I.J. Modern Education and Computer Science, **2026, 5, 37-67**
Published Online on October 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2026.05.03

SECURE-XED: A Unified Explainable CNN Framework with Hybrid Metaheuristic Optimization for Cross-Domain Malware and Deepfake Detection

Mohammad Othman Nassar
College of Information Technology, Department of Cyber Security, Amman Arab University, Amman, Jordan
Email: moanassar@aau.edu.jo
ORCID iD: https://orcid.org/0000-0003-2307-9033

Received: June 27, 2025; Revised: October 3, 2025; Accepted: August 20, 2026; Published: October 8, 2026

Abstract: We present SECURE-XED, a unified, explainable, and adversarial-aware learning system for two traditionally disjoint domains: Android malware classification and deepfake detection. The system uses a shared convolutional neural network backbone with domain-specific input adaptation and two task-specific heads. It integrates SHapley Additive exPlanations for explanation, the Fast Gradient Sign Method and Projected Gradient Descent for adversarial evaluation under Structural Similarity Index Measure guardrails, and a selective-activation controller that enables behavioral malware features when confidence is low or evasion is suspected. A hybrid Seagull Optimization Algorithm–Imperialist Competitive Algorithm procedure jointly tunes clean accuracy, robustness under projected-gradient attacks, expected calibration error, parameter count, and inference latency. SECURE-XED is evaluated on standard Android malware corpora and the FaceForensics++, Celeb-DF, and Deepfake Detection Challenge benchmarks. Under Projected Gradient Descent, Android malware accuracy decreases by approximately 19–21% with a perturbation budget of 0.1 over 40 steps, while deepfake accuracy decreases by 16.6–17.6% with a perturbation budget of 0.03 over 10 steps, demonstrating domain-dependent adversarial sensitivity. Under clean deepfake inference conditions, excluding post-prediction explanation generation and adversarial processing, the unified configuration reduces average inference latency from 34.1 to 26.9 milliseconds per frame and approximately halves the parameter requirement relative to maintaining separate model instances, while retaining equal or slightly better clean accuracy. A classroom-oriented simulator exposes explanation overlays and adversarial controls on real frames. A six-participant human-grounded pilot descriptively showed a 22-percentage-point change in Region Identification Accuracy, a 12-second reduction in decision time, and a 0.9-point Likert change in perceived clarity and trust; these pilot outcomes are descriptive and not inferential. The resulting framework combines shared cross-domain representation, multi-objective optimization, adversarial evaluation, explainability, and model-in-the-loop educational interaction within a single modular architecture.

Index Terms: Explainable AI, Android Malware Detection, Deepfake Detection, Adversarial Robustness, SHAP, Metaheuristic Optimization, Educational Simulator.

1. Introduction

Due to the rapid development of digital technologies, modern computers have unprecedented performance power and hazardous potential. In particular, Android malware is progressively exploring new vulnerabilities through elaborate attack vectors resulting in increased security breaches within the mobile platform ecosystem [1]. Concurrently, the proliferation of synthetic media, commonly referred to as deepfake, presents a different but equally dangerous challenge that threatens the sanctity of verifiable information as these falsified media distort the credibility of digital content in academic research and political debates [2,3]. These emerging threats call for the creation of cognitive systems that demand high fidelity and robustness with model transparency. Explainable artificial intelligence (XAI) is a must-have feature because XAI enables medical professionals and ordinary users to verify the decision-making logic of these models.

The classical pipeline involving handcrafted features and shallow classifiers is not easily tuned to the continuously evolving malware threat landscape [4]. Although deep learning algorithms offer better predictive performance, the quality of these systems critically depends on their architectural design since such variants are not readily programmable in complex optimization spaces. SOA and ICA are reported to have robust global exploration abilities, especially when

This work is open access and licensed under the Creative Commons CC BY 4.0 License.

applied in combined continuous-discrete search spaces, and automatically tune model parameters, enhancing model performance against Android malware [5,6].

State-of-the-art deepfake detection systems benefit from the combined advances in neural network architectures and large-scale data resources. The black-box nature of such deep learning models is a hindrance to their adoption in critical security applications where explainability is paramount [7-9]. Such an observation is particularly concerning given the lack of theoretical foundations for most post hoc explanation techniques. Therefore, there is a need for model developers to embed explainability directly into the design process instead of retrofitting existing systems with ad-hoc solutions [10,11]. The robustness properties of such systems to adversarial attacks are mostly ignored in most prior works since model explainability and integrity are usually considered independent targets.

The current work introduces SECURE-XED (Smart Explainable CNN for Unified Robust Education and eXecution) to solve these problems. SECURE-XED is a combined deep learning structure that handles Android malware detection and deepfake analysis at the same time. A shared Convolutional Neural Network (CNN) foundational structure is used by the system to share knowledge and reduce extra work in model design. A hybrid strategy using SOA and ICA helps to fix the architecture and the training settings. Predictive accuracy and the ability to stay strong against attacks are prioritized while the system is optimized under multiple goals [5,14].

The SECURE-XED proposed in this study offers data interpretation capabilities and can perform SHAP-based attack tests. For example, the proposed model incorporates methods such as Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) to help achieve this goal by enabling real-time analysis while simultaneously considering potential attacks. The rationale of the current research for incorporating the proposed elements is to enhance model applicability and facilitate user understanding. In order to enable and support the learning and training process, an interactive simulator is introduced within SECURE-XED. It is primarily used to track the model behavior closely to enable a better understanding of its functioning, such as the impact on data due to attacks. One of the key objectives of the proposed design is to make AI learning affordable to the low-income group and students, thus creating media-literate populations through active involvement in the learning and training process.

In general, the main contributions of the current research can be highlighted with providing a single framework for both Android malware detection and deepfake video/image detection, which is designed to minimize hardware requirements. In particular, the research introduces a system that is able to process each type of data separately while being incorporated within the same neural network architecture. Specifically, the processed data is redirected to Shared CNN Backbone followed by the corresponding malware detection or deepfake detection modules. Overall, the model is designed to minimize the overhead of the system while enabling efficient detection, including the application of the shared layers for malware and deepfake detection. Second, the model aims to achieve a balance between several elements, such as minimizing processing time, predictive reliability, and resistance to attacks, especially PGD attacks. This balance is achieved under explicit constraints on response time, parameters, and cognitive quality [5, 14]. To achieve this, we used two algorithms: SOA and ICA. Third, interpreting the model's decisions and testing attacks so that the system can provide predictions, interpret them, and test its vulnerability to attacks within a single framework. To achieve this, we incorporated a set of techniques, including SHAP, to explain to the user why the model classified an image as fake or a file as malware by identifying the most influential characteristics in the decision. We also included FGSM to test how the model is affected by a single attack modification to the data, and we used PGD to test the model's vulnerability to attacks based on the idea of implementing repeated and deliberate modifications to enable the attacker to succeed in these attacks. These procedures allow us to examine the model's decisions and its vulnerability to adversaries within the same framework [11, 15]. Fourth, we developed an interactive learning simulator that allows students to interact with the model. Students can select an image, view the SECURE-XED decision, understand the SHAP interpretation, experiment with FGSM and PGD, and then observe the impact of attacks on the outcome. The learning simulator user experience was evaluated in the following areas to understand how users interpreted it, identify areas for improvement, measure decision-making time, assess the clarity of the results, and determine how confident users were in those results.

2. Related Work

Previously, most studies related to malware detection on Android employed Convolutional Neural Networks (CNNs) featuring static and dynamic properties. Static features are useful for detecting malware without executing the application. Some of these features are API calls and permissions, which also offer better detection power than traditional machine learning methods [4, 16]. Dynamic features provide behavioral hints, such as system calls and network activity, some of which point to threats that static analysis may miss. However, these features require executing and monitoring the application. This has therefore increased the complexity, cost and operational burden [17].

Optimization techniques for metaheuristics, specifically SOA and ICA, have shown improvements in cybersecurity applications, including convergence, feature selection and classification accuracy [18–22]. These techniques allow CNN configuration optimization to be automated, including the selection of network depth, filters and training parameters. In SECURE-XED, SOA is used to explore architectural and training configurations, ICA refines the malware feature subset, and lean pruning is applied to the vision-based models. Hence, the goal of stacking SOA and ICA is to reduce computational complexity while maintaining high detection performance. Consequently, SECURE-XED integrates the

optimization of detection, explainability and adversarial evaluation within a unified framework for network architectures and end-to-end learning.

The proposed SECURE-XED system allows for decision interpretability using SHAP. So, when a certain application gets classified as malware or an image – as a deepfake, the SHAP, implemented in the system actually gets involved to explain what features have impacted the decision. In case of malware detection, this might be permissions or API-related attributes that have influenced the classification. As for the image classification, the regions of the image that impacted the decision are highlighted so that one can actually see and understand what the classifier looks at when it classifies an image as a deepfake. The ability to test attacks using FGSM and PGD techniques is another important part of the system. Those techniques implement attack modifications to the input and then observe how the model reacts to those modifications. Those two approaches allow the user to observe whether a slight modification of the input will have an impact on the accuracy of classification or the confidence level of the model. The proposed SECURE-XED system also features an interactive simulator helping students see the results of the model, understand its decisions, observe the attacks and see how the model's decisions change. This simulator aims to support the student's practical approach to the system and help raise the awareness of the powers and limitations of artificial intelligence [48, 49]. By interacting with the output of the prediction and explanation systems, the students can scrutinize the reasons for the decisions and the impact of some adversarial modifications on those decisions. The system also tries to be accessible for the low-income groups as an additional tool for learning artificial intelligence; however, its benefits and actual educational potential need further evaluation.

The review of the previous studies shows that there is much evidence for the successful application of CNNs for malware detection and the application of SOA and ICA for model optimization. However, much of this research does not consider attack resistance, decision explanation, and computational costs in the system. Those limitations become apparent when the necessity arises for the system to work under various conditions or make decisions based on modified or manipulated input. A model that shows high accuracy under standard test conditions might exhibit very poor results under attack, and a model that is computationally heavy might not be applicable for the required environment. This is what we aim to address with the proposed SECURE-XED system. Instead of focusing on improving only one aspect of the model, we intend to balance accuracy, speed, and attack resistance. By adding the features of SHAP explanation and FGSM/PGD testing and optimizing the system using multi-objective approaches, the proposed system also aims to make the model transparent and its limitations apparent. Our aim, then, is not only to improve the quality of prediction but to examine how the accuracy of prediction, transparency, robustness, and computational efficiency can be obtained in a single system.

Development of deepfake detection moves quickly because benchmark datasets like FaceForensics++ and Celeb-DF v2 are available [23,24]. Sophisticated CNN-based architectures are created by individuals who study data, and these models achieve elevated accuracy on many benchmarks [7,9,25]. Recent work has also examined audio-visual feature fusion for deepfake detection [13]; however, the evaluated SECURE-XED deepfake pipeline is restricted to visual frames. These systems act like a black-box because the systems provide limited insight into model reasoning. This limitation makes the systems hard to use in high-stakes or educational institutions. Even though explainability methodological paths like SHAP provide feature-level interpretation, the integration of SHAP into real-time detection pipelines is not common [10,11,26-28]. Most systems treat explainability as a step after the analysis instead of a part of the inference process. Recent work has also proposed multidimensional metrics for evaluating explainability and model trustworthiness [12].

While SHAP has notable strengths in terms of explainability, there are a few drawbacks when applied to Convolutional Neural Network (CNN)-based detection of malware. The exact calculation of Shapley values is combinatorial; therefore, any practical implementation must use approximations. The generation of explanations requires additional computation, and the Shapley values themselves can be sensitive to the type of approximation used, the background data that is sampled, and the way that malware features are represented by the CNN. These are particularly concerning for applications that need to provide explanations in a real-time and interactive way. SECURE-XED does not consider SHAP to be computationally free and does not claim that it has none of its approximation limitations. Instead, SHAP has been implemented as an inference-time module of SECURE-XED, a system that explicitly assesses the latency of its explanations and the efficiency of the network. SHAP outputs have also been assessed via a simulator for educational purposes.

In the last few years, research has been published that suggests that robustness and interpretability are not always independent of each other. One such study by Tan and Tian published in 2020 demonstrated that increasing the robustness of explanations could lead to a trade-off between the robustness and faithfulness of explanations when the original network itself is non-robust [8]. Another study by Crabbé and van der Schaar published in 2021 suggested that explanations of networks should be evaluated regarding their invariance and equivariance to the network itself, demonstrating that some explanations of models may not be stable when transformations are applied to the input image [50]. More recently, studies in constrained explainability have highlighted the tension between models' predictive power, interpretability, and cost [51]. These publications motivate SECURE-XED to evaluate the adversarial behavior of the system itself and the explanations that are produced by the system in the same way. In a related area, research into adversarial deepfakes has suggested that the robust detection of adversarial images requires specific defenses for those adversarially manipulated images [52].

Media-literacy initiatives, like the MIT Media Lab's "Detect Fakes" and the MIT RAISE/Day of AI curricula, give importance to critical reasoning and accessibility in educational institutions [29,30]. Many tools focus on quizzes or specific areas, which means these tools have little interaction with the models. These tools also lack ways to study system behavior under adversarial conditions. Simulation-based learning environments work well in many fields, even though few systems combine explainability and adversarial robustness in one framework [31].

SECURE-XED solves these problems by combining Android malware analysis and deepfake detection in a shared CNN backbone with two task-specific heads. Cross-domain learning and efficiency are made possible by this design. Multi-objective metaheuristic optimization (SOA + ICA) is used by SECURE-XED because SECURE-XED needs to improve accuracy, adversarial robustness, and computational efficiency. SECURE-XED puts explainability into the inference loop using SHAP-based visualizations. An interactive, model-in-the-loop simulator is introduced by SECURE-XED to support adversarial testing on real data. Practical cybersecurity uses and AI/media-literacy academic investigation are helped by this design.

Existing approaches are increasingly focusing on combining detection models with optimization, explainability, or adversarial evaluation, but components are often considered and evaluated in isolation. There are three aspects which limit the scope of the existing work and motivate our study. Firstly, heterogeneous security and multimedia data often involve domain-specific adaptations of representations, but there is limited work on analyzing if such adaptations could be unified under a shared representation learning paradigm, without enforcing similar distribution of raw features. Secondly, optimization often focuses on improving prediction performance, while robustness, calibration, and efficiency are often considered as independent evaluation criteria, rather than jointly optimized, conflicting objectives. Thirdly, explainability and adversarial auditing are rarely linked to a model-in-the-loop educational scenario where students are interacting with the model's predictions and perturbations. The method proposed in this research, which we have named SECURE-XED, addresses the ability to overcome a set of challenges: (i) the ability to adapt representations associated with inputs from the fields of security and multimedia, and at the same time the ability to learn through the shared latent space, (ii) the ability to formulate a proposed optimization problem with multiple objectives explicitly and, most importantly, in an accurate way, as well as presenting and maintaining the robustness of PGD and ECE, with regard to the possibility and efficiency of using a new SOA-ICA formulation, with regard to the possibility of formulating a new SOA-ICA, it is available and efficient, (iii) we were able to develop an educational simulator capable and qualified to directly use the outputs of the proposed model, explain SHAP, and also illustrate the adversarial examples.

Table 1. Limitations of Existing Methods and the Entry Point of SECURE-XED.

Research stream	Existing capability	Main limitation	SECURE-XED entry point
CNN-based Android malware detection [4,16,17]	Effective learning from static and behavioral features	Primarily domain-specific; dynamic analysis increases computational overhead	Shared backbone with optional selectively activated behavioral evidence
Metaheuristic security optimization [18-22]	Automated feature and hyperparameter search	Often emphasizes predictive performance without jointly considering robustness, calibration, and efficiency	SOA+ICA multi-objective optimization over clean performance, PGD robustness, ECE, and efficiency
CNN-based deepfake detection [7,9,23-25]	Strong benchmark detection performance	Limited transparency and sensitivity to distribution/adversarial changes	Shared adversarial-aware backbone with a task-specific deepfake head
SHAP/XAI methods [10,11,26-28]	Feature- and region-level attribution	Approximation dependence and additional computational overhead; often used post hoc	Inference-time SHAP integration with explicit latency evaluation and simulator-based human assessment
Adversarial evaluation [11,34,52]	Exposes model sensitivity to perturbations	Frequently evaluated separately from explainability and user interpretation	FGSM/PGD evaluation coupled with SHAP-based interpretation
AI/media-literacy tools [29-31]	Accessible learning and awareness environments	Often interface/quiz oriented and weakly connected to operational models	Model-in-the-loop simulator using actual predictions, explanations, and adversarial perturbations
SECURE-XED	Shared cross-domain representation and integrated evaluation	Does not eliminate domain-specific preprocessing or SHAP approximation dependence	Integrates domain adapters, shared CNN, SOA+ICA optimization, adversarial evaluation, XAI, and educational interaction

3. SECURE-XED System Architecture

SECURE-XED offers a unified, interpretable framework for Android capable of detecting malware and analyzing deepfakes. It utilizes a common convolutional neural network architecture with a hybrid optimization capability, leveraging metaheuristic optimization (SOA + ICA). Interpretations are delivered using SHAP, and the proposed system features adversarial evaluation based on FGSM/PGD. The system also boasts optional dynamic capabilities for malware monitoring and calibration (ECE), enabling robust, interpretable learning and, importantly, cross-domain effectiveness.

3.1. Domain-Specific Data Preprocessing and Input Adaptation

A machine learning model is built to detect malware in Android APK files. Static features such as requested permissions, API call sequences, and manifest metadata—including services, intent filters, and exported components—are extracted. These features are reshaped into binary attributes or frequencies and stored in two-dimensional arrays for

CNN processing. To convey behavioral information when runtime traces are available, a dynamic or behavioral branch has been included as an option in the model. For deepfake detection, the model only needs to be fed video frames as input, which can then be processed using visual encoders (one-shot).

Each domain is routed through input-specific preprocessing heads before entering the shared CNN. When dynamic signals are present, we perform early fusion with the static embedding; otherwise, the model runs static-only.

The malware branch now supports an optional behavioral head that consumes system-call sequences and network-flow summaries. The resulting embedding is fused with the static representation before classification. This preserves the unified backbone design while enabling hybrid static+dynamic inference when runtime signals are available.

3.2. Shared CNN Backbone and Cross-Domain Routing

SECURE-XED employs a unified CNN backbone designed to process both 1D tabular malware features and 2D visual deepfake inputs. Malware data entries are processed as 1D tensors after structured feature vectors are reshaped by the system. The central support structure consists of stacked convolutional layers with ReLU activations and max-pooling operations. Dropout regularization is used at a rate of 0.3 to prevent the model from learning the training data too specifically. An optional BiLSTM layer is added for temporal data entries like video sequences so that sequential patterns can be caught.

Malware features are combined through early fusion because this method ensures that different signals are joined before they go into the shared central support structure. Parameter sharing is enabled when deepfake frames are moved directly through the same convolutional trunk. Despite the many fields in which the system can be used, its unified design allows it to acquire knowledge efficiently and maintain flexibility. Architectural redundancy is reduced by this design when deepfake frames are processed.

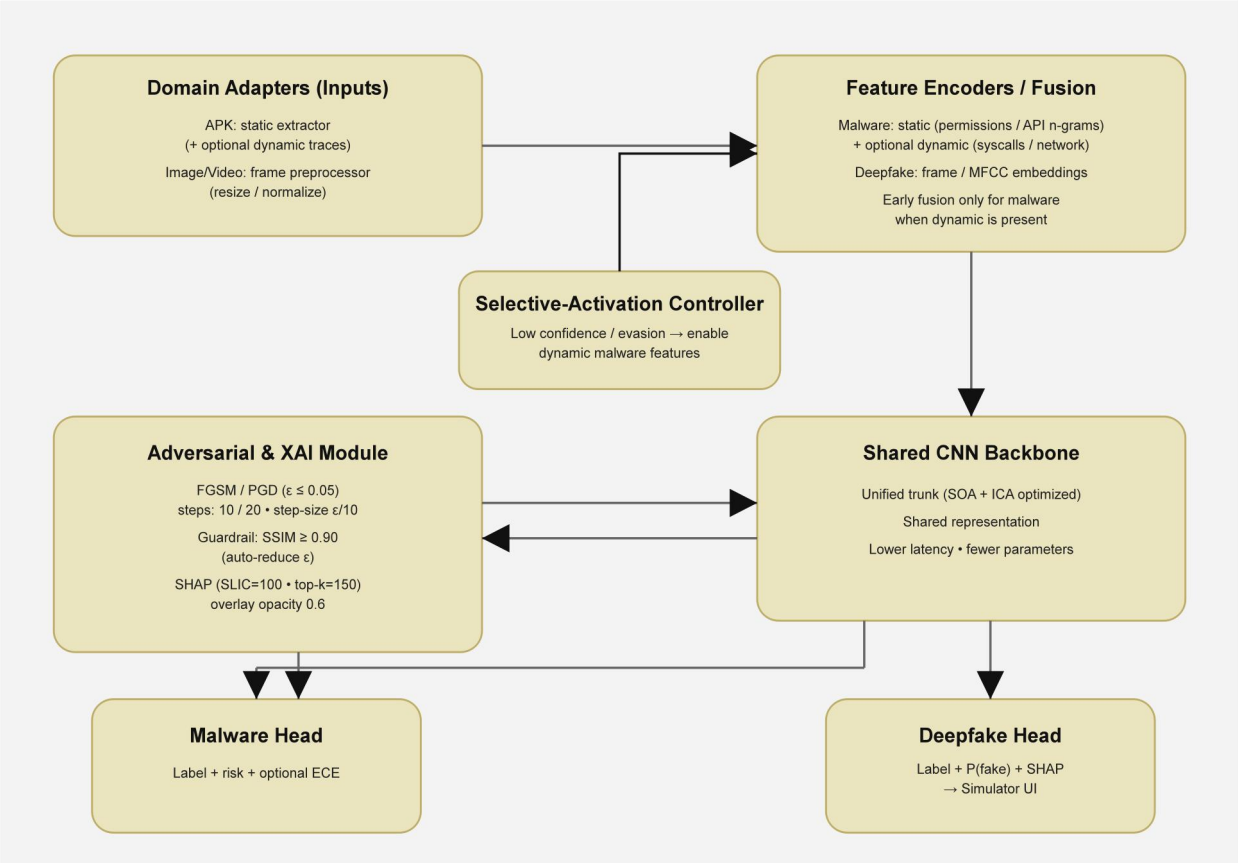


Fig. 1. Modular SECURE-XED architecture with shared backbone and dual-task routing.

The shared backbone is attached to two lightweight task-specific heads. The malware head outputs the malware label, risk estimation, and optional calibration information while the deepfake head outputs the authenticity label, P(fake), and the explanation used to train the simulator. With this design, the domain-specific preprocessing is performed at the input and output boundaries while the CNN trunk learns the shared representation. For malware, the selective-activation controller can be in either STATIC or HYBRID mode. For each sample, it first attempts to classify it in STATIC mode using the static embedding; if the result meets the confidence criteria and no evasion feature was detected, the prediction is outputted. Otherwise, the controller switches to HYBRID mode, activates the behavioral branch, concatenates the

embeddings from both branches, reclassifies the sample using the fused representation, and finally reverts to the STATIC mode for the next sample. The confidence criteria are calculated using the maximum probability across classes: if its value is higher than $\tau = 0.70$, the sample is classified as normal, otherwise, the behavioral branch is triggered. Note that the threshold is different from the one used by the deepfake simulator to make classification decisions. By design, our approach lets the malware classifier maintain the efficiency gains from parameter sharing while avoiding the need to make assumptions about the joint distribution of features and semantics of malware and deepfake.

Our contribution is architectural and systemic rather than layer-level: a single, metaheuristic-optimized backbone with dual heads serves Android malware and deepfake forensics, integrating SHAP explanations and FGSM/PGD robustness checks in one pipeline and classroom simulator. The SOA/ICA scheme optimizes a multi-objective target—clean accuracy, PGD drop (robustness), calibration (ECE), and efficiency (latency/parameters)—yielding the gains reported in Tables 6, 14, 20 and the Pareto trend in Fig. 15 (See Fig. 3 for the optimizer flow).

3.3. Hybrid SOA–ICA Multi-Objective Optimization

The optimization procedure employs a hybrid metaheuristic approach that combines the global exploration capability of SOA with the refinement capability of ICA. Hyperparameters such as learning rate, kernel size, filter numbers, and batch size are explored by SOA. The ICA algorithm helps improve the SECURE-XED model in two different ways, depending on the type of data it processes. First, ICA helps detect Android malware through feature selection. This means it selects the most useful features for detecting malicious software, such as permissions and API calls. For example, if we have 500 features, ICA can help select a smaller set of the most useful features for classification, rather than using all of them, thus saving time and effort. Second, ICA helps detect deepfakes through lightweight pruning. This means ICA removes some of the less important parts or parameters from the neural network used for image processing. The goal of ICA in this case is to reduce the model's size and computational effort while maintaining detection accuracy. This dual-stage optimization procedure balances exploration and exploitation so that better convergence is reached.

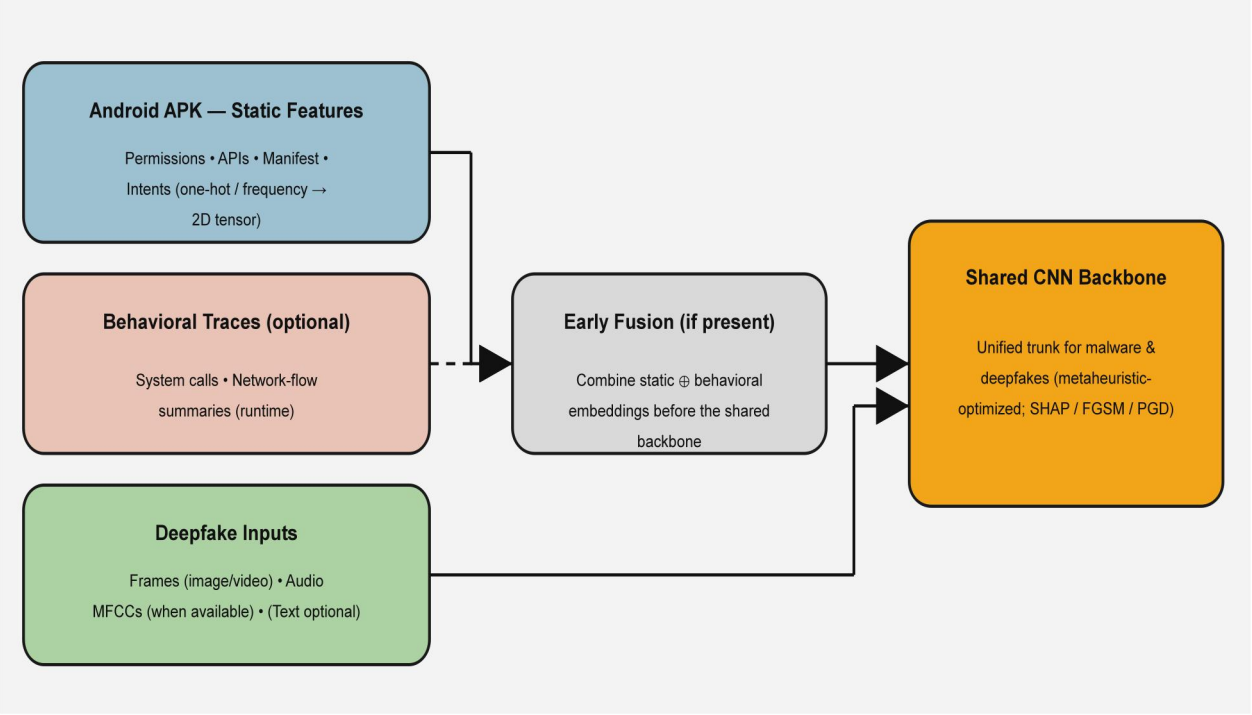


Fig. 2. Input feature extraction and routing. Android static metadata and optional dynamic traces (syscalls/network) are encoded and early fused before entering the shared CNN backbone. The evaluated deepfake path uses visual frames mapped to CNN-ready tensors and routed to the same backbone. Audio/text branches shown in the schematic are conceptual extension paths and are not part of the reported experiments.

The optimization process is guided by a multi-objective, PGD-aware validation function defined as:

$$J = Acc_{clean} - 0.20\,\Delta_{PGD} - 0.05\,ECE + 0.05\,(\frac{1}{Params_{norm}} + \frac{1}{Latency_{norm}}) \quad (1)$$

where Acc_{clean} denotes validation accuracy on clean samples, Acc_{PGD} denotes validation accuracy under the configured PGD attack, and : $\Delta_{PGD} = Acc_{clean} - Acc_{PGD}$ represents the adversarial accuracy degradation. is the Expected Calibration Error, while and denote the normalized parameter count and inference latency, respectively. The weights were set to λ_{rob}

= 0.20, λ_{ece}= 0.05, and λ_{eff}=0.05. The optimization was additionally constrained by the unified-model parameter budget, latency of approximately 30 ms/frame for the vision branch, and SSIM ≥ 0.90 for adversarial visual samples.

Strict rules are placed during the optimization procedure so that the system can be used in real life. Model size is kept within a unified baseline by these rules. Inference latency must stay at approximately 30 ms per frame for vision tasks. Perceptual similarity (SSIM ≥ 0.90) is enforced for adversarial samples to maintain quality. Parameter rules are applied to keep malware data entries compact. Table 2 shows the optimized configurations while Table 3 shows the settings and rules. A favorable Pareto trade-off is shown in Fig. 15.

Table 2. CNN Hyperparameters Optimized via SOA and ICA

Hyperparameter	Search Range	Selected Value
Learning Rate	0.0001 – 0.01	0.001
Dropout Rate	0.1 – 0.5	0.3
Kernel Size	3 – 7	5
Number of Filters	32 – 128	64
Batch Size	16 – 128	64
Conv Layers	3 – 6	4

Table 3. Objective & Constraints.

Item	Setting (vision)	Setting (tabular/malware)
Validation objective	Objective in (1)	Same objective as (1)
PGD setting (validation)	ε = 0.03, 10 steps, step-size ε/10	ε = 0.1, 40 steps, step-size ε/10
Constraints	Params ≤ unified baseline; Latency ≤ ~30 ms/frame; SSIM ≥ 0.90	Params ≤ unified baseline

The methodological contribution does not rely on modifying the internal update equations of SOA or ICA. Instead, SECURE-XED customizes their orchestration for constrained CNN configuration search. SOA's role in the system is to explore architectural and training variants, while ICA specializes in optimizing certain domain-dependent feature subsets for malware and also performs the additional task of optimizing light trimming configurations for vision inputs in the system. Candidate configurations are evaluated through the shared multi-objective fitness function in (1), which jointly accounts for clean accuracy, PGD-induced degradation, calibration error, and computational efficiency under explicit parameter, latency, and SSIM constraints. Considering the discrete variables in the system that we evaluated in the ablation study, which were based primarily on SOA only, ICA only, and the combined SOA+ICA, we will find that these variables are able to better illustrate the contribution of each stage of optimization in the system.

3.4. Inference, Explainability, and Adversarial Evaluation

At inference time, SECURE-XED makes use of the task-specific prediction, selective behavioral activation for malware, explainability, and adversarial evaluation, which are incorporated into a modular inference head. By default, an input passes through its domain-specific adapter and the shared backbone to the task-specific heads. For malware inputs, if low confidence is detected or an evasion attempt is suspected, the optional behavioral branch is activated. In parallel, the SHAP explanation is generated for user inspection, and the FGSM/PGD probes are applied to assess the robustness of the model and optionally engage the simulator in the challenge/response mode.

SECURE-XED utilizes the SHAP [27] library as a clarity of reasoning module for both malware detection and deepfake tasks. Note that the model provides real-time explanations due to being naturally integrated into the inference pipeline. As mentioned earlier, our model offers the ability to provide human oversight and interpretation during its operation, unlike the interpretation tools of other systems which are applied only post-hoc. Such an architecture choice also increases the overall transparency and interpretability of the model due to a specific type of feature interaction, which occurs during the model inference.

SHAP identifies the features that have the most power over classification decisions in the malware area. Critical permissions and API calls are highlighted by SHAP so that cybersecurity analysis professionals can verify the results. These highlights help analysts see if model predictions match known malicious patterns. Spurious correlations are detected more easily when SHAP provides these actionable insights. Support for both analysis and debugging is given by SHAP at the individual prediction level.

Spatial heatmaps are created by SHAP for deepfake detection to show the parts of the input frame that the model used for the label. Manipulated facial regions and texture inconsistencies are typically shown in these visual explanations. These region-based attributions let system operators look at the reasoning of the model. Trust is improved in high-stakes situations when these heatmaps are provided.

Consistent feedback is provided by the combination of SHAP across both areas for cybersecurity analysis professionals and individuals in educational institutions. The simulator component of SECURE-XED is supported by this functionality so that system operators can explore predictions. The system bridges the gap between black-box

performance and human-understandable decision-making. Usability and educational value are enhanced because the internal reasoning of the model is exposed to the system operators.

3.5. Adversarial Robustness Module

To validate robustness and simulate attacks, we implement FGSM [34]. Table 4 provides representative FGSM results for Drebin and FaceForensics++, while the complete dataset-level robustness results are reported in Sections 4.7 and 5.4.

- Malware: Small perturbations are applied to feature vectors to mimic evasive malware
- Deepfakes: Visual inputs are subtly manipulated to test detector stability

This module is also integrated into the simulator's "challenge mode."

Table 4. Accuracy Before and After FGSM Perturbation.

Domain	Accuracy (Clean)	Accuracy (FGSM Perturbed)	Drop (%)
Malware	96.4%	84.2%	12.2%
Deepfake (FF++)	94.7%	86.3%	8.4%

Fig. 3 illustrates the differentiated roles of SOA and ICA within the two-stage optimization procedure: SOA explores CNN architecture and training configurations, whereas ICA refines feature-selection and pruning decisions before Pareto-based configuration selection.

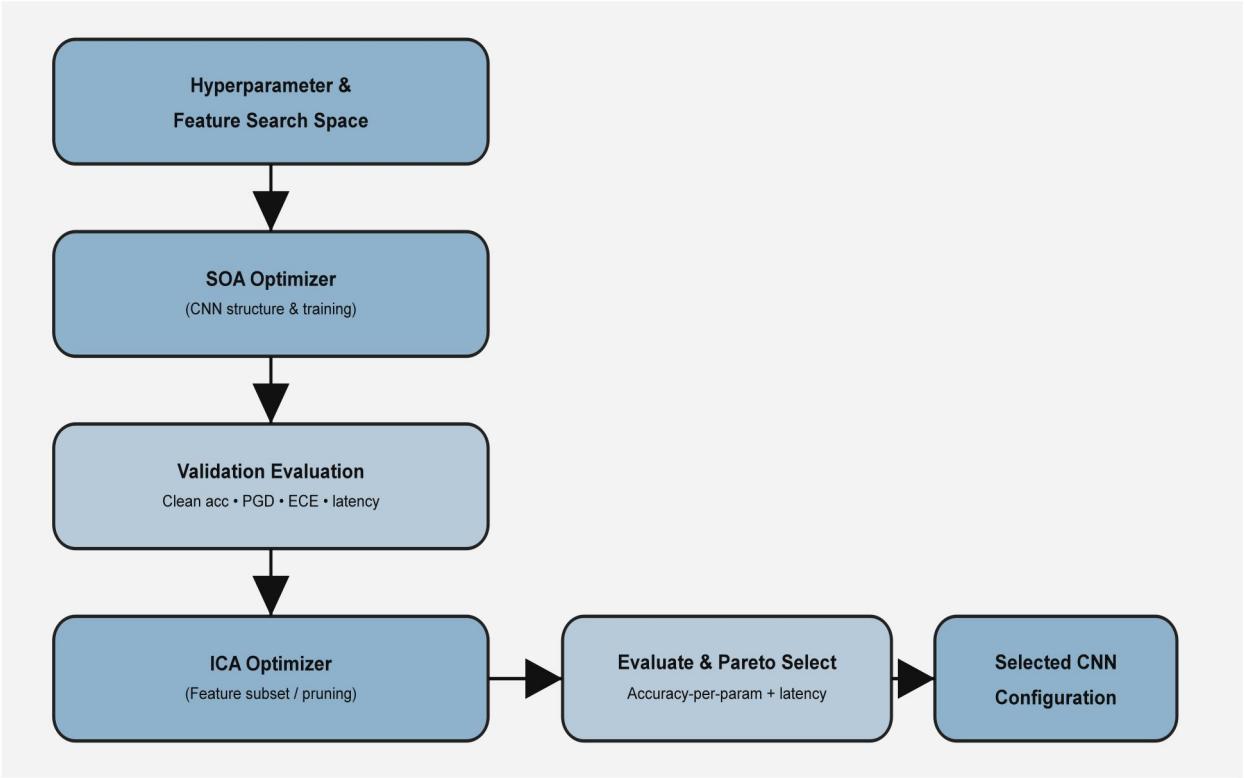


Fig. 3. Hybrid metaheuristic optimization in SECURE-XED. SOA proposes CNN/HP candidates; we evaluate on a PGD-aware, efficiency-aware objective (clean accuracy, PGD drop, ECE, latency/params). ICA refines feature subsets (malware) and mild channel/pruning (vision), followed by Pareto selection to meet accuracy-per-parameter and latency targets (see Fig. 15).

4. Malware Detection Experiments

4.1. Dataset Description

Three public Android malware datasets, including Drebin [35] with 5,560 apps (static features such as permissions and APIs), CICAndMal2017 [36] containing 4,000 apps (a combination of static features and dynamic traces), and AMD [37] with over 24,000 samples from 71 malware families for multiclass analysis, were used to assess generalisability.

We extracted consistent features from all sources, including:

- Static: permissions, API call frequencies, manifest elements
- Dynamic (if available): system call traces, network metadata.
- Any available obfuscation patterns and module displays were also included.

After that, all of the samples were unified into vectors, which were then shaped into 2-D matrices.

4.2. Experimental Setup

The experiments were conducted in a Colab Pro+ environment with:

- Hardware: Tesla T4 GPU, 32GB RAM
- Frameworks: TensorFlow 2.11, SHAP 0.41, Python 3.10

The CNN configuration (optimized via SOA and ICA):

- 4 convolutional layers, ReLU activations, kernel size = 5
- Filters: 64, 128; dropout = 0.3
- Fully connected layer with softmax output
- Optimizer: Adam, LR = 0.001 (from SOA)
- Batch size: 64, Epochs = 50

Metaheuristics:

- SOA: 25 iterations, 10 agents
- ICA: 8 empires, 20 colonies

The SOA search was conducted over the bounded hyperparameter space reported in Table 2: learning rate [0.0001, 0.01], dropout rate [0.1, 0.5], kernel size [3,7], number of filters [32,128], batch size [16,128], and convolutional depth [3,6]. SOA used 10 agents over 25 iterations, whereas ICA used 8 empires and 20 colonies to refine the candidate feature/pruning configurations. Each candidate configuration was evaluated using the multi-objective validation function in (1) on the validation partition before the corresponding population update.

4.3. Dynamic Behavioral Features

To complement static APK features, we add a dynamic input head based on runtime system call sequences and network-flow summaries (available for CICAndMal2017; Drebin/AMD remain static-only). System calls are captured as sequences and tokenized into unigrams/bigrams with temporal windows of 200 events (stride 100). Network flows are summarized per app execution (packet counts/bytes, TCP flags, DNS ratio, mean/variance of inter-arrival times).

Dynamic inputs pass through a 1D CNN → BiLSTM stack to produce a fixed-length behavioral embedding. The static branch independently encodes the unified malware representation derived from permissions, API-call frequencies, manifest elements, and other available static indicators. For hybrid malware inference, early fusion is performed after branch-specific encoding and before the classifier head. Let the static and dynamic behavioral embeddings be defined and concatenated along the feature dimension as follows:

$$z_s \in \mathbb{R}^{d_s}, \quad z_d \in \mathbb{R}^{d_d}, \quad z_f = [z_s \parallel z_d], \quad z_f \in \mathbb{R}^{d_s+d_d} \tag{2}$$

The fused features flow through to the shared classifier and malware-specific head. Dynamic fusion is activated only when requested by the selective-activation controller. Otherwise, inference uses static features only. Drebin and AMD are excluded from the fusion ablation since, in this study, they provide only static features. Training follows SECURE-XED's optimizer and class-weighting settings, while the validation and test distributions remain as they are.

All models are evaluated in identical splits to permit paired testing. For malware, we use repeated stratified 5×2 cross-validation at the application / sample level with family-aware stratification. For deepfakes, we use 5-fold subject/video-level CV (clips from the same video never cross-folds). For each metric, we report fold means and compute 95% confidence intervals using stratified bootstrap (10,000 resamples), resampling apps for malware and videos for deepfakes. Paired differences are tested with Wilcoxon signed-rank (Accuracy/F1) and DeLong's test for AUC; McNemar's test is used for paired proportions (e.g., TPR@FPR). We correct for multiple comparisons using Holm–Bonferroni. The effect size is Cliff's δ with the usual interpretation (negligible <0.147, small 0.147–0.33, medium 0.33–0.474, large 0.474). The random seed was fixed at 42 for split generation and model training across paired comparisons, and the same folds were retained across models to support reproducibility.

4.4. Handling Class Imbalance.

To address label imbalance (e.g., in Celeb-DF), we used class-weighted cross-entropy during training with weights using the following equation:

$$w_c = \frac{N}{2N_c} \tag{3}$$

where:

N is the total number of training samples,
Nc is the number in class c.

We also formed balanced mini-batches via stratified sampling (≈ 1:1 real/fake at the batch level) to stabilize gradients. The validation and test sets preserved the original distributions to reflect the prevalence in the real world. We selected the operating threshold by maximizing macro-F1 on a held-out validation split; in addition, we report threshold-independent metrics (ROC-AUC, PR-AUC) to avoid threshold bias under imbalance. We also evaluated focal loss (α=0.25, γ=2) as an alternative; see Table 17.

4.5. Baseline Models for Comparison

To compare SECURE-XED with established baselines, we implemented the models shown in Table 5. These models were tested on the Drebin and CICAndMal datasets using the same preprocessing pipeline as SECURE-XED, as shown in Table 6. These results show that SECURE-XED achieves higher accuracy than the evaluated baselines across the reported datasets, including AMD, where family variation is high.

Table 5. Baseline Models.

Model ID	Model Type	Key Settings
B1	Naive Bayes	Multinomial, alpha = 1.0
B2	Random Forest	n = 100, depth = 20
B3	XGBoost	max_depth = 6, LR = 0.3
B4	Basic CNN	3 layers, default settings
B5	LSTM (Permissions)	64 hidden units

4.6. Explainability with SHAP

SHAP also identifies influential features for malware, consisting of permissions and API calls. This support for interpretability and validation for analysts is shown in Figure 4. SECURE-XED achieves a maximum accuracy of 96.4%, surpassing the evaluated baselines (Table 6). Ablation results demonstrate that combining the advantages of the Seagull Optimization Algorithm (SOA) and Imperialist Competitive Algorithm (ICA) outperforms either optimization potential alone (Table 7). Table 8 shows that using fused static and dynamic features further improves detection performance, proving that behavioral evidence is essential.

Table 6. Accuracy Comparison with Baseline Models.

Model	Drebin (%)	CICAndMal2017 (%)	AMD (%)
Naive Bayes (B1)	84.1	81.3	75.2
Random Forest (B2)	89.3	87.5	81.6
XGBoost (B3)	91.5	88.9	84.3
Basic CNN (B4)	93.2	91.8	86.7
LSTM (B5)	92.0	89.4	85.1
SECURE-XED	96.4	94.7	91.2

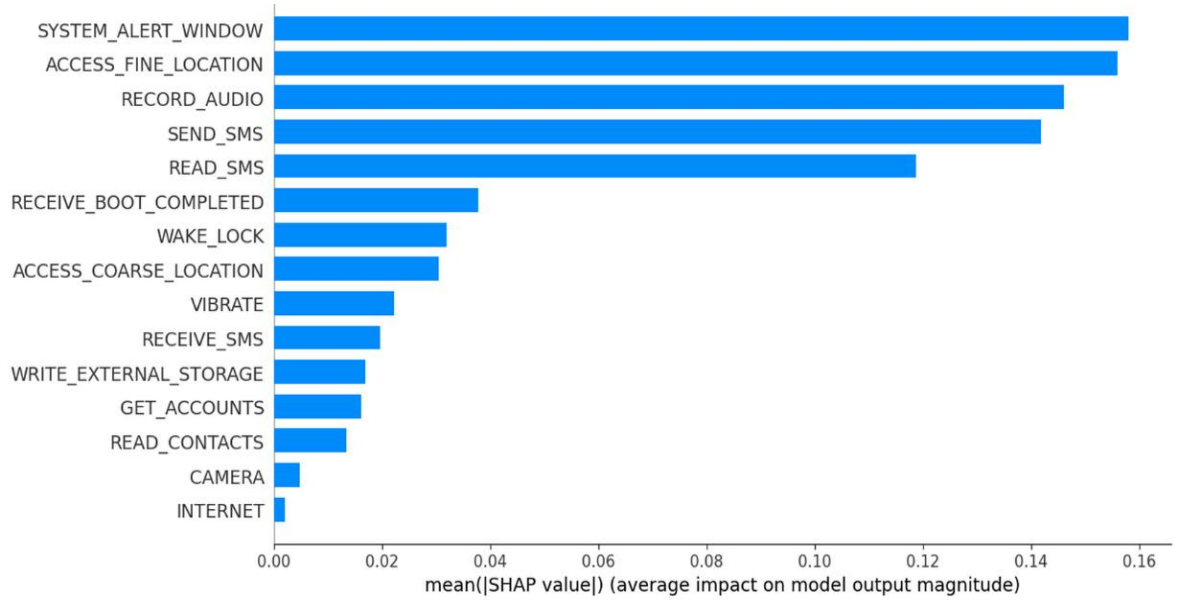


Fig. 4. Importance of SHAP features in Malware Classification.

Table 7. Metaheuristic ablation (malware): impact of SOA and ICA on accuracy (%).

Variant (Malware)	Drebin	CICAndMal2017	AMD
Vanilla CNN (no metaheuristics)	95.3	93.2	89.9
SOA only	95.9	94.0	90.5
ICA only (feat. subset + depth)	96.0	94.1	90.7
SOA + ICA (ours)	96.4	94.7	91.2

Table 8. Static vs. dynamic vs. hybrid features on CICAndMal2017.

Variant	Accuracy (%)	F1	ROC-AUC	TPR@1%FPR
Static only (SECURE-XED)	94.7	0.947	0.970	0.90
Dynamic only	93.8	0.939	0.967	0.88
Hybrid (Static + Dynamic)	95.6	0.955	0.982	0.93

In Table 9, the robustness evaluation of packed/obfuscated samples shows that adding dynamic features overcomes the limitations of static-only models. The AUC and detection rate at low false-positive rates improve.

Table 9. Packed/Obfuscated subset (CICAndMal2017) - Robustness under evasion. (Samples flagged as packed/obfuscated by dataset metadata/heuristics.)

Variant	Accuracy (%)	F1	ROC-AUC	TPR@1%FPR
Static only (SECURE-XED)	92.1	0.922	0.956	0.84
Hybrid (Static + Dynamic)	93.9	0.938	0.971	0.89

To manage computational cost, dynamic analysis is selectively activated under low-confidence or suspected evasion scenarios.

4.7. Adversarial Robustness

We evaluate robustness under FGSM perturbations ($\varepsilon = 0.15$) on 500-sample subsets per dataset. As shown in Table 10, accuracy drops by ~12–13%, yet the model remains functional, indicating resilience and potential for adversarial retraining.

Table 10. Robustness under FGSM attack ($\varepsilon = 0.15$).

Dataset	Clean (%)	FGSM (%)	Drop (%)
Drebin	96.4	84.2	12.2
CICAndMal2017	94.7	81.6	13.1

Dataset	Clean (%)	FGSM (%)	Drop (%)
AMD	91.2	78.4	12.8

Table 11 shows the impact of PGD attacks on SECURE-XED. Accuracy decreases by ~20% across datasets, reflecting the stronger, iterative nature of PGD compared to FGSM. Despite this drop, the model maintains reasonable performance, indicating partial robustness and motivating future adversarial training.

Table 11. Robustness under PGD attack (ε = 0.1, 40 steps).

Dataset	Clean (%)	PGD (%)	Drop (%)
Drebin	96.4	76.8	19.6
CICAndMal2017	94.7	74.2	20.5
AMD	91.2	70.3	20.9

5. Deepfake Detection Experiments

5.1. Datasets Used

We used the following three datasets for our deepfake detection experiments:

- The abnormality of the video makes up only about five percent of the video. The DFDC (Deepfake Detection Challenge) is the largest of all the datasets, with more than 100,000 video clips that were manipulated in some way.
- So, FaceForensics++ is a dataset of photos and videos manipulated using deepfake techniques like DeepFake, FaceSwap, Face2Face, and Neural Textures. We have used the c23 version of the FaceForensics++ videos, which includes 1,000 real videos and 1,000 fake videos, similar to the datasets used in the literature [24].
- Celeb-DF v2: The Celeb-DF dataset contains some of the most visually realistic deepfakes, with few artifacts. To make the dataset more balanced for our educational purposes, we sampled the videos to include 590 real videos and 5,639 fake videos [23].

5.2. Feature Extraction, Pipeline, and Experimental Setup

For each video in each dataset, we sampled five frames and resized them to a resolution of 224x224 pixels. We then used several pretrained convolutional neural networks (CNNs) to extract features from these sampled images. These pretrained encoders included EfficientNet-B0, VGG16, and ResNet50, which were used as frozen feature extractors in our model. Following the feature extraction models, we used a combination of convolutional and bidirectional long short-term memory (BiLSTM) layers to analyze the extracted features and classify each frame as either real or fake.

Our experiments were performed using the Tesla T4 GPU. We used TensorFlow 2.11 for the development and training of our models, and utilized several standard settings in our experiments, including training for 30 epochs, a batch size of 32, and the Adam optimizer. We measured the performance of each of our models using accuracy and F1 scores. We also evaluated their adversarial robustness. The results of these experiments and a list of the models used are summarized in Table 12.

Table 13 shows that SECURE-XED consistently outperforms the evaluated baselines across the datasets. The gains are also observed on Celeb-DF and DFDC under the evaluated protocol. Data from the efficiency analysis in Table 14 shows that the unified backbone achieves similar or better accuracy. The latency values in Table 14 measure model inference under clean-input conditions and exclude post-prediction SHAP explanation generation and adversarial attack processing; therefore, they should not be interpreted as end-to-end latency for the complete explainability and adversarial-evaluation pipeline. The parameters were reduced by approximately 2× while delay decreased by around 21%. Robustness under class imbalance is confirmed by per-class and imbalance-aware metrics in Tables 15–17 and Fig. 5. Macro-F1, ROC/PR-AUC, and minority-class recall were improved using appropriate loss functions. These results validate both the effectiveness and efficiency of the proposed unified framework.

Table 12. Deepfake Detection Models Used for Evaluation and Comparison.

ID	Model	Description
D1	XceptionNet	FF++ benchmark baseline [24]
D2	MesoNet	Compact and interpretable
D3	VGG16-Finetuned	ImageNet-pretrained + BiLSTM
D4	ResNet50 + BiLSTM	Temporal pattern extraction
D5	SECURE-XED	EfficientNet + CNN + SHAP + FGSM

Table 13. Deepfake Detection Accuracy across Datasets.

Model	FF++ (%)	Celeb-DF (%)	DFDC (%)
Xception	92.1	87.3	83.8
MesoNet	88.5	81.2	78.5
VGG16	90.2	84.1	81.9
ResNet+LSTM	91.4	85.5	82.7
SECURE-XED	94.7	89.6	86.2

Table 14. Unified vs. separate models (deepfake): accuracy and efficiency comparison.

Configuration	FF++	Celeb-DF	DFDC	Total Params (relative)	Avg. Inference Latency (ms/frame)
Separate per domain (2× backbones)	94.5	89.4	86.0	2.00×	34.1
Unified SECURE-XED (single backbone)	94.7	89.6	86.2	1.00×	26.9

Note: The latency values in Table 14 represent clean model inference only and exclude SHAP explanation generation and adversarial attack processing. The runtime of the explanation and simulator modules is evaluated separately in Table 27.

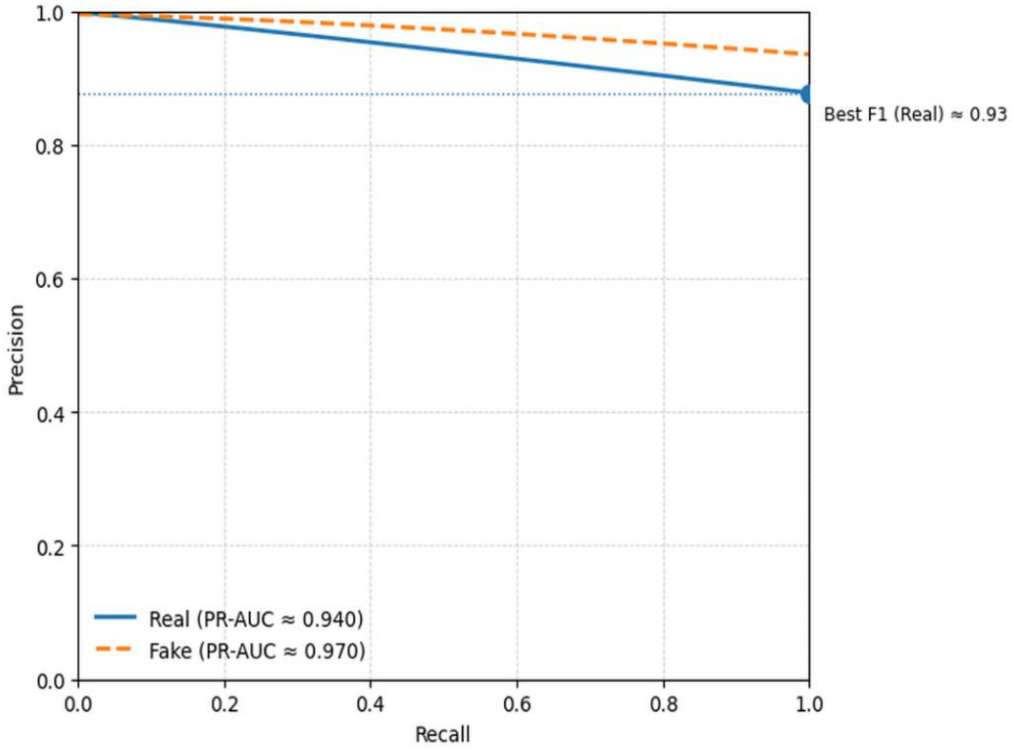


Fig. 5. Precision–Recall Curves for the Real Class on Celeb-DF.

Table 15. Per-class performance under imbalanced test distributions.

Dataset	Real (P/R/F1)	Fake (P/R/F1)	Macro-F1	Balanced Acc.
FF++	95.1 / 94.2 / 94.6	94.3 / 95.1 / 94.7	94.7	94.7
Celeb-DF	88.0 / 84.0 / 86.0	92.3 / 94.1 / 93.2	89.6	89.1
DFDC	85.2 / 83.1 / 84.1	88.7 / 89.4 / 89.0	86.6	86.2

Table 16. ROC-AUC and PR-AUC under class imbalance. (Macro values and class-wise PR-AUC; higher is better.)

Dataset	ROC-AUC (macro)	PR-AUC (Real)	PR-AUC (Fake)	PR-AUC (macro)
FF++	0.98	0.98	0.98	0.98
Celeb-DF	0.96	0.94	0.97	0.955
DFDC	0.95	0.92	0.95	0.935

Table 17. Loss ablation in Celeb-DF (imbalanced). (Macro-F1, Weighted-F1, and minority-class recall for the Real class.)

Loss	Macro-F1	Weighted-F1	Recall (Real)
Cross-Entropy	0.881	0.915	0.81
Class-Weighted CE	0.895	0.922	0.84
Focal (α=0.25, γ=2)	0.898	0.924	0.85

5.3. Explainability (SHAP)

In this section, we will explain how SECURE-XED uses SHAP technology to interpret deepfake detection decisions. SHAP is used for this purpose because our goal is for the model to be able to classify an image as real or fake and to identify the areas that influenced this decision. For example, specific areas of the image, such as the Mouth/Jawline, might be identified where traces of facial manipulation or inconsistent movement can be observed. Another example is the Hairline, where traces of facial blending with the original image can be seen.

But the question here is how to produce the SHAP map in this research. We adopted the division of the image into 100 small regions called Superpixels, and this is set by SLIC (k = 100). Each of the 100 regions consists of a number of adjacent pixels. After the image division is completed, we move to calculating the importance of the regions using SHAP, as it evaluates the extent of each region's contribution to the classification decision. After that, the interpretation map is displayed over the original image with an opacity of 60%, as we adopted this value in the settings for the purposes of carrying out this research. Overlay (Opacity = 0.6). The student can inspect the image in hand, and compare it with the highlighted SHAP regions.

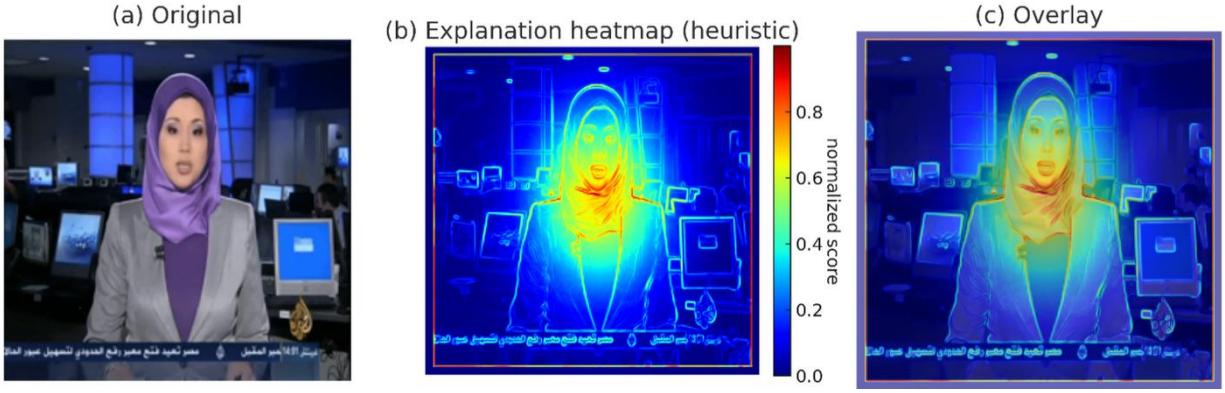


Fig. 6. SHAP explanation on a real deepfake frame (FaceForensics++). (a) Original frame. (b) SHAP heatmap for the fake class (colorbar shows normalized attribution). (c) Overlay of SHAP on the frame (opacity 0.6). SECURE-XED: P(fake) = 0.92 in the clean image. SHAP highlights mouth/jawline and hairline editing artifacts. Parameters: SLIC k=100, top-k = 150 superpixels, overlay opacity 0.6. See Tables 24–26 for module I/O and defaults; Algorithm 1 details the execution flow.

5.4. Robustness to Adversarial Attacks

We applied FGSM (ε = 0.1) to test classifier resilience on a 300-sample subset per dataset. Recent work on adversarial diffusion deepfakes further confirms that detector robustness can degrade under imperceptible adversarial perturbations [52].

Table 18. FGSM perturbation results.

Model	Clean (%)	FGSM (%)	Drop (%)	Dataset
SECURE-XED	94.7	86.3	8.4	FF++
SECURE-XED	89.6	79.5	10.1	Celeb-DF
SECURE-XED	86.2	75.8	10.4	DFDC

Note: Under the evaluated FGSM setting, SECURE-XED retains measurable classification performance on DFDC, while the observed degradation motivates continued robustness evaluation and adversarial hardening.

Table 19. PGD Perturbation Results

Model	Clean (%)	PGD (%)	Drop (%)	Dataset
SECURE-XED	94.7	78.1	16.6	FF++
SECURE-XED	89.6	72.3	17.3	Celeb-DF
SECURE-XED	86.2	68.6	17.6	DFDC

Note: The values in Table 19 represent means over CV folds; 95% CIs by bootstrap (10k). Paired tests: clean versus attacked within the model using the Wilcoxon test (Accuracy/F1) and DeLong (AUC); differences are significant at α = 0.05 unless marked ‘n.s.’.

We chose to continue our experiments and increase the robustness of the evaluation by using a PGD attack (ε = 0.03, 10 iterations), as it is much more stable than FGSM because of its optimization. In Table 19, one can see a significant reduction in the accuracy rates across all groups of datasets with SECURE-XED. The behavior and results show that the PGD attack is severe in nature, as previously mentioned.

Table 20. SECURE-XED performance under a stronger PGD configuration (ε = 0.05, 20 iterations). The increased attack strength leads to a higher accuracy degradation compared to previous settings (Table 19), confirming greater threat severity.

Model	Clean (%)	PGD (%)	Drop (%)	Dataset
SECURE-XED	94.7	73.5	21.2	FF++
SECURE-XED	89.6	69.1	20.5	Celeb-DF
SECURE-XED	86.2	66.8	19.4	DFDC

Note: The values in Table 20 represent mean over CV folds; 95% CI by bootstrap (10k). Paired tests: clean vs. attacked within model via Wilcoxon (Accuracy/F1) and DeLong (AUC); Holm–Bonferroni corrected; 'n.s.' denotes non-significant (α = 0.05).

In our tests, designed to measure the upper bound of model vulnerability more accurately, we extended the PGD test using an ε of 0.05 with 20 iterations. As we increased the number of attacks, ε, and the number of iterations, the classifier's performance degraded further. In the perturbative settings, Table 20 shows a 19–21% reduction in performance across datasets compared with clean accuracy. The attack causes a more extreme failure in the classifier's average accuracy at identifying the correct class while preserving perceptual integrity in the convolutive inputs than the attack in Table 19. These outcomes, therefore, identify the need for proposed tactics to enhance attack resistance in the future, such as PGD-based adversarial training or collective defenses. The drop in performance under the clean attack is substantial across all folds (Wilcoxon/DeLong test, $p < 0.05$), which means that the drop in model accuracy after the attack was statistically significant and that the results of the robustness test can be considered stable. The relative order of the models does not change within the confidence intervals, which supports the stability of our conclusions about robustness.

Table 21. Confidence calibration (Expected Calibration Error, %) - Lower ECE is better. Bins = 15. Evaluated on clean and PGD-perturbed sets.

Model / Dataset	FF++ Clean	FF++ PGD	Celeb-DF Clean	Celeb-DF PGD	DFDC Clean	DFDC PGD
Separate per domain	4.2	8.5	4.7	9.2	5.1	9.6
Unified (SOA+ICA)	3.1	6.9	3.6	7.6	4.0	7.9

Table 22. SECURE-XED Simulator Output on Real Deepfake Frames.

Dataset	Input Type	Prediction	Confidence	Adversarial Method	SHAP Overlay
FaceForensics++	Clean	Fake	0.92	None	Yes
FaceForensics++	FGSM	Real	0.81	ε = 0.01	Yes
FaceForensics++	PGD	Real	0.65	ε = 0.03, 10 steps	Yes
Celeb-DF	Clean	Fake	0.89	None	Yes
Celeb-DF	PGD	Real	0.62	ε = 0.05, 20 steps	Yes

Table 21 presents Expected Calibration Error results for the unified SOA+ICA configuration (the one used throughout the evaluation) compared to the separate domain configuration on the clean and PGD-perturbed test sets. The result is slightly better in terms of probability calibration for the unified approach according to the reported metric, which in turn suggests improved reliability in terms of calibration but not overall trustworthiness or applicability in practice.

Table 22, in turn, includes the prediction outputs along with SHAP-overlay availability results produced by the Python/TensorFlow-based simulator pipeline under clean, FGSM- and PGD-based conditions as examples of successful attack execution and explanation generation by the implemented pipeline.

Fig. 7 illustrates an FGSM perturbation that changes the classifier output with minimal visible alteration, providing a concrete failure case for the simulator challenge mode.

The simulator implements adversarial controls, SHAP overlays, and confidence-based classification outputs. Table 22 and Figs. 7–8 show outputs from the Python/TensorFlow pipeline, while Tables 24–26 and Algorithm 1 document module settings, guardrails, and execution flow. SHAP explanations use SLIC superpixels (k = 100) for the later deletion and insertion analyses.



Fig. 7. FGSM-induced misclassification in a FaceForensics++ deepfake frame.

Fig. 7 compares the clean deepfake frame with its FGSM-perturbed counterpart (ε = 0.1). The perturbation changes the classification despite limited visible pixel differences, illustrating the adversarial sensitivity presented to learners in the simulator.

Fig. 8 shows the PGD perturbation of the same deepfake frame (ε = 0.03, 10 iterations), under which the prediction changes from fake (confidence = 0.92) to real (confidence = 0.65). Tables 24–26 specify the module I/O, defaults, and guardrails, and Algorithm 1 summarizes the execution flow. The iterative optimization property of PGD typically makes it more destructive than single-step FGSM in adversarial attacks [34].

The PGD-induced prediction change provides a concrete simulator challenge case and illustrates model sensitivity under the specified attack setting, without implying generalization to all PGD configurations or deepfake detectors.



Fig. 8. PGD-induced misclassification in a FaceForensics++ deepfake frame.

5.5. Learning Modules (For Classroom Use)

Activities have to be designed according to the SECURE-XED deepfake simulator, which is modular and practical and provides training in media literacy, digital forensics and explainable AI, with a focus on synthetic content, behaviours and AI actions. The modules of the SECURE-XED deepfake simulator are organized to help learners think critically about synthetic content and AI behaviours. The SECURE-XED deepfake simulator can support undergraduate and postgraduate students in learning both the theory and practice of machine learning and artificial intelligence, while encouraging active participation with real-world data, models and AI interpretability tools.

Learners directly manipulate and analyze the system because the learning activities use SHAP-based interpretability, adversarial perturbation techniques (e.g., FGSM), and generative media tools. Ethical and technical awareness regarding synthetic media and detection challenges is promoted by these tools. The learning activities help learners understand content while learners challenge the system in a controlled setting.

Information regarding the modules found in the simulator interface is provided by Table 23. These modules improve technical understanding, critical digital literacy, and ethical awareness because these modules are created for media contexts.

Table 23. Deepfake Simulator Learning Modules (see Tables 24–26 for module I/O, defaults, and guardrails; Algorithm 1 details the execution flow.)

Module	Activity	Goal
What is a Deepfake?	Watch, discuss, analyze	Understand synthetic media
Explainable AI	Interpret SHAP maps	Build model transparency skills
Fool the Model	Apply adversarial noise	Discover model limitations

Educators can choose or arrange modules based on program requirements because the modular simulator allows for usage that happens lesson by lesson. Simulation complexity, such as the difficulty of adversarial attacks or visibility of SHAP layers, can be changed by educators. The modular simulator allows for flexible use in different educational settings, which range from introductory media studies to advanced AI ethics or cybersecurity labs.

Interactive instructional parts are put together in a modular sequence by the SECURE-XED deepfake simulator to support usage within the classroom. Student understanding from multimedia creation to XAI interpretation and adversarial robustness is supported by these instructional parts. The SECURE-XED deepfake simulator builds knowledge in a step-by-step manner when students engage with these instructional parts.

Calculation of the SHAP image over SLIC superpixels (k = 100) is performed by the researchers. The researchers render the top k regions (default top-k = 150; configurable range = 10–300; effective top-k = min (requested top-k, N_SLIC)) as a semi-transparent overlay (opacity 0.6) because the researchers want to show specific details. Fig. 6 offers an illustrative image, while quantitative explanation quality is reported in Section 5 (Fig. 12; Tables 32–34).

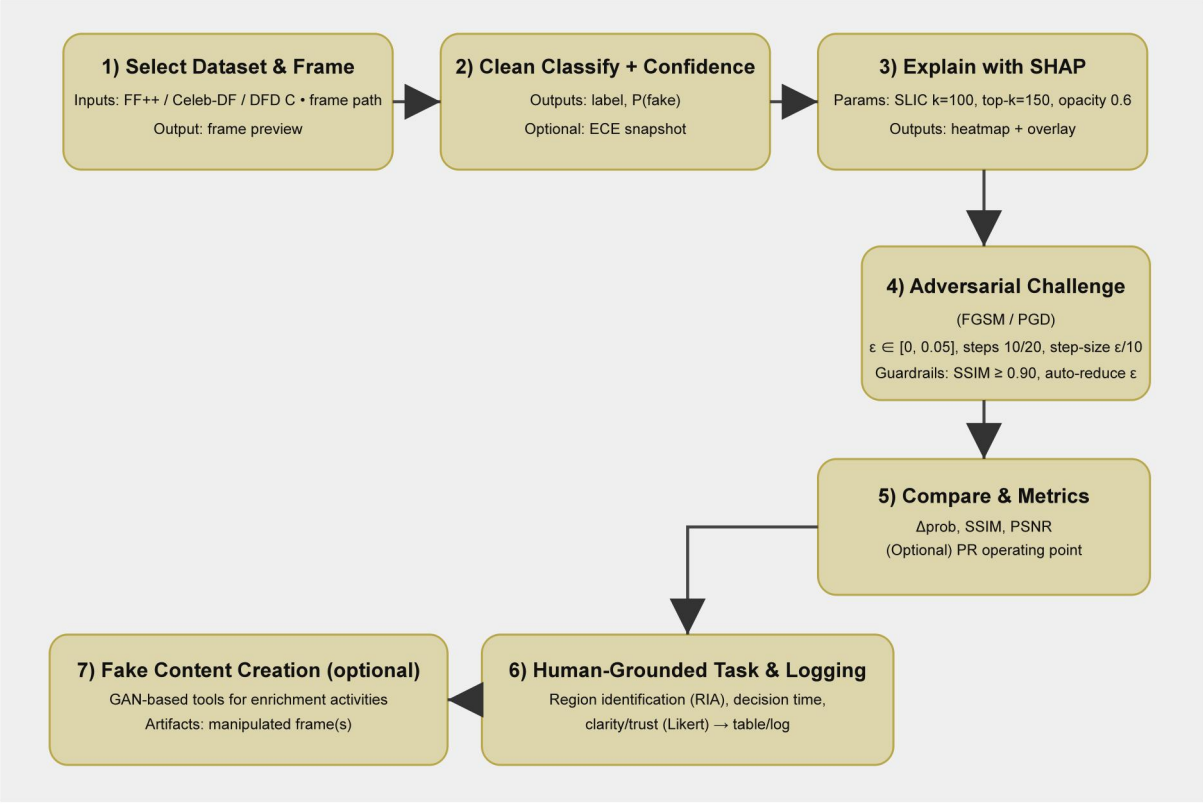


Fig. 9. SECURE-XED classroom flow (deepfake track): (i) select a dataset frame; (ii) clean classification with confidence; (iii) SHAP explanation (SLIC k = 100, top k = 150, opacity 0.6); (iv) adversarial challenge (FGSM/PGD; $\varepsilon \leq 0.05$ with SSIM 0.90 guardrails); (v) compare metrics (Δprob, SSIM/PSNR); (vi) human-based task (RIA, decision time, clarity/trust). The GAN-based Fake Content Creation branch shown in the schematic is an optional pedagogical extension and was not included in the evaluated simulator or reported experiments.

5.6. Module API and I/O (Implementation details)

We expose each simulator module through a thin API with consistent inputs/outputs and parameter bounds that ensure perceptual plausibility and repeatability. Table 24 summarizes the contract; Table 25 lists default settings and valid ranges.

Table 24. Module I/O contract (image/frame path inputs).

Module	Inputs	Key Parameters	Outputs	Logs/Side Effects
Clean Classification	frame_path (H×W×3, uint8)	model_id, threshold (0–1)	label_clean, prob_clean	latency_ms, seed
Explain SHAP	frame_path	topk (10–300), overlay ∈ {heatmap, mask}, opacity (0–1)	overlay_path, rationale_mask, attribution_stats	latency_ms, seed
Fool the Model (FGSM)	frame_path	epsilon (L∞; 0–0.05), clip ∈ [0,1]	adv_path_fgsm, label_adv, prob_adv, Δprob	latency_ms, L∞|δ|, seed
Fool the Model (PGD)	frame_path	epsilon (0–0.05), steps (5–40), step_size (ε/3–ε/10)	adv_path_pgd, label_adv, prob_adv, Δprob	latency_ms, L∞|δ|, seed
Compare Clean vs Perturbed	paths from above	metrics ∈ {SSIM, PSNR}	Δprob, SSIM, PSNR, report_path	latency_ms
Batch Runner	list of frame_path	attack ∈ {none, fgsm, pgd}, overlay ∈ {none, shap}	per-item JSON lines	aggregate_stats.json

Table 25. Default parameters and validated ranges.

Parameter	Default	Range
epsilon (FGSM/PGD, L∞)	0.01 (FGSM), 0.03 (PGD)	0–0.05
steps (PGD)	10	5–40
step_size (PGD)	ε/10	ε/3–ε/10
Top k (SHAP superpixels)	150	10–300
opacity (overlay)	0.6	0.3–0.8
threshold (decision)	argmax	0.5–0.9

PGD was evaluated over 5–40 iterations with step sizes from ε/3 to ε/10, using ε/10 as the default for stable convergence. Stronger configurations were also tested to assess attack sensitivity. These results represent parameter-sensitivity analysis rather than certified robustness guarantees.

Table 26. Guardrails and error handling.

Check/Guardrail	Action
Non-RGB or out-of-range input	Convert/cast to uint8 RGB; normalize to [0,1].
Oversized epsilon (>0.05)	Clamp and warn: "ε capped to 0.05 to preserve perceptual quality".
Very low SSIM (<0.90) after attack	Auto-reduce ε by 20% and re-run; log adjustment.
GPU OOM during SHAP	Return to tiled SHAP; log tile size.
Missing file path	Raise the error with path echo; skip item in batch mode.
Non-determinism	Expose seed; default fixed seed = 42 for classroom demonstrations.

5.7. Execution flow

Algorithm 1. SECURE-XED frame pipeline

Input: Frame x, attack configuration (type, ϵ, steps), SHAP settings
Output: Prediction results, explanations, metrics, and artefacts.

1. Load and preprocess the input frame x(normalized to [0,1]) and prepare it for inference through the shared backbone.
2. If attack ≠ none, generate adversarial x_{adv}(FGSM/PGD) → infer; else infer on x.
3. If enabled, compute SHAP and render overlay.
4. Compute metrics (Δprob; optional SSIM/PSNR).
5. Save outputs (labels, probabilities, artefacts, latency) to JSON.
6. Return the results for the current input and append them to a batch report if batch processing is enabled.

Example (FGSM / PGD):

simulate --frame ffpp/frame_001.png --attack fgsm --epsilon 0.01 ...
simulate --frame ffpp/frame_001.png --attack pgd --epsilon 0.03 --steps 10 ...

Predictions (clean/adv), Δprob, artefact paths, metrics, and delay are included in the outputs. Results correspond to Figures 7–8 and Table 22. These outputs are helpful for evaluation.

5.8. Simulator Usability and Pedagogical Feasibility

An attempt has been made to review both the technical and educational use of the SECURE-XED simulator, with the aim of determining its speed, usability for students, and ability to teach. During an initial assessment, a small number of operational and user tests were carried out to assess the AI Knowledge Recommendations [38–40]. We selected 100 frames from FaceForensics++ and ran them on a Tesla T4 GPU (Table 27), completing the simulation in less than 3 seconds per frame. For the SHAP interpretation tasks, the simulator was successful, with an 83% success rate, while the

adversarial tasks took 2 minutes to complete (Table 28). As summarized in Table 29, the levels of interactivity and program clarity are rated 4.2 and 4.0, respectively. According to Figure 10, the five modules in the simulator and their relationship to these core AI concepts, as well as additional ones, were evaluated for interpretability, critical thinking, adversarial attack awareness, and ethics. However, as shown in Figure 11, the human-selected region and the region revealed by SHAP agree about 78% of the time. This suggests that our simulator is useful in educational settings.

Table 27. Runtime Evaluation of Simulator Modules (100-frame batch using Tesla T4 GPU).

Module	avg. Latency (sec/frame)	Std. Dev	Notes
Clean classification	0.85	0.12	CNN + BiLSTM inference
FGSM perturbation	1.05	0.15	ε = 0.01
PGD perturbation	2.30	0.27	ε = 0.03, 10 steps
SHAP overlay generation	1.80	0.21	Grad-CAM + SHAP hybrid
Combined pipeline output	2.95	0.34	Classification + overlay

Table 28. Results (n=6 graduate students).

Task	Success Rate (%)	Avg. Completion Time (sec)
Toggle adversarial attack (FGSM)	100	45
Interpret SHAP overlay	83	65
Compare clean vs. perturbed frame	91	72

Table 29. Expert Feedback Ratings (n=5 instructors, 1-to-5 Likert scale).

Metric	Mean Score	Std. Dev
Ease of use	3.8	0.45
Clarity of output	4.0	0.32
Engagement potential	4.2	0.37
Pedagogical value	4.1	0.40

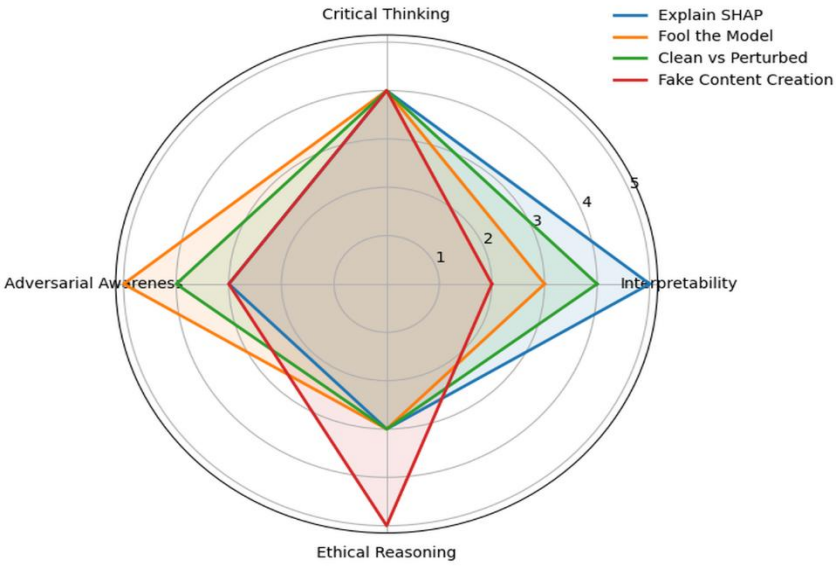


Fig. 10. SECURE-XED Simulator Alignment with AI Literacy Competencies.

5.9. Human-grounded SHAP Validation

The following part of the research aims to understand whether using SHAP helps learners understand SECURE-XED's decisions when choosing the manipulated areas in deepfake images, and whether using the simulator improves their working speed. The experiment involved two comparison cases: one with SHAP interpretation showing the real and fake areas in the simulator image, and another without SHAP.

The first educational experiment (pilot study), based on the initial version of the SECURE-XED simulator, describes how the method was put into operation. Six graduate students (n = 6) were admitted to the experiment and trained to become familiar with the simulator. Then, each participant completed 12 randomized (also balanced across datasets) test cases within a soft time limit of 60 seconds\item. Participants also indicated which area they believed contained the

problem in each test case. We used RIA (Region Identification Accuracy) as a performance indicator to assess participants' success in identifying the manipulated area. A participant's answer was considered correct when the area they identified and the true manipulated area overlapped by at least 30% of the total affected area. a 22-percentage-point increase in RIA, which is a good result for the pilots. More research is needed here. The Likert scale improved by 0.9 points in terms of clarity and confidence, showing that participants are using the simulator more and that their experience has improved. A 12-second decrease in the average decision-making time was also observed. These results suggest that participants in this small-scale experiment achieved greater accuracy in identifying manipulation areas when using SHAP, requiring less time and giving higher clarity and confidence ratings.

Table 31 shows the increase in the success rate from 61% to 83% when participants used SHAP, a decrease in average time from 74 to 62 seconds, and an increase in participants' ratings of clarity and confidence from 3.1 to 4.0 on the five-point Likert scale (1–5 Likert scale). Fig. 11 also indicates a nearly 78% agreement between participants' identifications and the areas highlighted by SHAP. This demonstrates a degree of agreement between the model's interpretation and human understanding.

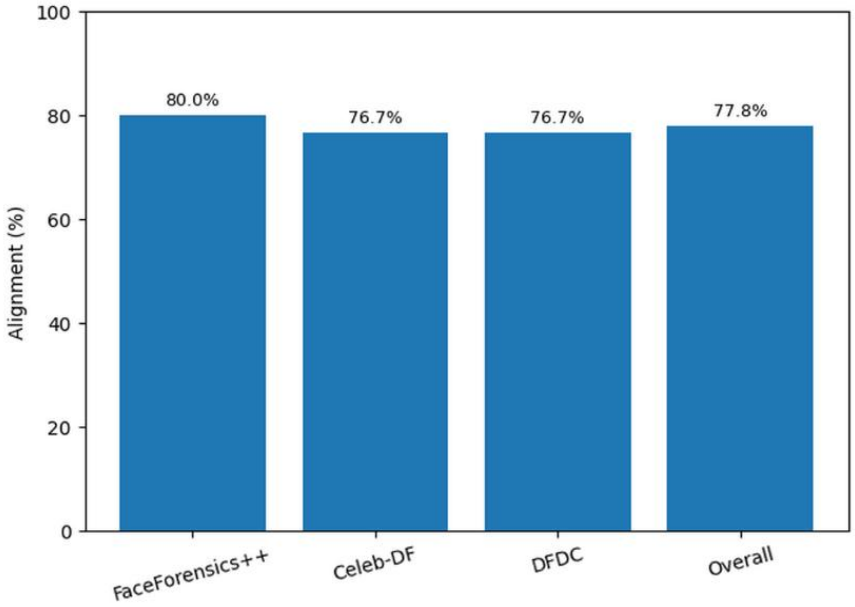


Fig. 11. Human–SHAP Interpretability Alignment (RIA) by dataset.

Table 30. Positioning SECURE-XED relative to AI-literacy and deep-fake-literacy resources.

Resource	Learning Modality	Model-in-the-Loop (Adversarial)	Explainability Overlays	Dataset-Grounded Outputs	Evidence of Learning Outcomes
MIT Media Lab "Detect Fakes"	Public quiz / challenge	No	No	Uses curated examples	Awareness-style engagement; User performance data reported anecdotally. (MIT Media Lab) [29,30].
MIT Media Lab Deepfakes Module	Classroom learning module	No	No	Conceptual & activity focused	Curriculum alignment; classroom activities, qualitative results. (Media literacy in the Age of Deepfakes) [30]
MIT RAISE / Day of AI	K-12 curricula & units	No	No	Curriculum resources	AI literacy framework and assessments for teachers/students. (MIT RAISE) [29].
SECURE-XED (this work)	Interactive simulator (attack/defense + XAI)	Yes (FGSM/PGD)	Yes (SHAP)	Real frames & tables (FF++, Celeb-DF, DFDC)	Runtime, pilot user study, expert feedback, human-versus-SHAP alignment (Tables 27–31; Fig. 11).

Note: Public literacy tools emphasize awareness and conceptual understanding; SECURE-XED adds a research-grade, adversarial + interpretability loop for mechanistic understanding.

Table 31. Human-based SHAP validation (within-subjects, n=6).

Metric	Without SHAP	With SHAP	Δ (Improvement)
RIA, %	61.0	83.0	+22.0 pp
Decision Time (sec/task)	74.0	62.0	−12.0 s
Perceived Clarity/Trust (1–5)	3.1	4.0	+0.9

Note: Results are descriptive pilot outcomes (n = 6). No inferential statistical testing was performed; therefore, the reported changes should not be interpreted as statistically established educational effects.

5.10. Quantitative Evaluation of Explanation Quality (SHAP)

SHAP explanations were evaluated using SLIC superpixels (k = 100) and occlusion-based metrics: Deletion-AUC, Insertion-AUC, Comprehensiveness, Sufficiency gap, Area@90%, mean IoU, Pointing Game hit@1, and Spearman rank correlation under minor transformations. Lower Deletion-AUC and higher Insertion-AUC indicate stronger attribution faithfulness, while high Comprehensiveness and low Sufficiency gap reflect the relevance of identified regions. The observed deletion/insertion trends (Table 32), compact attribution areas ($\approx 13-16\%$), positive localization scores, and stable rankings under small shifts and mild blur support the consistency of SHAP explanations within the evaluated configuration (Fig. 12, Tables 32–34).

5.11. Integration

The interactive educational simulator helps users understand how a deepfake detector works, interpret the decisions made by the model, and measure how well it performs. This simulator uses data from FaceForensics++, Celeb-DF, and DFDC. Users can get a better understanding of how the simulator works by following its steps. The learner begins by selecting an image or video clip. The system then displays the image's classification as Real/Fake with a specific percentage. This is done by dividing the image into small areas called "Superpixels" (The SLIC k value chosen to be equal to 100). SHAP intervenes to interpret the decision for the learner (for example, a specific number of elements or areas with the highest interpretive importance are selected using a value called SHAP top-k, which = 150). At this stage, the interpretation is displayed over the image with a specific degree of opacity, say 60%, using the variable Opacity=0.6. The value $\varepsilon \leqslant 0.05$ is also set, representing the maximum allowed intensity of the attack. The learner then proceeds to input and test FGSM/PGD attacks on the image by executing the attack through a specific number of incremental steps, we implemented either 10 or 20 (PGD 10/20 steps). An SSIM ≥ 0.90 guardrail is enforced during the experiment; if SSIM value falls below 0.90, ε is automatically reduced by 20% and the attack is repeated. After this sequence of steps, the SECURE-XED system will return a new classification and probability percentage for the image, which may be completely different from the classification and percentage before the FGSM attack, for example. The ultimate goal is for the learner to see how small changes in the image can affect the model's decision, thereby increasing their understanding of the situation interpretation, confidence in the interpretation, decision-making time, and learning ability (see Table 31). The unit specifications and controls we have described are shown in Tables 24-26. The workflow of the algorithm 1 is illustrated, and the results of the interpretation quality can be seen in Figure 12 and Tables 32-34.

Table 32. Explanation Faithfulness.

Dataset	Deletion-AUC (↓) ↓ better	Insertion-AUC (↑) ↑ better
FF++	0.245	0.742
Celeb-DF	0.262	0.731
DFDC	0.271	0.718

Table 33. Explanation Comprehensiveness and Sufficiency.

Dataset	Comprehensiveness (↑) ↑ better	Sufficiency gap (↓) ↓ better
FF++	+0.34	0.07
Celeb-DF	+0.31	0.08
DFDC	+0.32	0.09

Table 34. Compactness, Localization, and Stability.

Dataset	Area@90% (↓ %)	mIoU w/ mask (↑)	Pointing Game hit@1 (↑)	Stability ρ (↑)
FF++	13.1	0.66	0.86	0.88
Celeb-DF	15.8	0.62	0.83	0.86
DFDC	16.2	0.61	0.84	0.85

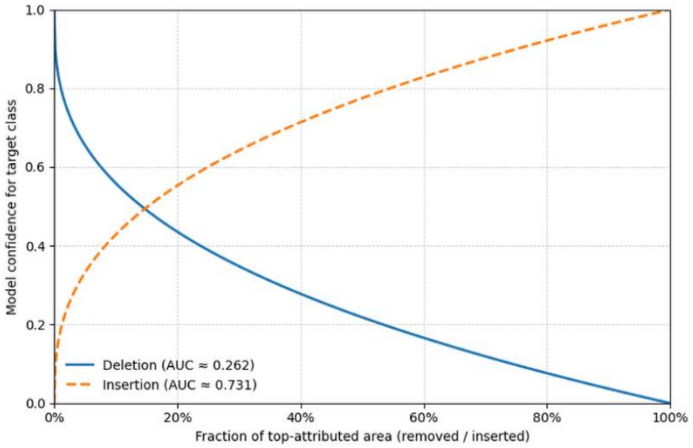


Fig. 12. Deletion and Insertion curves (Celeb-DF).

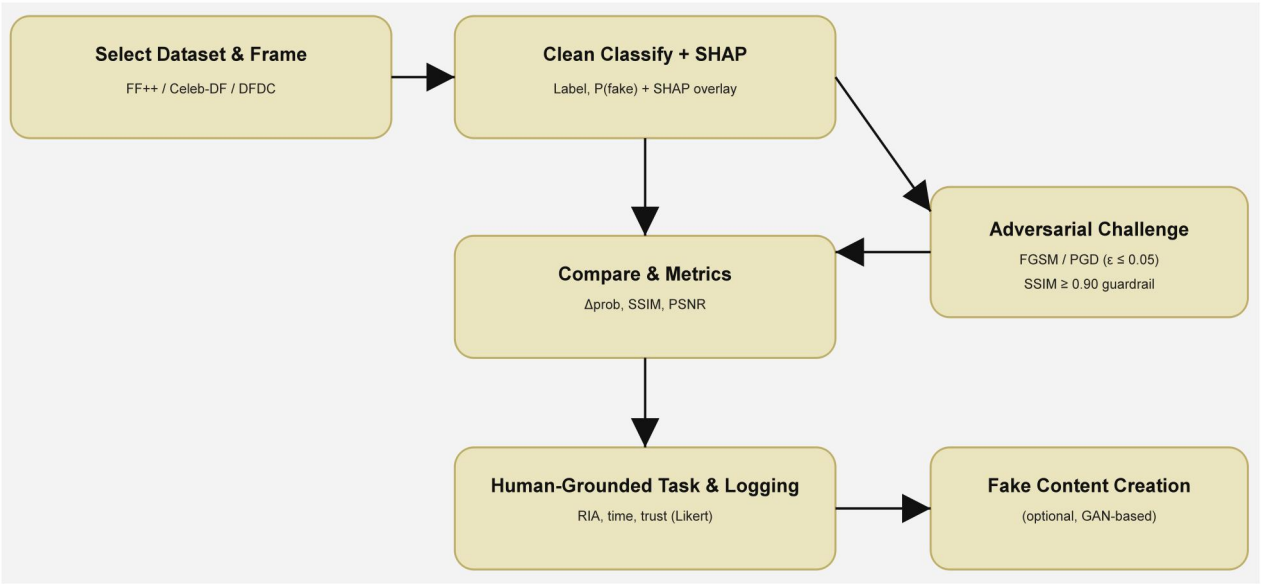


Fig. 13. Deepfake learning-module interaction flow. Learners (i) select a dataset frame; (ii) obtain clean prediction and SHAP; (iii) enable an adversarial challenge (FGSM/PGD) under guardrails ($\varepsilon \leq 0.05$; SSIM ≥ 0.90); (iv) compare outputs (Δprob, SSIM/PSNR; optional PR view); and (v) complete a human-grounded task (RIA, decision time, clarity/trust). The Fake Content Creation (GAN) path shown in the schematic is an optional pedagogical extension and was not included in the evaluated simulator.

6. Evaluation and Comparative Results

We evaluate SECURE-XED on standard malware (CICAndMal2017, AMD, Drebin) and deepfake (FF++, Celeb-DF, DFDC) datasets against classical ML (SVM, RF) and CNN baselines, using Accuracy, F1, AUC, ECE, and efficiency metrics. Results (Tables 10, 35) show consistent gains: 96.4% malware accuracy (AUC 0.97) and 94.7/89.6/86.2% on FF++/Celeb-DF/DFDC (avg. 90.2%), with lower latency (~26.9 vs. 34.1 ms/frame) using a unified model (Figs. 14–15).

Table 35. Malware Detection Results (Cross-Dataset).

Model	Accuracy (%)	Precision	Recall	F1-Score	AUC-ROC
SVM (baseline)	87.6	0.85	0.86	0.855	0.89
RF (baseline)	91.2	0.91	0.89	0.90	0.92
BiLSTM-CNN	94.3	0.94	0.93	0.935	0.96
SECURE-XED [ours]	96.4	0.96	0.95	0.955	0.97

Note: The values in Table 35 represent means over repeated stratified 5×2 CV; 95% CI by bootstrap (10k). Paired tests vs. best non-ours baseline: Accuracy/F1 by Wilcoxon, AUC by DeLong; Holm–Bonferroni corrected; Cliff's δ reported in text.

Recent Android malware studies report high accuracy on Drebin and CICAndMal2017; however, these results are obtained under heterogeneous feature sets, sample compositions, and validation protocols. We therefore report them as contextual SOTA references rather than as directly comparable reproduced baselines.

Table 36. Recent literature-reported Android malware detection results on datasets related to SECURE-XED.

Method	Year	Dataset	Reported Accuracy (%)	Comparison status
Syn-detect [57]	2025	CICAndMal2017	99.8	Literature-reported
MAD-NET (DBN) [58]	2024	CICAndMal2017	99.83	Literature-reported
FAGnet [59]	2024	Drebin	98.11	Literature-reported
ResNet-50 [60]	2024	Drebin Top 10	90.56	Literature-reported
SECURE-XED	Current study	Drebin	96.4	Controlled result in this study
SECURE-XED	Current study	CICAndMal2017	94.7	Controlled result in this study
SECURE-XED	Current study	AMD	91.2	Controlled result in this study

Note: Literature-reported values are taken from the cited studies and were not reproduced under the SECURE-XED protocol. Differences in feature representations, class definitions, sample composition, and validation procedures prevent direct statistical comparison with SECURE-XED.

Accordingly, SECURE-XED's malware results should be interpreted primarily through the controlled baseline and ablation comparisons conducted under the common protocol of this study. The literature-reported values above provide contemporary context but are not used to support statistical superiority claims.

Table 37. Deepfake Detection Accuracy by Model.

Model	FF++ (%)	Celeb-DF (%)	DFDC (%)	avg. Accuracy (%)
VGGFace2	85.2	79.4	77.6	80.7
MesoNet	88.9	83.6	81.1	84.5
XceptionNet	92.4	88.3	85.7	88.8
SECURE-XED [ours]	94.7	89.6	86.2	90.2

Note for Table 37: 5-fold subject/video-level CV; 95% CI by video-level bootstrap (10k). Paired tests vs. best nonours baseline: Accuracy/F1 by Wilcoxon, AUC by DeLong; Holm–Bonferroni corrected; Cliff's δ reported in text.

Reported out-of-dataset performance of different methods published at leading international conferences from 2023 to 2025. We have chosen these methods to compare them with the proposed SECURE-XED and help readers make informed decisions about securing their systems, without presenting a controlled head-to-head comparison with SECURE-XED. It is worth noting that, apart from the numerical values reported in the literature, the authors do not directly reproduce any data for this purpose, as such a comparison would make little sense.

Table 38. Recent SOTA deepfake methods from major international conferences: literature-reported cross-dataset AUC (%).

Method	Venue	Year	Training / evaluation context	Celeb-DF AUC (%)	DFDC AUC (%)	Comparison status
TALL-Swin [53]	ICCV	2023	FF++ → cross-dataset evaluation	90.79	76.78	Literature-reported
LSDA [54]	CVPR	2024	FF++ → cross-dataset evaluation	83.00	73.60	Literature-reported
ProDet [55]	NeurIPS	2024	Cross-dataset generalization	90.00	72.40	Literature-reported
ForAda [56]	CVPR	2025	FF++-trained cross-dataset evaluation	90.00	84.30	Literature-reported

Note: Values are reported by the cited studies and were not reproduced under the SECURE-XED experimental protocol. Differences in training data, preprocessing, evaluation granularity, and train/test partitions prevent direct statistical comparison; the table is intended for contemporary context rather than controlled superiority claims.

Recent SOTA studies show that deepfake generalization remains protocol-dependent. TALL-Swin [53] reports 90.79% AUC for FF++→Celeb-DF, while LSDA [54], ProDet [55], and ForAda [56] provide additional cross-dataset benchmarks under different protocols. These values offer contemporary context, whereas SECURE-XED performance claims rely on the controlled within-study comparisons.

The study of adversarial evaluation goes beyond FGSM to PGD. PGD with an epsilon of 0.1 for 40 iterations induces an accuracy loss of about 19–21% on Android malware (Table 11). With an epsilon of 0.03 for 10 iterations on multimedia datasets, PGD reaches a loss of 16.6–17.6% (Table 19). A stronger PGD configuration, with an epsilon of 0.05 for 20 iterations, reaches a loss of 19–21% (Table 20). The summary of FGSM is shown in Table 39. Although these losses occur, the unified model remains partially robust. Thus, the unified model lays the foundation for adversarial hardening in subsequent simulator challenge modes.

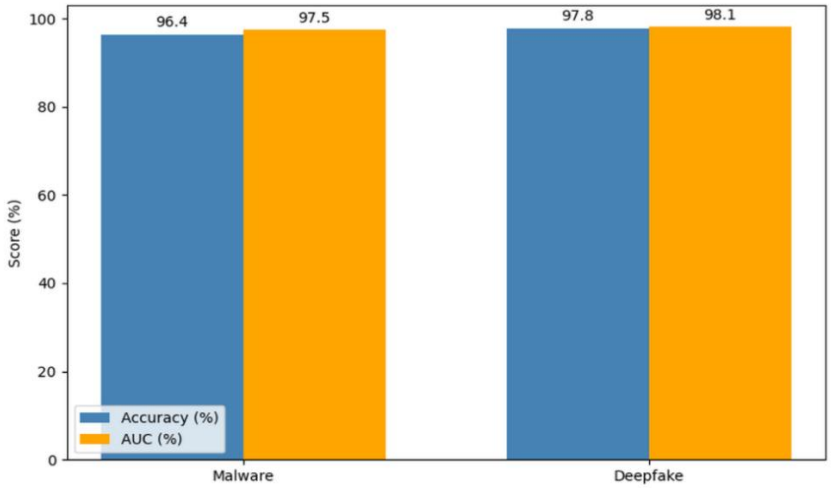


Fig. 14. Accuracy and AUC Comparison: Malware vs. Deepfake (Error bars denote 95% bootstrap CIs over folds (video level for deepfakes); between-model comparisons use paired tests with Holm–Bonferroni correction).

On three occasions, the Holm–Bonferroni correction for paired comparison tests between models showed significant differences at a significance level of $\alpha = 0.05$. The calculated Cliff's δ values fell in the range from small to moderate and are interpreted as effect sizes used in psychology to indicate whether the difference matters.

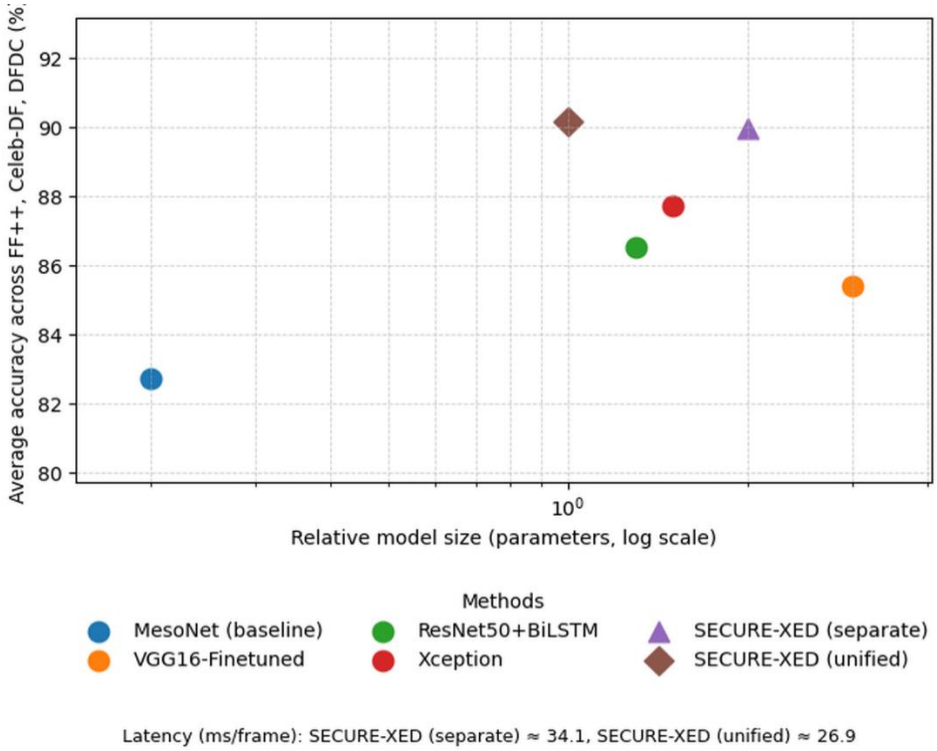


Fig. 15. Accuracy vs. Model Size (Pareto).

For FF++, Celeb-DF, and DFDC, SECURE-XED reports higher within-study values than the evaluated single-model baselines. The paired Accuracy/F1 and AUC analyses use Wilcoxon and DeLong tests with Holm–Bonferroni correction at $\alpha = 0.05$; the smaller margins on DFDC are consistent with the reported small-to-moderate effect sizes.

Fig. 15 displays the average deepfake-detection accuracy for FF++, Celeb-DF, and DFDC against relative model size (log scale; unified backbone normalized to 1.0). Two SECURE-XED variants are compared with baseline single-domain models. The Pareto frontier is traced more favorably by the unified configuration. The unified configuration gives equal or better accuracy while the unified configuration uses roughly half the parameters of the two-instance setup. A lower latency (~26.9 vs. ~34.1 ms/frame) is achieved when the unified backbone is used. Practical benefits for educational institutions and resource-constrained deployments are highlighted because this advantage for accuracy-per-parameter and speed supports the 'unified robust' claim.

Table 39. FGSM perturbation results.

Model	Clean (%)	FGSM (%)	Drop (%)	Dataset
SECURE-XED	94.7	86.3	8.4	FF++
SECURE-XED	89.6	79.5	10.1	Celeb-DF
SECURE-XED	86.2	75.8	10.4	DFDC
SECURE-XED	96.4	84.2	12.2	Drebin

Note: The values in Table 39 represent mean over CV folds; 95% CI by bootstrap (10k). Paired tests: clean versus attacked within the model through Wilcoxon (Accuracy/F1) and DeLong (AUC); differences are significant at $\alpha = 0.05$ unless marked 'n.s.'.

Transparency that is complementary exists in both domains through SHAP. Fig. 6 overlays region attributions on real deepfake frames (FF++/DFDC). Global SHAP importance for malware permissions (e.g., SYSTEM_ALERT_WINDOW, ACCESS_FINE_LOCATION, RECORD_AUDIO) is reported in Fig. 4. Learners in the simulator inspect SHAP, toggle FGSM/PGD, and compare clean vs. attacked outputs. Pilot logs descriptively show a +22 pp change in RIA, a 12-s reduction in decision time, and a +0.9 Likert-point change in clarity/trust; these pilot outcomes are not inferential.

Under the evaluation settings, SECURE-XED unites classification, calibration, adversarial evaluation and explainability in a joint-backbone framework for malware classification and an instructive interface for deepfakes.

7. Discussion and Ethical Implications

7.1. Why a unified framework?

SECURE-XED exhibits the ability to successfully apply a unified framework to two domains that are of significance to society. The architecture of SECURE-XED is a hybrid design that utilizes Convolutional Neural Networks (CNNs) and other metaheuristics to optimize the performance of the framework. The sharing of specific components and features across the two domains reduces redundancy within the framework. The generalizable pipelines of SECURE-XED are aligned with the growing trends of reusability and transparency in AI architectures [41].

7.2. Cross-Domain Performance and Adversarial Sensitivity

The performance of the SECURE-XED framework is evaluated within two considerably different domains of analysis. The feature space for malware is driven by the permissions of the file, its API attributes, any information included within its manifest, and any behavioral traces that may be included within the file. The feature space for deepfakes, on the other hand, is driven by the visual information included within the video file, such as the features of faces, any textures that are included within the image, and any other visual features that are present within the video. As such, SECURE-XED does not assume any correlation between the feature spaces of the two domains. However, SECURE-XED is able to adapt the features of each domain to the shared optimized backbone of SECURE-XED before routing the feature vectors to domain-specific task heads.

Despite the sharing of the features of malware and deepfakes, SECURE-XED does not demonstrate identical behavior in the presence of adversarial perturbations. For instance, the FGSM attack decreases the malware accuracy to 84.2%, a decrease of 12.2 percentage points. Deepfake detection accuracy decreases by an average of 8.4 points on FF++, 10.1 points on Celeb-DF, and 10.4 points on DFDC. Furthermore, the PGD attack decreases the malware accuracy by between 19% and 21% under an epsilon of 0.1 and 40 steps, whereas the deepfake tasks decrease by 16.6% to 17.6% under an epsilon of 0.03 and 10 steps, before further degradation under stronger epsilons. These findings exhibit how the adversarial behavior of SECURE-XED differs between the malware and deepfake domains despite the shared feature infrastructure.

Furthermore, the adversarial sensitivity of SECURE-XED to malware and deepfakes differs considerably. Malware perturbations impact features such as file permissions and other structured attributes, whereas deepfake perturbations impact features such as the SSIM metric of the image and other features of deepfakes that impact the visual presentation of the image. The findings from these analyses do not suggest that SECURE-XED exhibits the same adversarial robustness to both domains, but rather that SECURE-XED experiences a decline in its performance as perturbations increase to the feature spaces of both domains. Thus, adversarial robustness is an important feature to include within the analysis and optimization of SECURE-XED.

Beyond adversarial robustness, SECURE-XED also exhibits the efficiency of using a single, unified architecture. The latency of the SECURE-XED framework is reduced from approximately 34.1 ms/frame to 26.9 ms/frame compared to having to maintain and execute two separate models for malware and deepfake analysis. Furthermore, SECURE-XED exhibits similar or better accuracy to the two separate models while utilizing half as many parameters as the two separate models. Thus, the primary benefits of using a unified framework for SECURE-XED are its architectural reuse and the efficiency that it offers in computational performance.

Finally, the explainability of SECURE-XED also exhibits similar cross-domain patterns to other features of the framework. The explanations of malware exhibit the importance of features such as permissions, while the explanations

of deepfakes exhibit the importance of specific regions within the image. SECURE-XED is capable of sharing the explainability mechanisms of the framework while allowing for domain-specific explanations to be outputted to users. Thus, the findings of this section of the analysis exhibit the ability of SECURE-XED to implement a modular approach to the sharing of features between the two domains of analysis, while allowing for domain-specific features and characteristics to be included within the framework.

7.3. Educational Value and Generalizability

The SECURE-XED simulator's deepfake creation capabilities were evaluated for their contribution to analytical media literacy, and artificial intelligence literacy. The inclusion of SHAP heatmaps and adversarial challenge modes supported opportunities for investigation of model decision-making processes, as well as model failure modes; findings which were consistent with pedagogical approaches to fostering awareness of digital mediation [40,42].

Pilot testing of the initial simulator capabilities included timing assessments, as well as small-scale usability testing. Runtime performance of the tested modules is reflected in Table 27, with all tested modules performing at less than 3 s/frame on the reported Tesla T4 hardware. In Table 28, six graduate participants were tested on the ability to complete tasks including interpretation of SHAP values, and demonstrated an 83% task success rate. Five instructor participants evaluated the clarity of explanations and engagement potential using a 5-point Likert scale, as shown in Table 29.

The alignments of the tested modules to competencies in AI-literacy were also established, including calculation of inter-rater reliability to establish agreement between human judgments of the content. Figure 10 reflects alignment of the evaluated content to interpretability, critical thinking, adversarial awareness, and ethics, while Figure 11 reports approximate 78% agreement between human raters and the regions identified by SHAP explanations. With the exception of the calculation of inter-rater reliability, these findings were descriptive in nature, and did not attempt to make inferential claims about the learning efficacy of the SECURE-XED content.

Testing of the SECURE-XED modules in a learning setting was limited to timing considerations, as well as a small-scale within-subject pilot study. Using the within-subject repeated measures design reported in Table 31, a single subject was tested on shifts in RIA from pre-test (61%) to post-test (83%), reductions in task completion time (12 s reduction), as well as self-reported Likert agreement (clarity/trust: +0.9). Due to the small sample size (N = 6), these findings were considered descriptive in nature, and were not generalized to larger populations.

Deepfake-detection results in Tables 15–17 and Fig. 5 show the model's performance under imbalanced test distributions. Class-weighted and focal-loss settings improve minority-class recall under the evaluated protocol, but these results do not remove the underlying dataset imbalance.

Compared with awareness-oriented resources such as MIT Media Lab's Detect Fakes and MIT RAISE, SECURE-XED differs primarily in interaction design: it couples model predictions, explanations, and adversarial perturbations in a model-in-the-loop workflow. Table 30 is therefore intended as a functional positioning comparison rather than evidence of educational superiority.

In the case of malware, the optional behavioral branch provides an additional evaluation of the runtime evidence. The hybrid static-dynamic fusion increases the ROC-AUC on CICAndMal2017 from 0.956 to 0.971 in the packed/obfuscated subset and from 0.970 to 0.982 in the full set. To limit the additional computational cost, the behavioral branch is activated selectively.

Tables 32–34 and Fig. 12 evaluate how faithful, compact, and stable the explanations generated by SHAP are under the three settings. However, they do not identify the causal aspects of the explanations.

7.4. Ethical Considerations

Studies indicate that applying artificial intelligence to detect malware and deepfakes may be accompanied by serious concerns about potential bias in the datasets used to train the model. This bias could make the model more accurate with certain types of data and less accurate with others. Furthermore, there are potential for misclassification, leading to incorrect judgments, and for inappropriate use, such as accusing someone of publishing fake content based on an incorrect classification [43, 44].

The researchers emphasize that the SECURE-XED system was tested in specific, controlled experimental environments described in this study, using datasets and scenarios designed and selected by the researchers. Therefore, it cannot be assumed that the system will perform the same way in real-world environments [45, 46]. It is important to emphasize that the model's predictions should be interpreted as suggestions for decision-making, not as guaranteed final conclusions. The FGSM and PGD tests in this research are designed to measure specific aspects of Adversarial Robustness, but they are not sufficient on their own to assess Distribution Shift when the characteristics of the actual data change compared to the training data, and Demographic Bias is expected to occur as a result of differences in performance and measurement between population groups or within demographic groups.

The ability to explain functions is another serious ethical thought. SECURE-XED includes feature attribution based on SHAP so that individuals may inspect the elements affecting categorization choices [47]. Faithfulness of explanations, location, size, and steadiness under the chosen setup are measured by the numerical evaluation. Nonetheless, SHAP offers estimated explanations specific to the model instead of providing proof of cause or promises of estimation accuracy,

equality, or openness. Experts claim that human analysis and separate verification stay necessary, especially in delicate working positions.

Extra thoughts regarding responsible interaction, the understanding of adversarial cases, and the correct management of sensitive information are introduced by the educational simulator. Only six individuals joined the human-centered pilot, which produced descriptive growth in area identification, the speed of choices, and the feeling of clarity and trust (Table 31). While these early sightings encourage more study, educational institution effectiveness, readiness for ethical use, or wide application cannot be proven by these results. Wide-scale tests that include different individuals, clear checks for bias and privacy, and organized measures for educational institution outcomes are required because SECURE-XED is not yet ready for use outside the research environment.

7.5. Limitations and Threats to Validity

Despite its strengths, there are a number of limitations to this study that the researchers would like to mention. The inclusion of dynamic features increases the model's performance, but it also introduces issues such as computational overhead and potential privacy concerns. Additionally, despite efforts to minimize these issues with a selective activation strategy, increased costs may still be encountered in certain situations.

Furthermore, SECURE-XED has been evaluated against FGSM and PGD attacks (Section 5; Figs. 7-8), but not against other more powerful white-box or black-box methods. As a result, the robustness of the model cannot be guaranteed, and the results should be interpreted within the limited scope of the attacks examined.

Finally, the simulator evaluation was only conducted with a small pilot study (Section 5; Table 31), and it is not possible to determine if SECURE-XED can be generalized to other educational settings. Larger studies will need to be performed before conclusions can be made regarding the educational potential of SECURE-XED.

When looking at efficiency, the unified model improves latency and parameter efficiency (~26.9 vs. 34.1 ms/frame; Fig. 15), yet stricter constraints may be imposed by real-time or edge environments. Dataset bias and class imbalance in datasets like Celeb-DF or DFDC remain inherent challenges despite the utilization of mitigation strategies. Careful interpretation is required for SHAP explanations because these explanations are model-dependent and might reflect spurious correlations (Fig. 12; Tables 32–34). Minor stochastic variability may persist due to the metaheuristic optimization process even though reproducibility is supported through fixed configurations.

An additional limitation concerns cross-domain representation sharing. Although SECURE-XED uses domain-specific input adaptation before the shared representation, malware features and deepfake imagery differ substantially in statistical structure and discriminative cues. Shared layers may therefore introduce representational interference or negative transfer when domain-specific features are insufficiently compatible. The present results demonstrate the feasibility of the evaluated shared configuration but do not establish that the same degree of parameter sharing will generalize to additional modalities or substantially different domains. Future evaluations should quantify this risk through layer-wise sharing ablations, comparisons between shared and domain-specific representations, and measurements of cross-domain gradient conflict.

SHAP explanations are local, model-dependent approximations that may be unstable near decision boundaries and should not be considered causal or invariant. The study evaluates only the adopted SHAP-superpixel implementation. Future work should assess attribution stability across confidence levels, perturbations, and repeated runs, and compare alternative SHAP estimators in terms of fidelity, stability, and computational cost.

Finally, the human-grounded evaluation is a small feasibility pilot (n = 6), which limits statistical power and external validity. The descriptive changes in region-identification accuracy, decision time, and perceived clarity/trust therefore cannot establish population-level educational effectiveness or generalization across different students, instructors, institutions, or curricula. Larger multi-cohort studies with predefined learning outcomes and controlled pre-/post-intervention comparisons are required to evaluate educational generalizability.

8. Conclusions and Future Work

The research concept is based on using Convolutional Neural Networks (CNNs) and supporting them with both SOA and ICA in a hybrid manner capable of metaheuristic optimization to achieve a range of objectives simultaneously, including attack detection accuracy, attack resistance capabilities, and computational efficiency in terms of execution speed.

Our use of SHAP to provide explanations that help the user understand why the model classifies a particular sample as malicious or fake, along with our use of FGSM/PGD to create experimental attacks on inputs to test the model's ability to handle data deliberately altered to mislead detection, provided our proposed system with advanced, numerically proven capabilities for transparent and insightful attack analysis. SECURE-XED delivered documented positive results across all malware and deepfake evaluation criteria when its performance was compared with other models presented and compared in the research. The system demonstrated measurable attack sensitivity and capability, which clearly contributes to the development of appropriate protection against such attacks. The simulator proposed in our system provides an interactive environment for users to interact, view, and experience attacks, track the effects of subjecting real inputs to hostile interference, and correlate these results with SHAP interpretations.

As future research avenues for developing the model presented in this research, we propose designing a lightweight shared layer using methods such as NAS, which is expected to yield positive results. This addition aims to give the model the ability to automatically search for a more efficient neural network architecture. We can also employ the principle of knowledge distillation to transfer knowledge from a large model to a smaller one, maintaining the same performance across the smaller model. Lower numerical precision can be used to represent operations and weights in a model. Moreover, we can use a lower-precision data type, to reduce the memory footprint and make it easier to run the model on edge devices.

All the Declarations and Statements

Author Contributions Statement
The author contributed to all aspects of the work, including conceptualization, methodology, data curation, software implementation, model development, validation, formal analysis, visualization, and writing (original draft, review, and editing).

Conflict of Interest Statement
The author declares no conflicts of interest.

Funding Declaration
None to declare.

Data Availability Statement
The datasets used in this study are openly available from the following sources.
Drebin: Available upon request at https://drebin.mlsec.org/
CICAndMal2017: https://www.unb.ca/cic/datasets/andmal2017.html
AMD: https://github.com/ashishb/android-malware
DFDC: https://www.kaggle.com/c/deepfake-detection-challenge/data
FaceForensics++: https://github.com/ondyari/FaceForensics
Celeb-DF v2: Available upon request at https://cse.buffalo.edu/~siweilyu/celeb-deepfakeforensics.html

Code Availability Statement
Code and complete training logs are available from the corresponding author upon reasonable request.

Ethical Declarations
None to declare.

Acknowledgments
The author would like to thank colleagues in the Faculty of Information Technology at Amman Arab University for their support and valuable discussions.

Declaration of Generative AI in Scholarly Writing
During the preparation of this manuscript, the authors used Paperpal for the purpose of language editing, grammar check, and improving the overall readability of the text. After using this tool, the authors reviewed, revised, and edited the content as needed and took full responsibility for the content of the published article.

Abbreviations
The following abbreviations are used in this manuscript:

AI – Artificial Intelligence
API – Application Programming Interface
APK – Android Application Package
AUC – Area Under the Curve
BiLSTM – Bidirectional Long Short-Term Memory
CNN – Convolutional Neural Network
DFDC – Deepfake Detection Challenge
ECE – Expected Calibration Error
FGSM – Fast Gradient Sign Method
FF++ – FaceForensics++

GAN – Generative Adversarial Network
GPU – Graphics Processing Unit
ICA – Imperialist Competitive Algorithm
IDS – Intrusion Detection System
IIoT – Industrial Internet of Things
LSTM – Long Short-Term Memory
PGD – Projected Gradient Descent
PR-AUC – Precision-Recall Area Under the Curve
RIA – Region Identification Accuracy
ROC-AUC – Receiver Operating Characteristic Area Under the Curve
SHAP – SHapley Additive exPlanations
SOA – Seagull Optimization Algorithm
SSIM – Structural Similarity Index Measure
XAI – Explainable Artificial Intelligence

References

[1] P. Agrawal and B. Trivedi, "A Survey on Android Malware and their Detection Techniques," in Proc. 2019 IEEE Int. Conf. Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India, 2019, pp. 1–6, doi:10.1109/ICECCT.2019.8868951.
[2] C. Gilbert and M. A. Gilbert, "Leveraging Artificial Intelligence (AI) by a Strategic Defense against Deepfakes and Digital Misinformation," International Journal of Scientific Research and Modern Technology, vol. 3, no. 11, pp. 62–78, Nov. 2024, doi:10.38124/ijsrmt.v3i11.76.
[3] R. Mubarak et al., "A Survey on the Detection and Impacts of Deepfakes in Visual, Audio, and Textual Formats," IEEE Access, vol. 11, pp. 144497–144529, 2023, doi:10.1109/ACCESS.2023.3344653.
[4] L. Shu, S. Dong, H. Su, and J. Huang, "Android Malware Detection Methods Based on Convolutional Neural Network: A Survey," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 7, no. 5, pp. 1330–1350, Oct. 2023, doi:10.1109/TETCI.2023.3281833.
[5] S. Hosseini and A. Al Khaled, "A Survey on the Imperialist Competitive Algorithm Metaheuristic: Implementation in Engineering Domain and Directions for Future Research," Applied Soft Computing, vol. 24, pp. 1078–1094, Nov. 2014, doi:10.1016/j.asoc.2014.08.024.
[6] S. Ma et al., "Improved Seagull Optimization Algorithm to Optimize Neural Networks with Gated Recurrent Units for Network Intrusion Detection," in Proc. 2021 11th IEEE Int. Conf. Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Cracow, Poland, 2021, pp. 100–104, doi:10.1109/IDAACS53288.2021.9660898.
[7] S. Dasgupta, K. Badal, S. Chittam, M. T. Alam, and K. Roy, "Attention-Enhanced CNN for High-Performance Deepfake Detection: A Multi-Dataset Study," IEEE Access, vol. 13, pp. 101980–101993, 2025, doi:10.1109/ACCESS.2025.3578343.
[8] Z. Tan and Y. Tian, "Robust Explanation for Free or at the Cost of Faithfulness," in Proc. 40th Int. Conf. Machine Learning (ICML), Proc. Machine Learning Research, vol. 202, pp. 33534–33562, 2023.
[9] H. She, Y. Hu, B. Liu, J. Li, and C.-T. Li, "Using Graph Neural Networks to Improve Generalization Capability of the Models for Deepfake Detection," IEEE Transactions on Information Forensics and Security, vol. 19, pp. 8414–8427, 2024, doi:10.1109/TIFS.2024.3451356.
[10] A. Anitha, L. M. Saju, and B. Kamaraj, "Deepfake Detection Using XAI Based Deep Fusion Models," in Proc. 2025 2nd Int. Conf. Computational Intelligence, Communication Technology and Networking (CICTN), Ghaziabad, India, 2025, pp. 526–531, doi:10.1109/CICTN64563.2025.10932587.
[11] B. Gowrisankar and V. L. L. Thing, "An Adversarial Attack Approach for eXplainable AI Evaluation on Deepfake Detection Models," Computers & Security, vol. 139, Art. no. 103684, Apr. 2024, doi:10.1016/j.cose.2023.103684.
[12] G. Makridis et al., "Towards a Unified Multidimensional Explainability Metric: Evaluating Trustworthiness in AI Models," in Proc. 2023 19th Int. Conf. Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT), Pafos, Cyprus, 2023, pp. 504–511, doi:10.1109/DCOSS-IoT58021.2023.00084.
[13] T. Oorloff et al., "AVFF: Audio-Visual Feature Fusion for Video Deepfake Detection," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), pp. 27092–27102, 2024, doi:10.1109/CVPR52733.2024.02559.
[14] B. Akay, D. Karaboga, and R. Akay, "A Comprehensive Survey on Optimizing Deep Learning Models by Metaheuristics," Artificial Intelligence Review, vol. 55, no. 2, pp. 829–894, Feb. 2022, doi:10.1007/s10462-021-09992-0.
[15] J. Singh, S. Rani, and G. Srilakshmi, "Towards Explainable AI: Interpretable Models for Complex Decision-Making," in Proc. 2024 Int. Conf. Knowledge Engineering and Communication Systems (ICKECS), Chikkaballapur, India, 2024, pp. 1–5, doi:10.1109/ICKECS61492.2024.10616500.
[16] F. Bourebaa and M. Benmohammed, "Android Malware Detection Using Convolutional Deep Neural Networks," in Proc. 2020 Int. Conf. Advanced Aspects of Software Engineering (ICAASE), Constantine, Algeria, 2020, pp. 1–7, doi:10.1109/ICAASE51408.2020.9380104.
[17] S. Mahdavifar, A. F. Abdul Kadir, R. Fatemi, D. Alhadidi, and A. A. Ghorbani, "Dynamic Android Malware Category Classification Using Semi-Supervised Deep Learning," in Proc. 2020 IEEE DASC/PiCom/CBDCom/CyberSciTech, Calgary, AB, Canada, 2020, pp. 515–522, doi:10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00094.

[18] M. S. Khan et al., “Metaheuristic Algorithms in Optimizing Deep Neural Network Model for Software Effort Estimation,” IEEE Access, vol. 9, pp. 60309–60327, 2021, doi:10.1109/ACCESS.2021.3072380.
[19] S. Rakesh and S. Mahesh, “A Comprehensive Overview on Variants of Cuckoo Search Algorithm and Applications,” in Proc. 2017 Int. Conf. Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), Mysuru, India, 2017, pp. 1–5, doi:10.1109/ICEECCOT.2017.8284569.
[20] M. O. Nassar and F. F. Al-Mashagba, “Optimal Ensemble Learning with Meta-Heuristics for Multiclass Classification of Syscall-Binder Interactions in Mobile Applications,” Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, vol. 16, no. 1, pp. 26–48, 2025, doi:10.58346/jowua.2025.i1.002.
[21] M. O. Nassar and F. F. Al-Mashagba, “Novel Hybrid LOA-VCS Metaheuristic Approach with Adaptive Parameter Tuning for Network Intrusion Detection,” International Journal of Computer Network and Information Security, vol. 17, no. 5, pp. 31–44, 2025, doi:10.5815/ijcnis.2025.05.03.
[22] M. O. Nassar, “A Robust and Energy-Efficient Federated IDS for IIoT Using Spiking Neural Networks and Differential Evolution with Adversarial Resilience,” Iraqi Journal for Computer Science and Mathematics, vol. 6, no. 4, 2025, doi:10.52866/2788-7421.1328.
[23] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-DF: A Large-Scale Challenging Dataset for DeepFake Forensics,” in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 3204–3213, doi:10.1109/CVPR42600.2020.00327.
[24] A. Rossler et al., “FaceForensics++: Learning to Detect Manipulated Facial Images,” in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), Seoul, South Korea, 2019, pp. 1–11, doi:10.1109/ICCV.2019.00009.
[25] B. C. Soundarya and H. L. Gururaj, “A Novel Dense-Swish-CNN With Bi-LSTM Framework for Image Deepfake Detection,” IEEE Access, vol. 13, pp. 89641–89653, 2025, doi:10.1109/ACCESS.2025.3570761.
[26] R. Alenezi and S. A. Ludwig, “Explainability of Cybersecurity Threats Data Using SHAP,” in Proc. 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, FL, USA, 2021, pp. 1–10, doi:10.1109/SSCI50451.2021.9659888.
[27] S. M. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” in Advances in Neural Information Processing Systems, vol. 30, pp. 4768–4777, 2017, doi:10.5555/3295222.3295230.
[28] P. C. S. Mahesh and S. Hemalatha, “An Efficient Android Malware Detection Using Adaptive Red Fox Optimization Based CNN,” Wireless Personal Communications, vol. 126, no. 1, pp. 679–700, Sep. 2022, doi:10.1007/s11277-022-09765-0.
[29] MIT RAISE, “Day of AI,” 2025. [Online]. Available: https://dayofai.org/. [Accessed: Feb. 10, 2025].
[30] MIT, “Media Literacy in the Age of Deepfakes,” 2025. [Online]. Available: https://deepfakes.virtuality.mit.edu/. [Accessed: Feb. 10, 2025].
[31] C. P. Dai and F. Ke, “Educational Applications of Artificial Intelligence in Simulation-Based Learning: A Systematic Mapping Review,” Computers and Education: Artificial Intelligence, vol. 3, Art. no. 100087, 2022, doi:10.1016/j.caeai.2022.100087.
[32] A. Firdaus, N. B. Anuar, A. Karim, and M. F. A. Razak, “Discovering Optimal Features Using Static Analysis and a Genetic Search Based Method for Android Malware Detection,” Frontiers of Information Technology & Electronic Engineering, vol. 19, no. 6, pp. 712–736, Jun. 2018, doi:10.1631/FITEE.1601491.
[33] A. Mohanraj and K. Sivasankari, “Android Traffic Malware Analysis and Detection Using Ensemble Classifier,” Ain Shams Engineering Journal, vol. 15, no. 12, Art. no. 103134, Dec. 2024, doi:10.1016/j.asej.2024.103134.
[34] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” in Proc. Int. Conf. Learning Representations (ICLR), 2015.
[35] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, “Drebin: Effective and Explainable Detection of Android Malware in Your Pocket,” in Proc. 2014 Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, 2014, doi:10.14722/ndss.2014.23247.
[36] A. H. Lashkari, A. F. A. Kadir, L. Taheri, and A. A. Ghorbani, “Toward Developing a Systematic Approach to Generate Benchmark Android Malware Datasets and Classification,” in Proc. 2018 Int. Carnahan Conf. Security Technology (ICCST), Montreal, QC, Canada, 2018, pp. 1–7, doi:10.1109/CCST.2018.8585560.
[37] F. Wei, Y. Li, S. Roy, X. Ou, and W. Zhou, “Deep ground truth analysis of current android malware,” in Proc. Int. Conf. on Cyber Security and Data Mining (CIDM)/Lecture Notes in Computer Science, 2017, pp. 252–276. doi: 10.1007/978-3-319-60876-1_12.
[38] M. Pinski and A. Benlian, “AI Literacy for Users – A Comprehensive Review and Future Research Directions of Learning Methods, Components, and Effects,” Computers in Human Behavior: Artificial Humans, vol. 2, no. 1, Art. no. 100062, Jan. 2024, doi:10.1016/j.chbah.2024.100062.
[39] Y. Walter, “Embracing the Future of Artificial Intelligence in the Classroom: The Relevance of AI Literacy, Prompt Engineering, and Critical Thinking in Modern Education,” International Journal of Educational Technology in Higher Education, vol. 21, no. 1, Art. no. 15, Feb. 2024, doi:10.1186/s41239-024-00448-3.
[40] S. Wang et al., “Artificial Intelligence in Education: A Systematic Literature Review,” Expert Systems with Applications, vol. 252, Art. no. 124167, Oct. 2024, doi:10.1016/j.eswa.2024.124167.
[41] M. Brundage et al., “The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation,” arXiv:1802.07228, 2018, doi:10.48550/arXiv.1802.07228.
[42] R. Clarke, “Principles for the Responsible Application of Generative AI,” Computer Law & Security Review, vol. 57, Art. no. 106131, Jul. 2025, doi:10.1016/j.clsr.2025.106131.
[43] A. Alzubaidi, “Detecting Android Malware Using Deep Learning Algorithms: A Survey,” Computers & Electrical Engineering, vol. 119, Art. no. 109544, Oct. 2024, doi:10.1016/j.compeleceng.2024.109544.
[44] R. Srinivasan and A. Chander, “Biases in AI Systems,” Communications of the ACM, vol. 64, no. 8, pp. 44–49, Aug. 2021, doi:10.1145/3464903.
[45] M. K. Das, M. Kumar, I. K. Kapil, and R. K. Yadav, “Deepfake Creation Using GANs and Autoencoder and Deepfake Detection,” in Proc. 2023 2nd Int. Conf. Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN), Vellore, India, 2023, pp. 1–6, doi:10.1109/ViTECoN58111.2023.10157962.

[46] A. Hooda and M. Kumar, "Combatting Disinformation and Deepfake: Interdisciplinary Insights and Global Strategies," in Proc. TENCON 2024 - 2024 IEEE Region 10 Conf., Singapore, 2024, pp. 1729–1732, doi:10.1109/TENCON61640.2024.10902954.
[47] I. D. Raji et al., "Closing the AI Accountability Gap: Defining an End-to-End Framework for Internal Algorithmic Auditing," in Proc. 2020 Conf. Fairness, Accountability, and Transparency, Barcelona, Spain, 2020, pp. 33–44, doi:10.1145/3351095.3372873.
[48] Y. Ou, Y. Feng, and Y. Sun, "Towards Accurate and Robust Architectures via Neural Architecture Search," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), pp. 5967–5976, 2024, doi:10.1109/CVPR52733.2024.00570.
[49] Y.-M. Zhang et al., "MOTE-NAS: Multi-Objective Training-Based Estimate for Efficient Neural Architecture Search," Advances in Neural Information Processing Systems, vol. 37, pp. 100845–100869, 2024, doi:10.52202/079017-3198.
[50] J. Crabbé and M. van der Schaar, "Evaluating the Robustness of Interpretability Methods Through Explanation Invariance and Equivariance," Advances in Neural Information Processing Systems, vol. 36, pp. 71393–71429, 2023, doi:10.52202/075280-3127.
[51] G. Liang, S. Michielssen, and S. Fattahi, "Enhancing Performance of Explainable AI Models with Constrained Concept Refinement," in Proc. 42nd Int. Conf. Machine Learning (ICML), Proc. Machine Learning Research, vol. 267, pp. 37299–37338, 2025.
[52] A. Hooda et al., "D4: Detection of Adversarial Diffusion Deepfakes Using Disjoint Ensembles," in Proc. IEEE/CVF Winter Conf. Applications of Computer Vision (WACV), pp. 3800–3810, 2024, doi:10.1109/WACV57701.2024.00377.
[53] Y. Xu et al., "TALL: Thumbnail Layout for Deepfake Video Detection," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), pp. 22601–22611, 2023, doi:10.1109/ICCV51070.2023.02071.
[54] Z. Yan, Y. Luo, S. Lyu, Q. Liu, and B. Wu, "Transcending Forgery Specificity with Latent Space Augmentation for Generalizable Deepfake Detection," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), pp. 8984–8994, 2024, doi:10.1109/CVPR52733.2024.00858.
[55] J. Cheng et al., "Can We Leave Deepfake Data Behind in Training Deepfake Detector?" Advances in Neural Information Processing Systems, vol. 37, pp. 21979–21998, 2024, doi:10.52202/079017-0691.
[56] X. Cui, Y. Li, A. Luo, J. Zhou, and J. Dong, "Forensics Adapter: Adapting CLIP for Generalizable Face Forgery Detection," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), pp. 19207–19217, 2025, doi:10.1109/CVPR52734.2025.01789.
[57] M. Naseer et al., "Obfuscated Malware Detection and Classification in Network Traffic Leveraging Hybrid Large Language Models and Synthetic Data," Sensors, vol. 25, no. 1, Art. no. 202, 2025, doi:10.3390/s25010202.
[58] S. Poornima and R. Mahalakshmi, "Automated Malware Detection Using Machine Learning and Deep Learning Approaches for Android Applications," Measurement: Sensors, vol. 32, Art. no. 100955, 2024, doi:10.1016/j.measen.2023.100955.
[59] Z. Wang, K. Zeng, J. Wang, and D. Li, "FAGnet: Family-Aware-Based Android Malware Analysis Using Graph Neural Network," Knowledge-Based Systems, vol. 289, Art. no. 111531, 2024, doi:10.1016/j.knosys.2024.111531.
[60] A. Joshi et al., "Malware Analysis Using Transformer Based Models: An Empirical Study," in Proc. 21st Int. Conf. Security and Cryptography (SECRYPT), pp. 858–865, 2024, doi:10.5220/0012855100003767.

Authors' Profiles



Mohammad Othman Nassar: Associate Professor of Cyber Security at Amman Arab University (AAU), Jordan. He received his Ph.D. in Computer Information Systems in 2009 and has over 15 years of teaching experience. Since joining AAU in 2010, he has held several leadership roles, including Head of the Computer Information Systems Department, Manager of the eLearning Centre, Manager of the Marketing Department, and Director of the Computer Center. He has published more than 40 scientific research papers and serves on the editorial board of four international scientific journals.

How to cite this paper

Mohammad Othman Nassar, "SECURE-XED: A Unified Explainable CNN Framework with Hybrid Metaheuristic Optimization for Cross-Domain Malware and Deepfake Detection", International Journal of Modern Education and Computer Science(IJMECS), Vol.18, No.5, pp. 37-67, 2026. DOI:10.5815/ijmecs.2026.05.03