Esse comportamento também é evidenciado pela Figura 2, que apresenta as curvas Receiver Operating Characteristic (ROC) da DenseNet que obteve a melhor taxa de acerto. A estratégia proposta obteve valores superiores de Area Under the Curve (AUC) em relação às respectivas versões 2D, reforçando a eficácia da representação baseada em vídeo para a detecção de malware Android.

4.3. Discussão

Os resultados obtidos demonstram que preservar a organização estrutural do arquivo `classes.dex` tem impacto direto na capacidade de detecção de malware Android. Ao substituir a representação tradicional baseada em uma única imagem por uma sequência de imagens processada por uma 3D-CNN, o modelo passa a explorar relações espaciais e temporais que permanecem inacessíveis às arquiteturas bidimensionais. Embora o ganho absoluto de desempenho nas métricas de classificação seja relativamente modesto, sua consistência entre diferentes arquiteturas indica que a melhoria decorre da representação proposta, e não de um modelo específico. Esses resultados sugerem que a preservação da estrutura original do bytecode fornece informações discriminativas adicionais, favore-

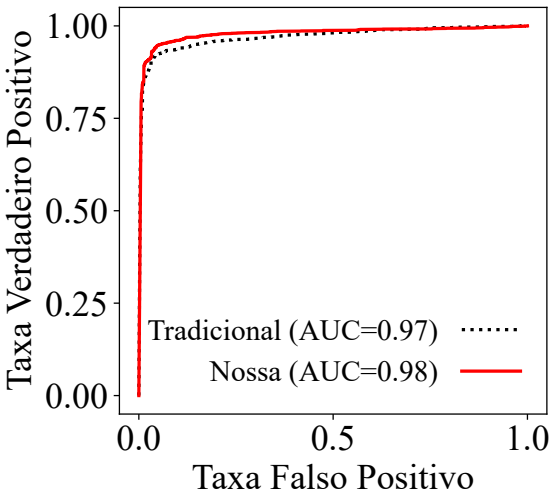

Figura 2. Curva ROC do nosso esquema $vs.$ 2D-CNN tradicional utilizando a DenseNet.

cendo a generalização do classificador diante da elevada variabilidade entre famílias de malware Android e evidenciando o potencial da abordagem para aplicações práticas de detecção.

5. Conclusão

Neste artigo, apresentamos um novo framework espaço-temporal para detecção de malware Android que busca mitigar a perda de informação presente nas abordagens tradicionais de análise binária baseada em imagens. Ao substituir a representação estática 2D por um esquema de codificação *Byte-to-Video*, a proposta preserva a estrutura original do arquivo `classes.dex`, permitindo que uma arquitetura 3D-CNN explore relações espaciais e temporais entre diferentes segmentos do bytecode. Os resultados experimentais, obtidos em um novo dataset de larga escala, demonstram que a abordagem proposta supera consistentemente os métodos tradicionais baseados em 2D-DNN. Além de apresentar melhorias no F1-Score, o modelo evidencia maior capacidade de generalização em diferentes famílias de malware Android.

Agradecimentos

Este trabalho foi parcialmente financiado pela Fundação Araucária e pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), sob os números de processo 302937/2023-4, 407879/2023-4, 442262/2024-8, 406603/2025-1 e 307706/2025-7.

Referências

de Oliveira, V. M., de Oliveira, H. M., Santos, G. M., Geremias, J., and Viegas, E. K. (2025). A big data framework for scalable and cross-dataset capable machine learning in network intrusion detection systems. *IEEE Access*, 13:129419–129431.

Dong, S., Shu, L., and Huang, J. (2026). A novel detection method for unknown android malware via image representation. *Journal of Information Security and Applications*, 99:104451.

Dong, S., Shu, L., and Nie, S. (2024). Android malware detection method based on cnn and dnn bybrid mechanism. *IEEE Transactions on Industrial Informatics*, 20(5):7744–7753.

Espindola, A. d. S., Casimiro, A., Santin, A. O., Ferreira, P. M., and Viegas, E. K. (2026a). Enhancing intrusion detection generalization via diversity-driven multi-view ensemble learning in industrial systems. *Future Generation Computer Systems*, 182:108458.

Espindola, A. d. S., Santin, A. O., Casimiro, A., Ferreira, P. M., and Viegas, E. K. (2026b). Understanding the adversary: A survey of adversarial machine learning in network intrusion detection. *Computer Science Review*, 62:100995.

Geremias, J., Britto, A. S., Santin, A. O., and Viegas, E. K. (2026). Sparse mixture of experts for image-based multi-view android malware detection. In *2026 International Wireless Communications and Mobile Computing (IWCMC)*, page 1487–1492. IEEE.

Geremias, J., Viegas, E. K., Santin, A. O., Britto, A., and Horchulhack, P. (2022). Towards multi-view android malware detection through image-based deep learning. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, page 572–577. IEEE.

Geremias, J., Viegas, E. K., Santin, A. O., Britto, A., and Horchulhack, P. (2023). Towards a reliable hierarchical android malware detection through image-based cnn. In *2023 IEEE 20th Consumer Communications amp; Networking Conference (CCNC)*, page 242–247. IEEE.

Haidros Rahima Manzil, H. and Manohar Naik, S. (2024). Detection approaches for android malware: Taxonomy and review analysis. *Expert Systems with Applications*, 238:122255.

Haq, I. U., Khan, T. A., and Akhunzada, A. (2021). A dynamic robust dl-based model for android malware detection. *IEEE Access*, 9:74510–74521.

Li, H., Cheng, X., Wang, L., and Wang, H. (2026). Towards improved dnn-based android malware detection via uncertainty estimation. *ACM Transactions on Software Engineering and Methodology*.

Li, H., Cheng, X., Zhang, G., Xu, G., Xu, G., and Wang, H. (2025). Mitigating emergent malware label noise in dnn-based android malware detection. *Proceedings of the ACM on Software Engineering*, 2(FSE):1136–1159.

List, S. (2026). The mobile threat landscape in 2025.

Misalkar, H. D. and Harshavardhanan, P. (2025). Tdbamla: Temporal and dynamic behavior analysis in android malware using lstm and attention mechanisms. *Computer Standards & Interfaces*, 92:103920.

Musikawan, P., Kongsorot, Y., You, I., and So-In, C. (2023). An enhanced deep learning neural network for the detection and identification of android malware. *IEEE Internet of Things Journal*, 10(10):8560–8577.

Shu, L., Dong, S., Su, H., and Huang, J. (2023). Android malware detection methods based on convolutional neural network: A survey. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 7(5):1330–1350.

Usama Tanveer, M., Munir, K., Alabdulatif, A., Najdawi, A. R., and Jhaveri, R. H. (2025). Malware-seqguard: An approach utilizing lstm and gru for effective detection of evolving malware in android environments. *IEEE Access*, 13:117355–117373.

Vasan, D., Alazab, M., Wassan, S., Naeem, H., Safaei, B., and Zheng, Q. (2020). Imcfn: Image-based malware classification using fine-tuned convolutional neural network architecture. *Computer Networks*, 171:107138.

Yapici, M. M. (2025). 3dmaldroid: A novel 3d image based approach for android malware detection and classification. *Computers and Electrical Engineering*, 127:110542.

Zhang, X., Wang, J., Xu, J., and Gu, C. (2023). Detection of android malware based on deep forest and feature enhancement. *IEEE Access*, 11:29344–29359.