

Large Language and Visual Assistant for User-Centric Cybersecurity

Vitalii KRIVKOV

vitalii.krivkov.001@student.uni.lu

University of Luxembourg
Luxembourg

Muriel-Larissa FRANK

muriel.frank@uni.lu

University of Luxembourg
Luxembourg

Andriii SHRAMENKO

andriii.shramenko.002@student.uni.lu

University of Luxembourg
Luxembourg

Pol HÖLZMER

pol.hoelzmer@uni.lu

University of Luxembourg
Luxembourg

Abstract

Users worldwide face ever-evolving scam techniques, such as phishing emails, deceptive advertisements, social engineering, and more. These techniques undermine trust in digital services. Automatic scam detection presents a promising defense, and recent advances in Large Language Models (LLMs) offer new opportunities in this area. LLMs' vast pre-trained knowledge and contextual reasoning capabilities can be leveraged to spot textual and visual scam indicators with minimal task-specific tuning. This paper presents Cybersecurity Visual Assistant (CSVA), a consumer-grade visual assistance system that integrates multimodal LLMs to analyze user screenshots. We describe our prototype implementation, which supports local, server-side and API-based inference using models ranging from lightweight quantized variants to 400B-parameter state-of-the-art architectures. We evaluate performance on a custom dataset of 902 annotated desktop screenshots, balanced between legitimate and scam content.

1 Introduction

As services such as digital identity verification, document exchange, and online banking become more common (e.g. the European Digital Identity initiative [25, 51]), they not only increase trust in digital interactions but also open new avenues for cybercriminals. These services handle highly sensitive personal and financial data, making them prime targets for scam. Despite significant technological advances that have reduced certain risks, online scams remain a pervasive threat, damaging trust in digital communications and making people hesitant to use online tools. Scams cause not only financial losses (exceeding \$1.3 trillion in 2023 alone [10]) but also reputational damage, data breaches, and, in extreme cases, human rights abuses such as trafficking and forced criminality [24, 66].

Simultaneously, modern cybersecurity has matured beyond classic RSA/AES cryptography into an integrated, multilayered defense system that leverages quantum-safe key exchange and AI-driven anomaly detection [43]. This system enables adaptive, data-centric protection against both classical and emerging quantum threats. However, these technical advances are often undermined by the human element [21]. They remain vulnerable to various phishing attacks, scam messages, deceptive advertisements, and social engineering techniques. Scammers manipulate text, images, and website

layouts to bypass traditional detection methods [3]. Empirical research on scam identification shows that awareness training can boost detection rates significantly – up to 72% in Australia [10]. And while such trainings are effective in increasing scam detection, they also inadvertently cause the misclassification of legitimate communications. This makes automatic scam detection a critical research area.

In parallel with these challenges, breakthroughs in Artificial Intelligence (AI) have reshaped the digital landscape. LLMs in particular have shown remarkable capabilities in understanding natural language, generating human-like text, and assisting in complex security tasks such as code vulnerability detection and risk assessment [16]. Recent advances in LLM technology have reduced the computational demands of these models. This progress makes it possible to deploy LLMs on modest hardware while protecting user privacy [12]. Such innovations allow the development of lightweight systems that can operate without compromising sensitive user data, which is a critical consideration for personal cybersecurity tools.

This paper introduces a visual assistance system designed to enhance user-centric cybersecurity. By integrating advanced LLM capabilities with computer vision techniques, this project aims to develop a visual assistance system capable of analyzing screenshots to detect potential scams, ranging from phishing emails and deceptive advertisements to malicious websites and fraudulent chat messages, in near real-time. By leveraging LLMs, we will assess both textual and visual elements in order to allow context awareness, generate risk assessment reports, and issue alerts when scam indicators are identified.

The remainder of the paper is structured as follows. Section 2 reviews the related work in scam detection, LLM-based security, and efficient model deployment. Section 3 details the design and implementation of our visual assistance system, including the methodologies for analyzing both text and image data, and evaluation of the system's performance by comparing local execution with cloud-based solutions.

1.1 Contributions

Throughout the research period, the students held regular in-person meetings to jointly define the project structure and discuss key design decisions. Both contributed substantially to the development of this paper and the underlying prototype. In particular:

- *Andrii Shramenko* led the initial and final prototype development phases and conducted a comprehensive survey of relevant technologies, including large language models and supporting frameworks.
- *Vitalii Krivkov* performed an in-depth literature review of scam-detection methods, integrated draft material into the final manuscript, and executed extensive testing of the completed prototype across multiple LLMs, generating detailed performance statistics, categorizations, and visualizations.

2 Background

In this chapter, we conduct a literature review focusing on existing scam detection technologies [72], emphasizing approaches leveraging LLMs. This literature review pursues three sequential goals:

- (1) To examine how traditional scam-detection systems operate and to analyze their strengths and shortcomings;
- (2) To survey the current state-of-the-art in LLM-based detection, identifying the gains and remaining gaps;
- (3) To combine insights from both streams into a blueprint for a hybrid system that leverages the best of both techniques.

Our research includes studying advanced text-analysis methods, image recognition methods for identifying visual indicators of scam, and context-aware detection strategies that utilize sequences of inputs to detect scam patterns.

2.1 Traditional Scam Detection

Traditional approaches to scam detection have largely focused on well-defined domains such as fraudulent financial transactions [50]. In these scenarios, the problem is typically formulated as an Machine Learning (ML) labeling task (where each record is encoded as a feature vector and classified as fraud or legitimate) or as a graph anomaly detection problem (where unusual connectivity patterns in transactional networks signify potential scams) over transaction databases [13, 9, 11]. These methods generally achieve high accuracy – often exceeding 95% – when applied to structured financial data, where input patterns remain relatively stable [50]. However, such systems are confined to the financial domain and offer little utility in detecting scams that occur outside of structured transactions.

An advancement has been the development of scam detection systems that employ text processing techniques [14]. In these systems, key linguistic features – such as specific keywords (e.g., money, phone numbers) and characteristic sentence constructions (e.g., expressions of urgency) – are extracted from text using regular expressions. The extracted elements are then transformed into queries (for example, using libraries like Pytholog [55]) to a pre-compiled database of private facts. If the query returns a non-zero value (i.e., the analyzed message seems to request private information), then the alert is triggered [14]. Although this rule-based text processing approach represents a significant improvement over methods that rely solely on non-content metadata, its effectiveness is limited to detecting scam messages that conform to known patterns, and it struggles with messages that deviate from these established templates. To address these limitations, researchers have integrated ML and Deep Learning (DL) techniques into scam detection frameworks. In such hybrid systems, models are trained on

extensive databases of scam messages using supervised (where each training example is labeled scam/non-scam) or semi-supervised (where a small set of labeled data is augmented with larger unlabeled corpora to improve generalization) learning [6, 57, 15]. When the input schema remains fixed (i.e., identical fields and datatypes), these models can detect scams with high accuracy; however, their performance can drop dramatically – sometimes to as low as 50% – when scammers employ evasion tactics such as paraphrasing, character swapping, or spacing alterations [57]. This vulnerability to minor textual modifications highlights the inherent challenge of designing robust detection systems in a landscape where scam tactics evolve rapidly.

The challenges inherent in traditional scam detection methods primarily stem from two issues:

- *Availability of data* – real scam data is both limited and heavily imbalanced, as most victims do not report scam incidents [10]. Although synthetic data is sometimes generated to supplement training sets, it often lacks the diversity needed to capture the full spectrum of scam variations. For instance, [57] reports a compilation of roughly 153,551 scam records from various sources; however, these records may not encompass all the nuances and emerging trends in scam tactics.
- *New challenges and attacks* – ML and DL models are fundamentally limited by the scope of the data on which they are trained. If scammers develop new techniques that fall outside the patterns present in the training data, the models are unlikely to recognize these novel tactics, leading to significant drops in detection accuracy described above [57]. This limitation underscores the need for continuous training on updated and specialized datasets and adaptive model architectures to keep pace with the dynamic nature of scam strategies.

2.2 LLM-based solution

LLMs are deep neural networks that have demonstrated remarkable ability to model and generate human language [2, 29, 16, 12]. At their core, an LLM first tokenizes incoming text into discrete units (subwords or word-pieces), then maps each token to a high-dimensional embedding vector. These vectors flow through a stack of interleaved self-attention and feed-forward blocks, where every token “pays attention” to all others via scaled dot-product operations, before being updated. After many such layers, the final hidden state of each position is projected back into the token vocabulary and normalized to yield next-token probabilities [67, 5]. Depending on model size, this architecture can involve up to hundreds of billions of parameters, all learned during pre-training on vast amounts of unlabeled text [2]. In the realm of scam and fraud detection, LLMs bring powerful, contextualized representations, but they also confront many of the same obstacles as classical machine-learning models [29]: adversarial rephrasing of phishing messages, scarcity of labeled examples for novel schemes, and attackers’ continual evolution of tactics. We therefore try to identify, how well an LLM can deal with these challenges, given its inherent limitations, compared to established methods.

Deception itself is multi-faceted: it can be embedded in the wording of a phishing email, hidden in the visual design of a counterfeit website, or spread out over a sequence of actions in a multi-step campaign [58]. To address this diversity, the chapter unfolds in three parts:

- (1) *Textual Scam Detection* – evaluating how well LLMs identify deceptive language compared with classical ML pipelines, and where they still fail;
- (2) *Image Recognition* – assessing multimodal LLMs that unify vision and language to expose visual fraud indicators (e.g., spoofed layouts, branding inconsistencies);
- (3) *Context Awareness* – analyzing whether sequential reasoning over interaction histories further reduces false positives.

2.2.1 Textual Scam Detection. In the context of scam detection, LLMs-based solutions offer significant advantages over traditional ML / DL [22]. Their ability to understand context and infer meaning from ambiguous or multi-modal inputs makes them particularly effective at identifying scam-related language, deceptive narratives, and anomalous communication patterns. Moreover, LLMs require minimal task-specific tuning, easily adapting to new scam tactics [29].

Several studies have demonstrated the potential of LLMs for phishing detection [29, 22, 33]. For example, one paper on LLM-based real-time detection of phone scams [60] outlines a simple yet powerful principle: during a phone call, speech is transcribed to text in real time and then analyzed by an LLM. If the text contains suspicious characteristics, a "Fraud" signal is immediately triggered to alert the user. Remarkably, the ChatGPT 4o model has demonstrated near-perfect (0.99) accuracy in detecting scams across various scenarios [60]– ranging from finance and travel to everyday communications – showcasing the flexibility of this approach, especially when compared to traditional ML models .

The detection capabilities of LLM-based systems can be further enhanced by incorporating additional inputs. For instance, another study on fake website detection [22] describes a tool that takes a URL of a suspicious website as input and analyzes its content in depth. If the website can be confidently identified as fraudulent, the system returns a result; otherwise, the analysis continues using supplementary methods (such as search engine results or Transport Layer Security (TLS) certificate analysis), which can improve detection accuracy by approximately 13% in cases where scam websites are meticulously designed.

Constructing an effective prompt is crucial for guiding an LLM's decision-making process. The work in [22] introduces a comprehensive prompt template that includes:

- *Task setting* – clear definition of the role of the LLM in the system.
- *Scam Characteristics* – descriptions of common indicators of scam activity.
- *Tool Definitions* – instructions for using external tools (e.g., search engine queries) to gather additional information.
- *Analysis Method* – steps that combine reasoning and action to adapt based on new inputs.
- *Output Format* – a standardized structure for the response.

This detailed prompt formulation, which builds on the approach introduced in [71], greatly improves the LLM's ability to articulate its reasoning and adapt its actions – even when deployed on older or lighter models, such as ChatGPT 4 or Mistral 7B respectively.

An essential component of an effective prompt is the inclusion of key scam indicators. This strategy directs the LLM's attention toward specific red flags (e.g., suspicious elements, unusual links, a sense of urgency) that are often associated with fraudulent communications [22]. Based on our literature search, in the Table 1 we summarize the most prevalent scam indicators observed across emails, websites, and messaging platforms [65, 29, 14, 22].

2.2.2 Image Recognition. Beyond text, modern scammers increasingly leverage visual elements to enhance their deceptive tactics [33]. For example, when constructing a counterfeit website or application, attackers often replicate the visual layout and design of established, trusted platforms. They achieve this by copying key design features and then publishing the replica on a slightly altered domain (e.g., by substituting characters with visually similar ones or appending an additional suffix), thereby deceiving users into believing they are interacting with a legitimate entity [18]. Moreover, scammers frequently employ psychological triggers encoded in images (such as promises of easy profit, remarkably low prices, or free giveaways) to further mislead potential victims. Such manipulations are not limited to websites; they are also prevalent in emails, social media posts, and other trusted web applications [33].

Instead of relying on potentially obfuscated HTML or URL patterns, multimodal LLM-based visual assistants analyze the exact screenshots users see (layout, typography, images, and embedded text) so they can flag the same subtle inconsistencies (off-brand fonts, low-resolution logos, text–image mismatches) that a trained human reviewer would [45]. Traditionally, LLMs have employed Optical Character Recognition (OCR) to extract text from images [61]; however, OCR-based approaches focus solely on textual information and ignore other critical visual information [61]. Recent advances extend this capability through the use of dedicated visual encoders, as detailed in [70] and [56]. These encoders transform raw pixel data into multiple high-dimensional embeddings (visual tokens). These embeddings are high-dimensional vectors – often 768 dimensions or more – that encapsulate not only text but also features of objects, scenes, and their interrelationships, representing the image content. Later, visual tokens are integrated with the LLM language processing capabilities, providing a deep, human-like interpretation of both visual and textual content [37, 70, 56]. Such models are called multimodal LLMs [33].

In the context of scam detection, multimodal LLMs combine textual analysis with computer vision techniques to evaluate images, screenshots, and website layouts. Systems such as the one described in the paper [33] leverage these capabilities to perform brand identification by analyzing both the visual appearance (e.g., logos, color schemes, layout consistency) and the textual content. By integrating these data streams through carefully engineered prompts, the LLM can provide a comprehensive assessment of whether a webpage or message is likely to be a scam [22, 33].

2.2.3 Context Awareness. While many approaches to scam detection focus solely on individual snapshots or messages, recent research highlights the importance of incorporating *context* – a series

Key Indicator	Description
Email Scam Indicators	
Suspicious Sender Address	Sender address mismatch or unknown
Unusual Links	Redirects to dubious or unexpected sites
Sense of Urgency	Pressures for immediate action
Lack of Personalization	Generic greetings or sign-offs
Website Scam Indicators	
Low Prices / Free Claims	Unrealistically low prices or freebies
High Payout Promises	Promises of unusually large sums
Non-existent Company Info	Fake or ambiguous company details
Suspicious URL	Unusual URL patterns or IP-based addresses
Branding Inconsistencies	Inconsistent logos or color schemes
Image Manipulation Artifacts	Signs of tampering or low-quality images
Text-Image Mismatch	Discrepancies between text and visuals
UI/UX Irregularities	Unusual or mismatched interface elements
Messenger Scam Indicators	
Unsolicited Messaging	Generic messages from unknown contacts
Urgent Financial Requests	Demands for immediate transfers
Sensitive Info Requests	Asks for personal or sensitive details

Table 1: Common Key Scam Indicators [65, 29, 14, 22]

of user actions or system states preceding the current event – to improve accuracy. In credit card fraud detection, for instance, analyzing sequential transaction patterns significantly outperforms static methods, reducing false positives and uncovering long-term malicious campaigns [31]. Following this concept to scam detection, visual-scam detector should likewise track a user’s screenshot history rather than treating each capture in isolation.

A naïve approach might be to feed all screenshots to the model directly. This procedure is straightforward, yet it involves unnecessary re-computations. Given that many of these screenshots have already been processed (i.e., encoded into an LLM-readable format during previous interactions), encoding them again is wasteful. A more efficient solution is to store the already encoded images as fixed-length vector embeddings, which can then be aggregated over time to form a context representation without recomputing previously evaluated screenshots. Although this method provides a detailed representation of each image, it may overload the LLM’s context window [23]. Specifically, each image is split into small patches (e.g., 14×14 pixels), and each patch is encoded into a high-dimensional vector. Consequently, hundreds of such vectors per image are presented to the LLM [4, 37], rapidly exhausting both available memory and context length of the LLM [23].

A potential alternative is to replace the image data with a detailed textual description generated by an LLM, thereby eliminating the need to re-process visual information. Although this method is easily integrated into existing multimodal LLMs, it significantly increases the prompt size for each request: a single comprehensive image description can exceed 300 words (each word can be between 1 and 3 tokens), thereby adding thousands of words to the LLM’s input as a context. This can lead to degraded performance, as the model struggles to manage hundreds or thousands of tokens while still accurately discerning subtler scam indicators [23].

Instead of storing sets of hundreds of visual tokens or large textual descriptions, a more balanced approach is to compress each screenshot into a single (or small set of) high-level embeddings – similar to the [CLS] token in many vision-language models [37]. This collection of “global” vectors can then be further reduced, if necessary, into one or two compact “context summary” vectors that still retain essential spatiotemporal information. By applying targeted compression steps at each stage, one can efficiently handle multiple frames while avoiding the computational and token constraints. It is also crucial to consider the content of the image being compressed. In our application, a large portion of the screenshots contain text, and many of the indicators in Table 1 are textual by nature. Solely relying on summary vectors would obscure such textual information, so we additionally extract and retain any text found in the image before applying compression mechanisms. Consequently, the following methods can help compress multi-frame embeddings effectively:

Global Pooling: A straightforward strategy is to apply global pooling across the embeddings. Originally introduced in [39], global average pooling replaces expensive fully connected layers with an operation that averages the final layer’s feature maps. In a multimodal context, we could compute a single global embedding per image (e.g., a [CLS] token [37, 4]), then apply global average pooling across N screenshots to produce a single vector summarizing the “mean scene.” Alternatively, global max pooling can be employed to emphasize salient or critical features. Although these pooling methods are computationally simple, they merge features indiscriminately, risking the loss of finer temporal or spatial cues. Consequently, global pooling is best suited for capturing high-level context in scenarios where local details are less crucial.

Learned Linear Compression: A more nuanced alternative is learned linear compression [34]. Rather than merely averaging embeddings, we concatenate multiple screenshot-level [CLS] tokens and pass

them through a small feed-forward network trained to yield a single d -dimensional vector. This approach can learn to weight certain screenshots or features more than others, preserving key spatiotemporal details while filtering out repetitive or irrelevant information. Conceptually, it extends the basic pooling operation by incorporating trainable parameters that adapt to the domain of scam detection.

Other Techniques: Research in video-language modeling has introduced even more advanced strategies for summarizing long sequences. Temporal transformers and Q-Formers [36, 35, 73] select relevant tokens from each frame, while memory-consolidation approaches [62] merge tokens once the model reaches its token limit, discarding or compressing redundant information. These methods can retain more fine-grained information at the cost of additional architectural and computational complexity.

In this research, we concentrate on more lightweight solutions – specifically Learned Linear Compression, Global Pooling, and Textual Representation – because our system must run locally under strict time constraints. Although more powerful methods could potentially achieve higher accuracy, they would be impractically slow, given the substantial computations required before prompting the LLM.

3 Prototype

In this chapter, we discuss how we design and implement the prototype that integrates the identified scam detection techniques, leveraging LLMs such as Vicuna, Gemma, and Mistral for text and image-based scam detection. The system will be developed as an end-to-end solution capable of processing screenshots, analyzing their content, and providing scam risk assessments. The prototype will be tested to assess its accuracy and performance under various conditions. In addition, we will evaluate the feasibility of running models locally versus using cloud-based solutions, ensuring scalability and efficiency.

3.1 Overview of Models

This research leverages the Ollama platform [48], which enables easy integration of numerous open-source machine learning models. As of the date of this study, Ollama supports 172 open-source models, including 12 multimodal variants [48]. From this extensive selection, we carefully curated a subset optimized for accuracy, computational efficiency, and suitability for scam detection on consumer-grade hardware.

During our selection process, models were assessed based on their ability to run effectively on typical consumer hardware, specifically focusing on minimal RAM usage and execution time. Performance benchmarks were primarily conducted using an Apple M2 processor as a reference system. Given the nature of scam detection, involving both visual and textual elements, we prioritized multimodal models. For scenarios requiring higher customization, we also evaluated combinations of text-only models integrated with OCR and specialized visual encoders. Models were selected based on documented accuracy in tasks relevant to scam detection, including text recognition, visual anomaly detection, and context-aware reasoning.

The chosen models and their relevant characteristics are summarized in Table 2. Each model and its selection rationale is further described below.

3.1.1 Optical Character Recognition.

- EasyOCR [27] was selected due to its balance between high accuracy and minimal computational overhead. Compared to alternatives like Tesseract [63], EasyOCR demonstrated superior text extraction accuracy [68]. Its lightweight design (24.6 million parameters) allows quick execution even on modest hardware, making it ideal for real-time applications.

3.1.2 Visual Encoders.

- ViT-GPT2 [46] was chosen for its compact size and rapid inference capabilities. ViT-GPT2 provides an efficient baseline for encoding visual content into a textual format easily interpretable by LLMs.
- BLIP [37] is also lightweight but optimized specifically for vision-language tasks. BLIP offers contextual understanding, capturing subtle visual anomalies that might indicate scam content.
- BLIP-2 [36] was selected for its higher precision in complex visual recognition tasks. Although more demanding computationally, BLIP-2 significantly enhances accuracy in image encoding, justifying its use in scenarios where increased processing overhead is acceptable.

3.1.3 Text-only Language Models.

- Gemma3:1b [20], due to its minimal resource requirements, provides reliable language processing capabilities suitable for real-time systems or devices with limited computational resources.
- Mistral 7B: [28] represents a balance between computational demands and language processing robustness, providing more nuanced text interpretation and higher accuracy in detecting linguistic indicators of scam content.

These models were manually paired with the visual encoders and OCR tools to create custom multimodal pipelines.

3.1.4 Multimodal Models.

Multimodal models inherently support simultaneous text and image inputs, providing a unified, integrated analysis. Four models were specifically selected:

- Moondream 2 [69] is the smallest multimodal model available in Ollama. It effectively handles small-complexity visual tasks. Its relatively low computational requirements make it ideal for everyday consumer devices.
- Gemma3:4b [20] offers a balance between accuracy and resource consumption. It performs well on mid-range devices, enhancing visual and textual contextual comprehension.
- LLaVA [8] provides performance comparable to Gemma3:4b. It was selected to enable a direct comparison between the two models. It is particularly effective in multimodal reasoning and detecting subtle inconsistencies across visual and textual data, allowing us to evaluate which model better suits the requirements of scam detection tasks on mid-range hardware.

Model type	Resource demand	Model name	Total parameters
OCR	Low ^a	EasyOCR	24.6M
Image Encoders	Low	ViT-GPT2	239M
	Low	BLIP	247M
	Medium ^b	BLIP-2	2.7B
Text Only	Low	Gemma3:1b	1B
	Medium	Mistral 7B	7.3B
Multimodal	Low	Moondream 2	1.8B
	Medium	Gemma3:4b	4B
	Medium	LLaVA	7B
	High ^c	Gemma3:27b	27B
	Ultra-high ^d	Llama 4 Maverick	400B

Table 2: Overview of models used [27, 46, 37, 36, 19, 28, 69, 8, 44]

Note: The “Resource demand” column reflects real-world performance benchmarks conducted on a MacBook Air M2 with 8GB of RAM.

^a *Low*: These models typically require 5–15 seconds per input for inference, and are well-supported on the MacBook Air M2. They may utilize a moderate portion of available memory, but remain comfortably within the device’s operational limits.

^b *Medium*: These models may take from 15–120 seconds per input to produce results. While still executable on the MacBook Air M2, their higher computational requirements result in noticeable delays and heavily limit multitasking during use.

^c *High*: These models generally cannot be run on a MacBook Air M2 (8 GB RAM). They are suitable only for workstations with significantly greater resources (e.g., high-end desktops or servers).

^d *Ultra-high*: Models such as Llama 4 Maverick are well beyond the capabilities of consumer-grade hardware, requiring advanced GPU clusters or specialized cloud infrastructure. They are included primarily for theoretical benchmarking.

- Gemma3:27b [20] is selected primarily for high-accuracy scenarios. It demonstrates exceptional sensitivity to sophisticated scam indicators but requires significant computational resources, limiting its practical deployment to powerful devices or cloud environments.
- Llama 4 Maverick [44] is representing the state-of-the-art in multimodal processing. It was chosen for theoretical benchmarking due to its high accuracy and depth of context understanding. Given its exceptionally high computational demands, it serves mainly as a reference model for comparative evaluations rather than practical everyday use.

3.1.5 Quantization. Even relatively small models can exceed the memory capacity of a typical PC: for example, the 4 billion-parameter Gemma3 requires at least 16 GB of RAM when using full 32-bit precision [19]. Quantization techniques mitigate this by representing model weights in lower-precision formats, trading a reduction in accuracy for substantially lower memory footprints [30]. In this work, both Gemma3:4b and Gemma3:27b were quantized to 4-bit integers (INT4), yielding an eightfold decrease in parameter storage. Despite this, the quantized Gemma3:4b still demands up to 6 GB of RAM during inference. [19].

3.2 Development

3.2.1 System Architecture Overview. The CSVA project implements a modular pipeline for real-time scam detection using multimodal AI analysis. The system is designed around a flexible architecture

that separates concerns into distinct components, enabling experimentation with different models and processing approaches. The core philosophy emphasizes offline deployment capabilities while maintaining the ability to test various detection strategies systematically.

3.2.2 Core Pipeline Implementation. The ScamEvaluatorPipeline class serves as the central coordinator, implementing a dual-path processing approach that routes inputs based on the processor output type. For text-only models like Mistral, the system follows an OCR-based path where screenshots are processed through text extraction before being analyzed. For multimodal models like LLaVA or Gemma 3, the system uses an inherent processing path that preserves visual information alongside textual data. The pipeline begins with image analysis through configurable ImageProcessor implementations, followed by context retrieval from the memory system. The system then constructs appropriate prompts based on the processing path, evaluates them through the LLM, and manages resulting data through logging and memory updates. This design enables efficient experimentation with different processing strategies while maintaining a consistent interface for the rest of the system.

3.2.3 Image Processing Components. The processor system implements a plugin-based architecture with specialized components for different types of data processing. The OCRIImageProcessor provides multi-language support through a configurable language

list and implements sophisticated text extraction with confidence scoring. It uses EasyOCR [27] as the underlying OCR engine, which operates entirely within the Python environment without requiring external system dependencies. The OCR processor includes advanced image preprocessing capabilities using OpenCV, including noise reduction through cv2.fastNlMeansDenoising, contrast enhancement through Contrast Limited Adaptive Histogram Equalization (CLAHE), and Gaussian blur for noise reduction. The processor filters results by confidence threshold and provides debug capabilities for troubleshooting extraction quality.

The InherentImageProcessor is designed for multimodal models. This processor preserves the original image path and allows the LLM to analyze both visual elements and any extracted text, providing a more comprehensive analysis capability.

3.2.4 Language Model Integration. The OllamaLLM class provides a sophisticated wrapper for model interactions. It supports a range of text-only and multimodal models through a unified interface. The implementation handles message formatting with role-based content organization and supports image embedding in messages for multimodal models. The client management system implements on-demand initialization with configurable connection parameters. The system supports various model configurations including Mistral for text-only processing, LLaVA for multimodal analysis, and Gemma3 variants for different performance characteristics. Each model can be configured with specific system prompts, temperature settings, and generation parameters to optimize for different use cases.

3.2.5 Prompt Engineering. We developed three distinct prompt templates. Each of them is optimized for a different balance of brevity, domain knowledge, and reasoning depth.

1. *Minimal Prompt* – A lightweight template that minimizes token usage and dependency on external context.

You are acting as a professional scam and security threat detection system.

You are analyzing a screenshot from a user's screen. Your task is to evaluate whether the current screenshot contains any indications of scams, fraud, or malicious behavior to assist your judgment.

Final Answer Format:

```
{  
  "reasoning": "<...>",  
  "scam_indicators": "[indicators]",  
  "scam_type": "<type of scam, if applicable>",  
  "Scam Probability": <integer from 0 to 5>  
}
```

2. *A prompt with a list of common scam indicators provided* – builds on the minimal template by embedding a comprehensive list of common scam indicators from the table 1 along with a detailed probability scale.

You are acting as a professional scam and security threat detection system. You are analyzing a screenshot from a

user's screen. Your task is to evaluate whether the current screenshot contains any indications of scams, fraud, or malicious behavior to assist your judgment.

LIST POSSIBLE SCAM INDICATORS:

Email Scam Indicators:

- Suspicious Sender: Sender address mismatch or unknown
- Unusual Links: Redirects to dubious or unexpected sites
- Sense of Urgency: Pressures for immediate action
- Lack of Personalization: Generic greetings or sign-offs

Website Scam Indicators:

- URL mismatch: URL doesn't match with the web site
- Suspicious URL: unusual patterns, attempts to seem like a trusted URL
- Low Prices / Free Claims: Unrealistically low prices or freebies
- High Payout Promises: Promises of unusually large sums
- Non-existent Company Info: Fake or ambiguous company details
- Branding Inconsistencies: Inconsistent logos or color schemes
- Image Manipulation: Signs of tampering or low-quality images
- Text-Image Mismatch: Discrepancies between text and visuals
- UI/UX Irregularities: Unusual or mismatched interface elements

Messenger Scam Indicators:

- Unsolicited Messaging: Generic messages from unknown contacts
- Urgent Financial Requests: Demands for immediate transfers
- Sensitive Info Requests: Asks for personal or sensitive details

Interface Scam Indicators:

- Confirmshaming: Guilt-tripping users into making a specific choice
- Fake Scarcity / Fake Urgency: Creating false scarcity or urgency ("Only 3 left in stock")
- Nudge: Emphasizing a preferred option visually while de-emphasizing others
- Obstruction: Designing unnecessary hurdles to prevent opting out or cancelling
- Forced-Action: Requiring completion of an unrelated action before proceeding
- Pre-Selection: Defaulting checkboxes or options to "on"
- Visual Interference: Hiding critical information via low contrast or small text
- Jargon: Using complex or misleading language to confuse users
- Hidden Subscription: Not clearly stating that signup enrolls in a service
- Sneaking Hidden Costs: Revealing extra fees only at final

checkout

- Disguised Ads: Embedding ads disguised as normal UI elements
- Trickwording: Phrasing that misleads or reverses opt-in and opt-out actions

Scam Probability Scale:

- 0 - No scam signs: Trusted, verified source; no signs of scam.
- 1 - Uncertain: No immediate red flags, but not enough data to confirm safety.
- 2 - Low scam probability: Weak anomalies; likely safe, but caution advised.
- 3 - Moderate scam probability: Suggests possible scam/fraud. Review is needed. ANY popups about prizes, rewards, or milestones MUST be at least this level.
- 4 - High scam probability: Strong indicators of malicious or scam-related activity.
- 5 - Critical scam probability: Confirmed scam, malware, or phishing attempt. Urgent action required.

Answer Format:

```
{
  "reasoning": "<...>",
  "scam_indicators": "[indicators]"
  "scam_type": "<type of scam, if applicable>",
  "Scam_Probability": <integer from 0 to 5>
}
```

3. Reasoning prompt – Extends the previous version by defining available tools (e.g., image-to-text, URL verifier, window-type classifiers) and prescribing a chain-of-thought format. This template guides the model through repeated Thought–Action–Observation cycles before producing its final answer.

You are acting as a professional scam and security threat detection system. You are analyzing a screenshot from a user's screen. Your task is to evaluate whether the current screenshot contains any indications of scams, fraud, or malicious behavior to assist your judgment.

LIST POSSIBLE SCAM INDICATORS:

... Similar to the previous prompt ...

Scam Probability Scale:

... Similar to the previous prompt ...

Tools Available:

- Image-to-Text (input -- image): A tool which produces a detailed textual representation of a given image.
- Verify URL (input -- URL string): A tool which analyzes a URL of a web-page.
- IsMessenger (input -- textual representation of an image): A tool which tells, whether a current open win-

dow is a messenger

- IsBrowser (input -- textual representation of an image): A tool which tells, whether a current open window is a browser
- IsEmail (input -- textual representation of an image): A tool which tells, whether a current open window is an email client
- Verify Scam Indicators (input -- textual representation of an image): A tool which verifies, whether the text contains one or more indicators of scam form the LIST OF POSSIBLE SCAM INDICATORS:

Use the following structured reasoning format:

Question: Does this screenshot indicate any scam or security threat?

Thought: [Reason about the image]

Action: Visual Analyzer

Action Input: Current screenshot

Observation: [What Visual Analyzer detects visually] (Repeat Thought / Action / Observation up to 5 times if needed)

Thought: I now know the final answer.

Action: Final Answer.

Final Answer Format:

```
{
  "reasoning": "<...>",
  "scam_indicators": "[indicators]"
  "scam_type": "<type of scam, if applicable>",
  "Scam_Probability": <integer from 0 to 5>
}
```

3.2.6 Configuration Management. The system uses a YAML-based configuration system that allows for modification of prompt templates, and selection of different model configurations. The configuration system supports multiple profiles, each defining specific combinations of image processors and LLM configurations. The active profile determines which processing pipeline is used, allowing for easy switching between different experimental setups. Each profile can specify custom system prompts, model parameters, and processing options tailored to specific use cases.

3.2.7 Captioning and Multimodal Capabilities. The system includes specialized captioning components for enhanced visual understanding. The FastCaptioner class provides rapid image captioning capabilities using BLIP models, while the ViT-GPT2Captioner offers alternative captioning approaches using Vision Transformer and GPT-2 architectures. These components enable the system to generate textual descriptions of visual content, which can then be analyzed alongside extracted text for more comprehensive scam detection. The captioning system operates independently of the main OCR pipeline, providing additional context for models. This approach allows the system to capture both explicit textual content through OCR and implicit visual information through captioning, creating a more robust analysis framework.

3.2.8 Testing and Evaluation Framework. The project includes an integration testing of the full pipeline, statistical evaluation of the results, visualization methods, statistical significance testing and an API-driven version of the core functionality (to provide an upper-bound reference). The full-test function systematically processes both legitimate and scam datasets, providing timing information and classification results for each image.

3.2.9 Context Awareness Implementation. During initial prototyping, the context-awareness component proved infeasible to integrate. First, Ollama library did not support transmitting raw embedding vectors at the time of writing, precluding any image-level compression without modifying its source code or introducing a custom adapter. Second, a purely textual encoding of multiple screenshots generated excessively large and noisy prompts that exceeded the effective attention span of smaller models. For instance, when four images (one scam, three legitimate) were encoded as text in a five-shot prompt, Gemma3:4b failed to identify the scam screenshot, despite correctly classifying each image in isolation. Because the added context did not improve detection accuracy and increased per-input processing time substantially, we discontinued further development of this feature.

3.2.10 Extensibility and Future Directions. The modular architecture supports several enhancement strategies under investigation, including global pooling across embeddings, temporal alignment weighting, and selective frame summarization. These techniques aim to improve model accuracy and response consistency without exceeding resource limits. The system’s design allows for easy integration of these improvements as they are developed. The architecture’s flexibility is particularly valuable given that the context awareness implementation, while not yielding immediate improvements in detection accuracy, can be easily modified or replaced with alternative approaches. This adaptability is crucial for a system that needs to evolve with changing scam patterns and detection requirements.

3.3 Dataset

As a number of other papers have mentioned, the availability and quality of data remain significant challenges within the scam detection field [22, 33, 45]. Previous research has addressed this issue to some extent by constructing custom datasets; however, these datasets often become quickly outdated, or formatted in a way unsuitable for our particular use case. Indeed, obtaining genuine scam data remains inherently challenging, as approximately 70% of scam incidents go unreported [10].

While large-scale, publicly accessible phishing databases (e.g., PhishTank [52], OpenPhish [49]) and platforms like Kaggle [64] or Hugging Face [40] offer various datasets for phishing emails and URLs, our project specifically requires visual data – realistic desktop screenshots containing scam indicators. Such screenshots must depict real user scenarios, including open browser windows, email clients, messenger applications, or other desktop applications potentially displaying scam content embedded within an otherwise legitimate desktop environment. To our knowledge, such a dataset, which is balancing realistic user context with annotated scam and legitimate content, does not currently exist at scale. As a result, we

undertook the task of constructing a new dataset tailored to the requirements of vision-based scam detection.

Given the focus of our project on visual analysis, we required a dataset comprising desktop screenshots, including cases where scam content is present within a broader digital context. Examples include a screenshot with an active browser window displaying a phishing page, or an email client with a scam message,. Consequently, after a comprehensive review of publicly available resources, we found that none offered both the breadth and realism required.

3.3.1 Legitimate Images. For legitimate images, our goal was to capture the diversity of authentic user environments across platforms and applications. We relied on existing publicly available datasets, such as Desktop UI [7] and OmniACT [32]. These datasets contain extensive sets of screenshots capturing typical desktop environments across various operating systems, including Windows, macOS, and Linux. These screenshots display everyday software in realistic scenarios, highlighting a range of interface elements, window layouts, and typical programs. Examples include widely used programs like Microsoft Office applications, Adobe Creative Suite, various web browsers, app stores, and system settings dialogs.

We believe these images provide an accurate cross-section of what can be considered “normal” desktop activity, and thus serve as a robust baseline for the detection of anomalies or scam indicators. No additional filtering was applied except for ensuring that all images contained no visible scam content, based on manual review.

In addition to these datasets, we compiled roughly 100 real browser screenshots featuring legitimate web-pages drawn from the Ahrefs’ 1000 most-visited sites [1].

3.3.2 Scam Images. Two projects offered a starting point:

- CIRCL [17]: contains approximately 460 screenshots of phishing sites, scam e-mails, and malware pop-ups. Although helpful, the set is small, many images lack real-world context, and duplicates are common. After manual quality filtering, only about 150 screenshots proved usable.
- ScamFerret [22]: provides pairs of pre-rendered images and .html files for scam websites, with supplementary metadata (e.g., URLs). Similarly to CIRCL dataset, these images fail to replicate actual user scenarios, since they lack browser environment. Although the collection spans a wide range of scam types and would be valuable for future model evaluations.

3.3.3 Dataset Creation. In this project, we decided to focus primarily on phishing web page screenshots. This decision is motivated by evidence suggesting that more than 60% of email and messenger scams involve links directing victims toward fraudulent websites [54]. Consequently, realistic representations of these phishing websites within the user’s actual desktop environment with browser interfaces and realistic URL displays were essential.

Initially, we considered a straightforward approach: a Python script designed to sequentially open stored HTML pages from the ScamFerret dataset in a local browser environment and then capture full-screen screenshots (via macOS’s built-in shortcut Cmd+Shift+3). Although functional, this method failed since the browser URL bar

displayed a local file path rather than the authentic scam URL, severely diminishing the realism of the screenshots and obscuring a crucial scam indicator (the suspicious URL). Moreover, the HTML pages from existing datasets were often devoid of style sheets and scripts, resulting in overly simplistic and unrealistic web page representations.

To address these issues, we designed a custom data collection pipeline:

- (1) Running on macOS 15.1, the script automatically launched a real instance of one of the four major browsers (Chrome and Edge via PLaywright [53]; Safari and Firefox via Selenium [59]) to increase diversity and avoid browser-specific artifacts.
- (2) In this browser window, up to five tabs were opened from a predefined list of 100 popular, legitimate websites, simulating a realistic browsing session.
- (3) The target URL from a .txt was loaded in one *target* tab.
- (4) The script simulated user actions, such as scrolling down the page by a random amount, to trigger any deferred or dynamic scam content.
- (5) Full-screen or fill-screen modes were used to capture as much visual context as possible, and AppleScript was employed to take screenshots (Cmd+Shift+3) at the system level.
- (6) Alongside the screenshot, the script saved a web archive of the target page and a JSON file containing structured information (headers, location, titles, meta tags, forms, links, etc.).
- (7) Each browser session was closed after capture.

The script design intentionally mimics genuine user activity as closely as possible, producing highly realistic, diverse images that incorporate authentic scam indicators within desktop scenarios.

3.3.4 Dataset Structure. Over the course of a month, we crawled a diverse set of legitimate and malicious URLs. Because malicious sites tend to be short-lived, roughly 85% of the URLs we collected were already blocked when our crawler reached them. Even after applying response-type filters, many samples proved invalid. Every remaining sample was therefore inspected and verified manually.

Using this systematic approach, we created an extensive dataset capable of evaluating our prototype’s efficacy comprehensively. Table 3 summarizes the distribution:

- *Desktop* – screenshots of the desktop showing system settings, the file manager, or a terminal session.
- *App* – captures of standalone applications (both built-in and third-party) in use.
- *Website* – browser windows displaying either legitimate web-pages or scam content, with the full browser interface.
- *Email* – screenshots of e-mail clients, either showing individual messages or message lists, labeled as scam or legitimate.
- *Website (no URL bar)* – webpage captures where the surrounding browser interface is cropped out.
- *Ransom* – screens displaying an active ransomware lock-screen or ransom note.

In total, the dataset comprises 902 images, evenly split between legitimate (451) and scam (451) categories. The distribution across categories includes realistic desktop screenshots, authentic application windows, genuine websites and applications, phishing emails, ransomware notifications, scam websites with clearly visible URL bars, and a smaller subset without visible URLs, simulating scenarios where URL-based indicators are absent. This dataset, reflecting diverse and authentic scenarios, addresses the inherent limitations identified in previously existing datasets and provides a robust foundation for evaluating and finetuning the proposed visual assistance system for scam detection.

3.4 Testing and Evaluation

Experiments were conducted on the full dataset: each test run processed 451 legitimate and 451 scam images. The table 4 summarizes all evaluated models and their results. The following metrics were recorded per model: accuracy, precision, recall (true positive rate), specificity (true negative rate), F_1 -score. Table 5 then summarizes each model’s operational performance, including the number of failed requests, and the average execution time per request. A sample input and output are displayed in the following figures: 14a - input images, 14b - output of OCR and image captioning engines, 15 - output of various models.

3.4.1 Image Captioning. We have used BLIP, BLIP-2 and Moon-dream 2 for image captioning. While effective in general cases, these solutions fall short for a specific task of describing screenshots. As displayed in the figure 14b, their captions omitted critical UI elements (e.g., URL bar text, button labels), making them unsuitable for our pipeline.

3.4.2 Text-Only. As displayed in figure 14b, initial tests revealed that OCR produced excessive noise (misrecognized characters and separate fragments) which overwhelmed the LLM and caused it to label nearly every image as a scam. To address this, we applied contrast enhancement and thresholding to the screenshots, and filtered OCR output to remove low-confidence. Under these optimized conditions, Mistral 7B correctly classified 100% of legitimate images and approximately 10% of scam images, yielding an overall accuracy of 55.3%. Although its recall remains low, this configuration minimizes false positives and may be preferable in scenarios where avoiding false alarms is critical.

3.4.3 Multimodal models. Two mid-tier multimodal models, such as Gemma3:4b and LLaVA 7B, were evaluated on the MacBook Air M2 (8 GB RAM). Despite its larger parameter count, LLaVA 7B operated with memory requirements comparable to those of Gemma3:4b. However, LLaVA 7B repeatedly failed to process the extended system prompt required for chain-of-thought reasoning, resulting in 257 failed inference requests and a specificity of only 36.5%. In contrast, Gemma3:4b adhered more reliably to the prompt structure, successfully activating the ReAct [71] reasoning mechanism and achieving higher overall accuracy. A side effect of this reasoning loop was occasional repetition of identical or similar outputs, which ultimately exhausted the 128K token limit and produced 24 failed requests.

Type	Legitimate				Scam			
Category	Desktop	Website	App	Email	Ransom	Website	Website (No URL Bar)	Email
Count	62	231	137	21	17	285	132	17
Total	451				451			

Table 3: Overview of Collected Dataset Images

3.4.4 High-Performance. Experiments with Gemma3:27b were conducted on a dedicated high-performance workstation accessed via an Secure Shell (SSH)-tunneled Ollama server [19]. The remote machine was equipped with 8x Tesla V100-SXM2, 128GB VRAM and 1800GB RAM. Due to budget constraints, evaluation was limited to the 451-image scam subset. Under these conditions, Gemma3:27b achieved a recall that was 25% higher than Gemma3:4b, with only seven failed requests, most likely caused by concurrent processes and prototype bottlenecks. Although this deployment results in the highest operational cost, it also delivers maximal data security (since the instance and the model were under our direct control) and seamless integration with the existing prototype, while substantially enhancing detection sensitivity.

3.4.5 API. To assess a state-of-the-art model without incurring local deployment overhead, Llama 4 Maverick (400B parameters) was accessed via Lambda AI’s inference API. This configuration enabled rapid experimentation at minimal setup cost, and the model achieved an overall accuracy of 83% in scam detection.

In a parallel evaluation, Gemma3:27b was deployed at full 32-bit precision through DeepInfra’s inference API. With a list of scam indicators from the table 1 provided, Gemma3:27b outperformed

Model	Type	AET(s) ^a	Failed
Gemma4-qt	N(1)	42.99 ^b	0
Gemma4-qt	R	100.23 ^b	24
Gemma4-32bit	R	6.56 ^d	5
Llava7	N(1)	42.12 ^b	1
Llava7	R	43.32 ^b	257
Mistral7	N(1)	46.66 ^e	1
Gemma27-qt	N(1)	11.48 ^c	7
Gemma27-32bit	N(1)	7.69 ^d	0
Gemma27-32bit	N(2)	6.05 ^d	0
Gemma27-32bit	R	8.82 ^d	0
Mav	N(1)	2.79 ^d	0
Mav	N(2)	3.20 ^d	0
Mav	R	5.87 ^d	0

Table 5: Model performance metrics.

^a Average Execution Time in seconds.

^b on Apple MacBook Air 2023, M2, 8GB RAM.

^c 8x Tesla V100-SXM2, 128GB VRAM, 1800GB RAM.

^d via Inference API.

^e on ROG Zephyrus G14, AMD Ryzen 7 6800HS, AMD ATI Radeon RX 6650 XT, 16GB RAM

Model	Type ^a	Acc	Pre	Rec	Spe	F1
Gemma4-qt ^d	N(1) ^b	0.639	0.888	0.319	0.960	0.469
Gemma4-qt	R	0.675	0.840	0.423	0.921	0.563
Gemma4-32bit	R	0.788	0.729	0.917	0.658	0.812
Llava7	N(1)	0.648	0.864	0.352	0.944	0.500
Llava7	R	0.544	0.564	0.696	0.365	0.623
Mistral7	N(1)	0.553	1.000	0.106	1.000	0.192
Gemma27-qt	N(1)	0.673	1.000	0.673	0.000	0.805
Gemma27-32bit	N(1)	0.891	0.846	0.955	0.827	0.897
Gemma27-32bit	N(2) ^c	0.841	0.773	0.966	0.716	0.859
Gemma27-32bit	R	0.895	0.847	0.964	0.827	0.902
Mav	N(1)	0.803	0.942	0.645	0.960	0.766
Mav	N(2)	0.799	0.950	0.632	0.967	0.759
Mav	R	0.830	0.933	0.712	0.949	0.808

Table 4: Model results comparison.

^a “Type” stands for a “Test Type” in which “R” means that reasoning is activated, and “N” means that the test was conducted without reasoning.

^b ‘N(1)’ stands for no reasoning, the list of scam indicators is provided.

^c ‘N(2)’ stands for no reasoning, minimal prompt, no list of scam indicators.

^d “qt” stands for “quantized”

Llama 4 Maverick, attaining an accuracy of 89.5%, despite its lower memory footprint and smaller total parameter count.

We also tested Gemma3:4b at full 32-bit precision via the same DeepInfra API. Even this smaller model delivered a strong performance, achieving 78% accuracy with only 5 failed inference requests. This result demonstrates that, given sufficient hardware resources, compact LLMs at higher precision can approach the effectiveness of much larger models. Consequently, deployment choices should consider either a high-parameter model in a quantized format or a smaller model at full precision, as both configurations are viable for robust scam detection.

3.5 Reasoning

ReAct chain-of-thought prompting was integrated into Gemma3:4b, Gemma3:27b, and Llama 4 Maverick to quantify its impact on classification accuracy, recall, and specificity.

3.5.1 Statistical Significance. To determine whether there is a significant difference between reasoning being activated or not, we have conducted a statistical significance test. We have used McNemar’s test [41]. McNemar’s test is a non-parametric test for paired binary outcomes. It evaluates whether the proportions of discordant pairs differ significantly between two classifiers.

McNemar’s test consists of the following steps:

- (1) For each of the N paired observations (here, images), record for Test A and Test B whether the prediction was correct (\checkmark) or wrong (\times).
- (2) Construct the 2×2 contingency table:

	B correct (\checkmark)	B wrong (\times)	
A correct (\checkmark)	a	b	
A wrong (\times)	c	d	

- (3) Compute McNemar’s test statistic (without continuity correction):

$$\chi^2 = \frac{(b - c)^2}{b + c}.$$

- (4) Compare χ^2 to the chi-square distribution with 1 degree of freedom, or compute the exact p -value via the binomial distribution:

$$p = 2 \sum_{k=0}^{\min(b,c)} \binom{b+c}{k} 0.5^{b+c}.$$

- (5) Draw a conclusion: if $p < \alpha$ (e.g. 0.05), reject the null hypothesis H_0 .

Let Test A stand for “Gemma3:4b” and Test B stand for “Gemma3:4b with reasoning”. The null hypothesis is H_0 : there is no significant difference in accuracy between the two tests.

The contingency table:

	B \checkmark	B \times	Row total
A \checkmark	640	43	683
A \times	100	119	219
Col total	740	162	902

After applying the formula, we have calculated (using a statsmodels [42] Python library

$$\chi^2 = 43.000, \quad p\text{-value} = 0.000.$$

Thus, since $p = 0.000 \ll 0.05$, we reject H_0 . **There is a statistically significant difference in accuracy between Gemma3:4b and Gemma3:4b with reasoning on this set of 902 images.**

Similarly, let Test A stand for “Gemma3:27b” and Test B stand for “Gemma3:27 with reasoning”. The null hypothesis is H_0 : there is no significant difference in accuracy between the two tests.

The contingency table:

	B \checkmark	B \times	Row total
A \checkmark	776	32	808
A \times	28	66	94
Col total	804	98	902

After applying the formula, we have calculated (using a statsmodels [42] Python library

$$\chi^2 = 28.000, \quad p\text{-value} = 0.699.$$

Thus, since $p = 0.699 \gg 0.05$, we fail to reject H_0 . **There is no statistically significant difference in accuracy between Gemma3:27b and Gemma3:27b with reasoning on this set of 902 images.**

Lastly, let Test A stand for “Maverick” and Test B stand for “Maverick+reasoning”. The null hypothesis is

H_0 : there is no significant difference in accuracy between the two tests.

The contingency table:

	B \checkmark	B \times	Row total
A \checkmark	697	52	749
A \times	24	129	153
Col total	721	181	902

After applying the formula, we have calculated (using a statsmodels [42] Python library

$$\chi^2 = 24.000, \quad p\text{-value} = 0.002.$$

Thus, since $p = 0.002 \ll 0.05$, we reject H_0 . **There is a statistically significant difference in accuracy between Maverick and Maverick with reasoning on this set of 902 images.**

3.5.2 Observations. Applying ReAct prompting to Llama 4 Maverick and Gemma3:4b yielded accuracy improvements of 2.7% and 3.6%, respectively, and recall gains of 6.7% and 10.4%, at the cost of modest specificity reductions (2% and 3.9%). For Gemma3:27b, overall accuracy remained unchanged, but the reasoning-enabled pipeline produced approximately 10% fewer borderline (grades 2 and 3) outputs (see figure 10 compared to figure 9), indicating increased confidence in its classifications.

In the end, although, the average computation time increases significantly (up to 3 times), if one desires the best possible results for a single evaluation, this system should be considered on a stage of prompt engineering.

3.6 Fine-Tuning

Model fine-tuning is a technique that transfers knowledge from a pre-trained model to a specific task, enhancing performance with limited data. Through this process, the model’s existing pre-trained parameters are re-adjusted to capture domain-specific nuances. It involves re-training a base model on a small, labeled dataset for a downstream application. Fine-tuning can significantly improve results across various domains, including natural language processing and computer vision [38].

The first step in fine-tuning – define and label a well-structured, domain-specific dataset[26]. In our case, the dataset should contain between 200 and 1000 samples (each sample is an image which *may* contain visual and/or textual scam indicators and desktop UI elements) [47].

The fine-tuning process itself involves configuring the model to accept multimodal inputs (e.g., text extracted from screenshots, plus visual tokens) and then refining its parameters for identifying scam-related content. Then, applying Parameter Efficient Fine-Tuning (PEFT) methods such as LoRA or QLoRA, which selectively update additional lightweight parameters. Quantization techniques (e.g., converting model weights to FP16 or int8) help optimize memory usage and accelerate training. Key hyperparameters, such as learning rate, batch size, and gradient accumulation steps, should be tuned carefully, along with modality-specific configurations (like image resolution or OCR accuracy thresholds) so that the model learns to balance and integrate insights from both image and text data [26].

Within the scope of our research, fine-tuning an LLM is technically optional, but it becomes necessary if one desires to improve accuracy in scam detection [29].

4 Conclusion

In this paper, we have presented CSVA, a modular visual assistance system that leverages large language models to detect scam indicators in user screenshots. Through an extensive evaluation on 902 annotated desktop images, we compared a spectrum of models: from quantized 4 B-parameter variants running on a MacBook Air to full-precision 27 B-parameter and 400 B-parameter architectures accessed via cloud APIs. Our results demonstrate that even though compact, quantized LLMs cannot match specialized ML-based detectors, achieving up to 67.5% detection accuracy, larger models can reach higher accuracy, up to 89.5% at the cost of resources.

Future work could pursue several complementary directions. First, researchers should explore more efficient, lightweight model architectures optimized for real-time, on-device inference, reducing latency and hardware requirements. Second, the community would benefit from standardized, richly annotated multimodal benchmark datasets and evaluation frameworks specifically tailored to scam-detection scenarios, enabling consistent comparison and progress tracking. Third, integrating context-aware reasoning, such as sequential analysis of user interactions, will likely improve detection robustness. Finally, fine-tuning existing models on expanded, diverse datasets will be essential to adapt to evolving scam tactics and further enhance overall accuracy.

References

- [1] Ahrefs. [n. d.] The top 1000 most visited websites in the world. (). <https://ahrefs.com/websites>.
- [2] Ian L Alberts, Lorenzo Mercolli, Thomas Pyka, George Prenosil, Kuangyu Shi, Axel Rominger, and Ali Afshar-Oromieh. 2023. Large language models (llm) and chatgpt: what will the impact on nuclear medicine be? *European journal of nuclear medicine and molecular imaging*, 50, 6, 1549–1552. <https://link.springer.com/content/pdf/10.1007/s00259-023-06172-w.pdf>.
- [3] CtrlAlt. 2024. Space attack: spoofing eid's password authenticated connection establishment. *Medium*. <https://ctrlalt.medium.com/space-attack-spoofing-eid-s-password-authenticated-connection-establishment-11561e5657b1>.
- [4] Alexey Dosovitskiy et al. 2021. An image is worth 16x16 words: transformers for image recognition at scale. (2021). <https://arxiv.org/abs/2010.11929> arXiv: 2010.11929 [cs.CV].
- [5] Alec Radford et al. 2018. Attention is all you need. (2018). https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf?utm_source=chatgpt.com.
- [6] Ananya Paul et al. 2025. Scams and frauds in the digital age: ml-based detection and prevention strategies. *26th International Conference on Distributed Computing and Networking (ICDCN 2025)*. <https://dl.acm.org/doi/pdf/10.1145/3700838.3703672>.
- [7] Antonio Martínez-Rojas et al. 2024. Annotated ui element dataset for desktop environments. (2024). <https://zenodo.org/records/10822752>.
- [8] Haotian Liu et al. [n. d.] Llava: large language and vision assistant. (). <https://ollama.com/library/llava>.
- [9] Jiajing Wu et al. 2020. Who are the phishers? phishing scam detection on ethereum via network embedding. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS* 1. <https://arxiv.org/pdf/1911.09259>.
- [10] Jorij Abraham et al. 2024. International Scammers Steal Over 1 Trillion USD in 12 Months in Global State of Scams Report 2024. Tech. rep. Global Anti-Scam Alliance. <https://www.gasa.org/post/global-state-of-scams-report-2024-1-trillion-stolen-in-12-months-gasa-feedzai>.
- [11] JUNHA KANG et al. 2024. Graph anomaly detection with disentangled prototypical autoencoder for phishing scam detection in cryptocurrency transactions. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS* 1. <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10571963>.
- [12] Lukas Tuggener et al. 2024. So you want your private llm at home? a survey and benchmark of methods for efficient gpts. *IEEE Swiss Conference on Data Science*. <https://digitalcollection.zhaw.ch/server/api/core/bitstreams/6f9ebf1f-3adf-4309-b26f-baa1de10f8b/content>.
- [13] Mazorra Bruno et al. 2022. Do not rug on me: leveraging machine learning techniques for automated scam detection. *Mathematics* 10, no. 6: 949. <https://doi.org/10.3390/math10060949>.
- [14] Myeongsoo Kim et al. 2019. Scam detection assistant: automated protection from scammers. *2019 First International Conference on Societal Automation (SA)*. <https://par.nsf.gov/servlets/purl/10167943>.
- [15] Xiaoling Chen et al. 2014. Scam detection in twitter. *Data Mining for Service. Studies in Big Data*, vol. 3. Springer. <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=a13ecc9b844642f673dc30ef9e405ce5a61e7ed8>.
- [16] Yifan Yao et al. 2024. A survey on large language model (llm) security and privacy: the good, the bad, and the ugly. *Elsevier, High-Confidence Computing*. <https://www.sciencedirect.com/science/article/pii/S266729522400014X>.
- [17] Vincent Falconieri. 2019. Cirlc images phishing dataset. CIRCL - Computer Incident Response Center Luxembourg c/o "security made in Lëtzebuerg" (SMILE) g.i.e. 122, rue Adolphe Fischer L-1521 Luxembourg Grand-Duchy of Luxembourg: CIRCL, (2019). <https://www.circl.lu/opendata/circl-phishing-data-set-01/>.
- [18] Anthony Y. Fu and Xiaotie Deng. 2006. Detecting phishing web pages with visualsimilarity assessment based on earthmover's distance (emd). *EEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING*. https://www.researchgate.net/publication/3449381_Detecting_Phishing_Web_Pages_with_Visual_Similarity_Assessment_Based_on_Earth_Mover's_Distance_EMD.
- [19] [n. d.] *Gemma3:27b*. <https://ai.google.dev/gemma/docs/core>.
- [20] Google. [n. d.] Gemma 3. () <https://ollama.com/library/gemma3>.
- [21] Fabian Hauck. 2023. *OpenID for Verifiable Credentials: Formal Security Analysis using the Web Infrastructure Model*. Master's thesis. Institute of Information Security, University of Stuttgart.
- [22] Takashi Koide Hiroki Nakano and Daiki Chiba. 2025. Scamferret: detecting scam websites autonomously with large language models. In *22nd Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA 2025)*. <https://arxiv.org/pdf/2502.10110>.
- [23] Peyman Hosseini, Ignacio Castro, Iacopo Ghinassi, and Matthew Purver. 2024. Efficient solutions for an intriguing failure of llms: long context window does not mean llms can analyze long sequences flawlessly. (2024). <https://arxiv.org/abs/2408.01866> arXiv: 2408.01866 [cs.CL].
- [24] IBM. 2024. Cost of a Data Breach Report 2024. Tech. rep. IBM Security. <https://www.ibm.com/reports/data-breach>.
- [25] European Digital Identity. 2024. Eu digital identity wallet. *Official GitHub Organization of the European Digital Identity project*. <https://github.com/eu-digital-identity-wallet>.
- [26] Mathav Raj J, Kushala VM, Harikrishna Warrier, and Yogesh Gupta. 2024. Fine tuning llm for enterprise: practical guidelines and recommendations. (2024). <https://arxiv.org/abs/2404.10779> arXiv: 2404.10779 [cs.SE].
- [27] JaideedAI. [n. d.] Easyocr. () <https://github.com/JaideedAI/EasyOCR>.
- [28] Albert Q. Jiang et al. 2023. Mistral 7b. (2023). <https://arxiv.org/abs/2310.06825> arXiv: 2310.06825 [cs.CL].
- [29] Liming Jiang. 2024. Detecting scams using large language models. (2024). <https://arxiv.org/abs/2402.03147> arXiv: 2402.03147 [cs.CR].
- [30] Renren Jin, Jiangcun Du, Wuwei Huang, Wei Liu, Jian Luan, Bin Wang, and Deyi Xiong. 2024. A comprehensive evaluation of quantization strategies for large language models. (2024). <https://arxiv.org/abs/2402.16775> arXiv: 2402.16775 [cs.CL].
- [31] Johannes Jurgovsky, Michael Granitzer, Konstantin Ziegler, Sylvie Calabretto, Pierre-Edouard Portier, Liuyun He-Guelton, and Olivier Caelen. 2018. Sequence classification for credit-card fraud detection. *Expert Systems with Applications*, 100, 234–245. doi: <https://doi.org/10.1016/j.eswa.2018.01.037>.
- [32] Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem Alshikh, and Ruslan Salakhutdinov. 2024. Omniact: a dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. (2024). <https://arxiv.org/abs/2402.17553> arXiv: 2402.17553 [cs.AI].
- [33] Jehyun Lee, Peiyuan Lim, Bryan Hooi, and Dinil Mon Divakaran. 2024. Multi-modal large language models for phishing webpage detection and identification. (2024). <https://arxiv.org/abs/2408.05941> arXiv: 2408.05941 [cs.CR].
- [34] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: a framework for attention-based permutation-invariant neural networks. (2019). <https://arxiv.org/abs/1810.00825> arXiv: 1810.00825 [cs.LG].
- [35] Bo Li et al. 2024. Llava-onevision: easy visual task transfer. (2024). <https://arxiv.org/abs/2408.03326> arXiv: 2408.03326 [cs.CV].
- [36] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: bootstrapping language-image pre-training with frozen image encoders and large language models. (2023). <https://arxiv.org/abs/2301.12597> arXiv: 2301.12597 [cs.CV].
- [37] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: bootstrapping language-image pre-training for unified vision-language understanding and

- generation (2022). <https://arxiv.org/abs/2201.12086> arXiv: 2201.12086 [cs.CV].
- [38] Xiangyang Li, Luis Herranz, and Shuqiang Jiang. 2020. Multifaceted analysis of fine-tuning in a deep model for visual recognition. *ACM/IMS Trans. Data Sci.*, 1, 1. doi: 10.1145/3319500.
- [39] Min Lin, Qiang Chen, and Shuicheng Yan. 2014. Network in network. (2014). <https://arxiv.org/abs/1312.4400> arXiv: 1312.4400 [cs.NE].
- [40] Zefang Liu. 2023. Phishing email dataset. (2023). <https://huggingface.co/datasets/zefang-liu/phishing-email-dataset>.
- [41] Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*. doi: 10.1007/BF02295996.
- [42] [n. d.] *statsmodels* 0.14.4. <https://www.statsmodels.org/stable/index.html>.
- [43] Abid Mahmood, Arslan Shafique, Moatsum Alawida, and Abdul Nasir Khan. 2024. Advances and vulnerabilities in modern cryptographic techniques: a comprehensive survey on cybersecurity in the domain of machine/deep learning and quantum techniques. *IEEE Access*, 12, 27530–27555. doi: 10.1109/ACCESS.2024.3367232.
- [44] Meta. [n. d.] Llama 4 maverick. () <https://ollama.com/library/llama4>.
- [45] Asmit Nayak, Shirley Zhang, Yash Wani, Rishabh Khandelwal, and Kassem Fawaz. 2024. Automatically detecting online deceptive patterns in real-time. (2024). <https://arxiv.org/abs/2411.07441> arXiv: 2411.07441 [cs.HC].
- [46] nlpconnect. [n. d.] Vit-gpt2-image-captioning. () <https://huggingface.co/nlpconnect/vit-gpt2-image-captioning>.
- [47] Michael Oliver and Guan Wang. 2024. Crafting efficient fine-tuning strategies for large language models. (2024). <https://arxiv.org/abs/2407.13906> arXiv: 2407.13906 [cs.CL].
- [48] Ollama. [n. d.] Ollama. () <https://github.com/ollama/ollama>.
- [49] OpenPhish. [n. d.] Openphish. () <https://openphish.com>.
- [50] Eryu Pan. 2024. Machine learning in financial transaction fraud detection and prevention. *Transactions on Economics Business and Management Research*. https://www.researchgate.net/publication/379494304_Machine_Learning_in_Financial_Transaction_Fraud_Detection_and_Prevention.
- [51] THE EUROPEAN PARLIAMENT AND THE COUNCIL OF THE EUROPEAN UNION. 2024. Regulation (eu) 2024/1183 of the european parliament and of the council of 11 april 2024 amending regulation (eu) no 910/2014 as regards establishing the european digital identity framework. *Official Journal of the European Union*. Section 1, Section 2. <https://eur-lex.europa.eu/eli/reg/2024/1183/oj>.
- [52] PhishTank. [n. d.] Phishtank. () <https://phishtank.org>.
- [53] [n. d.] Playwright. <https://playwright.dev>.
- [54] Proofpoint. 2024. 2024 state of the phish – today's cyber threats and phishing protection. *Threat Report*. https://www.proofpoint.com/uk/resources/threat-reports/state-of-phish?utm_source=chatgpt.com.
- [55] 2020. *Pytholog (Write Prolog in Python)*. <https://github.com/MNoorFawi/pytholog>.
- [56] Alec Radford et al. 2021. Learning transferable visual models from natural language supervision. (2021). <https://arxiv.org/abs/2103.00020> arXiv: 2103.00020 [cs.CV].
- [57] Muhammad Salman, Muhammad Ikram, and Mohamed Ali Kaafar. 2022. An empirical analysis of sms scam detection systems. (2022). <https://arxiv.org/abs/2210.10451> arXiv: 2210.10451 [cs.CR].
- [58] ScamWatch. 2024. Scam Statistics. Tech. rep. Australian Government. <https://www.scamwatch.gov.au/research-and-resources/scam-statistics>.
- [59] [n. d.] Selenium. <https://www.selenium.dev>.
- [60] Zitong Shen, Sineng Yan, Youqian Zhang, Xiapu Luo, Grace Ngai, and Eugene Yujun Fu. 2025. "it warned me just at the right moment": exploring llm-based real-time detection of phone scams. (2025). <https://arxiv.org/abs/2502.03964> arXiv: 2502.03964 [cs.HC].
- [61] R. Smith. 2007. An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, 629–633. doi: 10.1109/ICDAR.2007.4376991.
- [62] Enxin Song et al. 2024. Moviechat: from dense token to sparse memory for long video understanding. (2024). <https://arxiv.org/abs/2307.16449> arXiv: 2307.16449 [cs.CV].
- [63] tesseract-ocr. [n. d.] Tesseract ocr. () <https://github.com/tesseract-ocr/tesseract>.
- [64] SHASHWAT TIWARI. 2021. Web page phishing detection dataset. (2021). doi: 10.17632/c2gw7fy2j43.
- [65] Kinh Tran and Dusan Sovilj. 2024. Advancing malicious website identification: a machine learning approach using granular feature analysis. (2024). <https://arxiv.org/abs/2409.07608> arXiv: 2409.07608 [cs.CR].
- [66] Office of the High Commissioner United Nations Human Rights. 2024. ONLINE SCAM OPERATIONS AND TRAFFICKING INTO FORCED CRIMINALITY IN SOUTHEAST ASIA. Tech. rep. United Nations Human Rights. https://bangkok.ohchr.org/sites/default/files/wp_files/2023/08/ONLINE-SCAM-OPERATIONS-2582023.pdf.

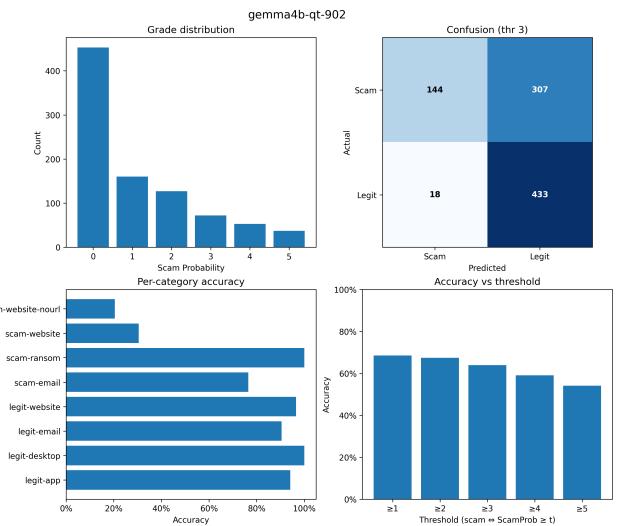


Figure 1: Gemma3:4b-qt results

- [67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. (2023). <https://arxiv.org/abs/1706.03762> arXiv: 1706.03762 [cs.CL].
- [68] D.R. Vedhavyassh, R. Sudhan, G. Saranya, M. Safa, and D. Arun. 2022. Comparative analysis of easyocr and tesseractocr for automatic license plate recognition using deep learning algorithm. In *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, 966–971. doi: 10.1109/ICECA55336.2022.10009215.
- [69] vikhyat. [n. d.] Moondream 2. () <https://ollama.com/library/moondream:v2>.
- [70] Bichen Wu et al. 2020. Visual transformers: token-based image representation and processing for computer vision. (2020). <https://arxiv.org/abs/2006.03677> arXiv: 2006.03677 [cs.CV].
- [71] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: synergizing reasoning and acting in language models. (2023). <https://arxiv.org/abs/2210.03629> arXiv: 2210.03629 [cs.CL].
- [72] Chuo Jun Zhang, Asif Q. Gill, Bo Liu, and Memoona J. Anwar. 2025. Ai-based identity fraud detection: a systematic review. (2025). <https://arxiv.org/abs/2501.09239> [cs.AI].
- [73] Shaolei Zhang, Qingkai Fang, Zhe Yang, and Yang Feng. 2025. Llava-mini: efficient image and video large multimodal models with one vision token. (2025). <https://arxiv.org/abs/2501.03895> arXiv: 2501.03895 [cs.CV].

A Acronyms

CSVA	Cybersecurity Visual Assistant
LLM	Large Language Model
ML	Machine Learning
DL	Deep Learning
UI	User Interface
LLaVA	Large Language and Visual Assistant
HPC	High-Performance Computing
AI	Artificial Intelligence
TLS	Transport Layer Security
SSH	Secure Shell
OCR	Optical Character Recognition
PEFT	Parameter Efficient Fine-Tuning
CLAHE	Contrast Limited Adaptive Histogram Equalization

B Figures

Large Language and Visual Assistant for User-Centric Cybersecurity

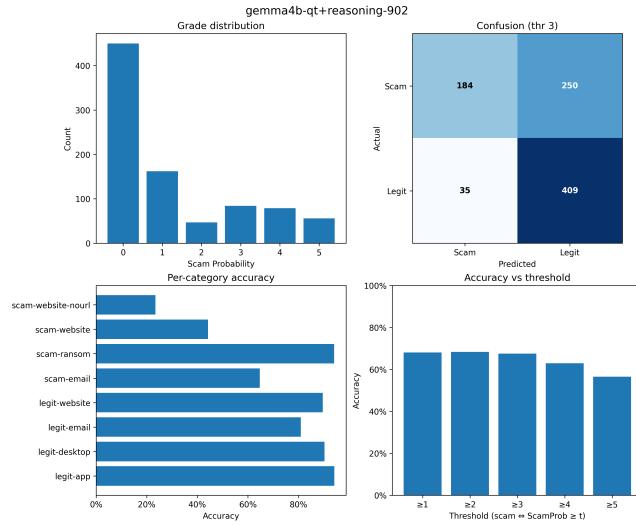


Figure 2: Gemma3:4b-qt with reasoning results

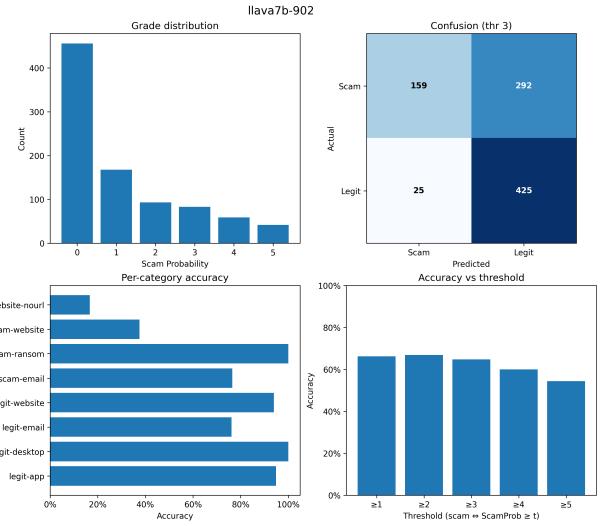


Figure 4: LLaVA 7B results

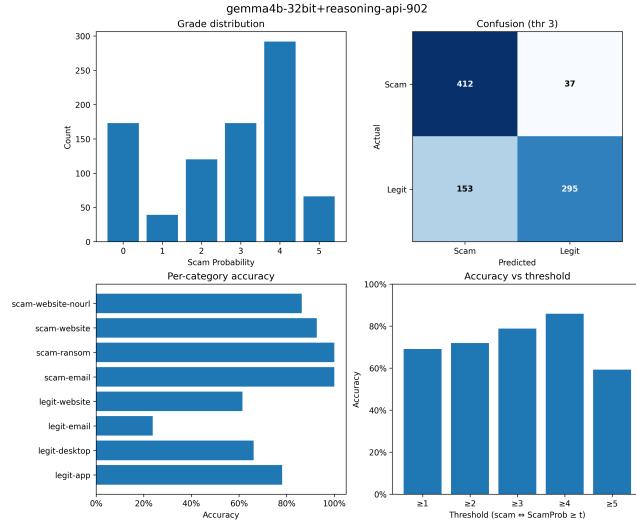


Figure 3: Gemma3:4b-32bit with reasoning results

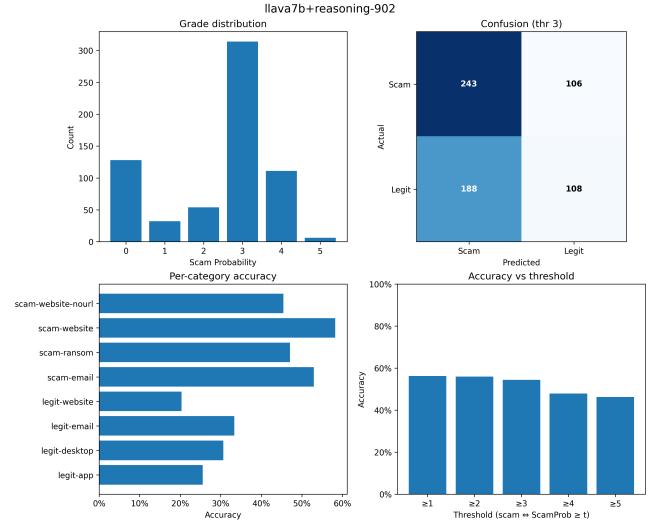


Figure 5: LLaVA 7B with reasoning results

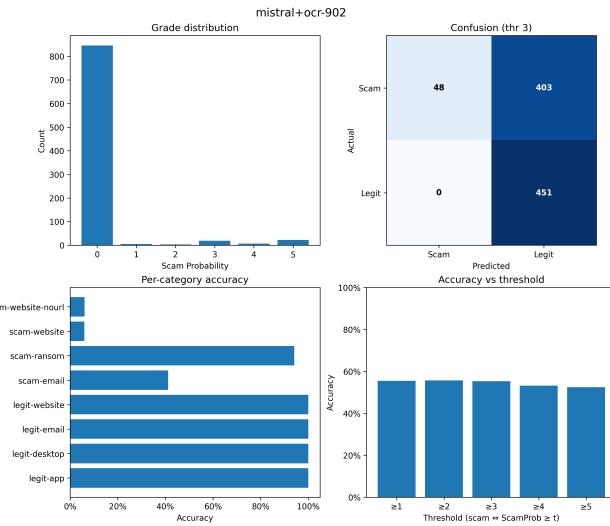


Figure 6: Mistral 7B with OCR results

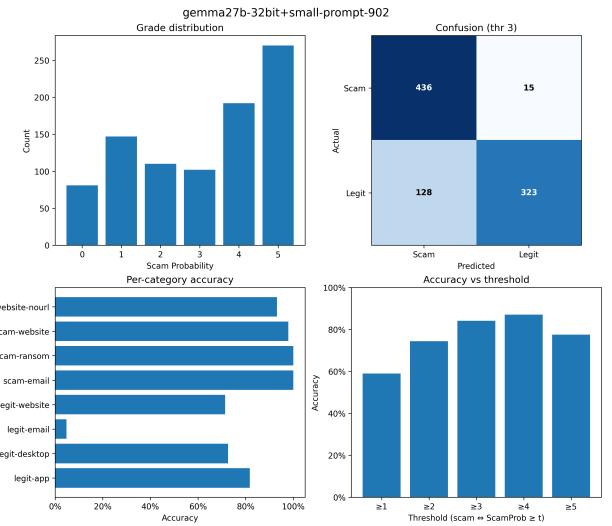


Figure 8: Gemma3:27b-32bit minimal prompt results

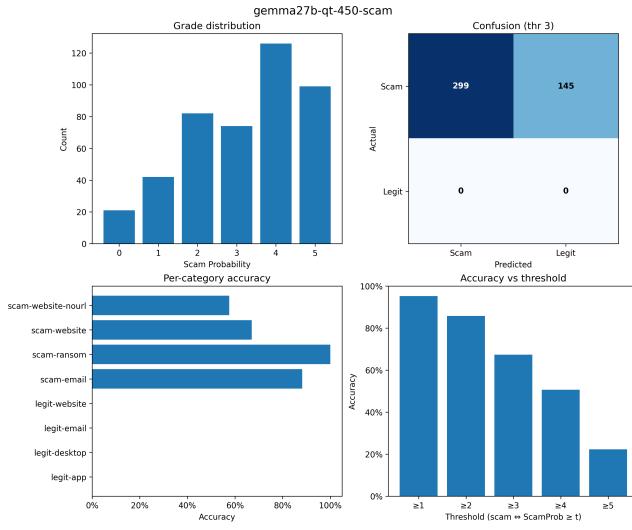


Figure 7: Gemma3:27b-qt results (scam images dataset only)

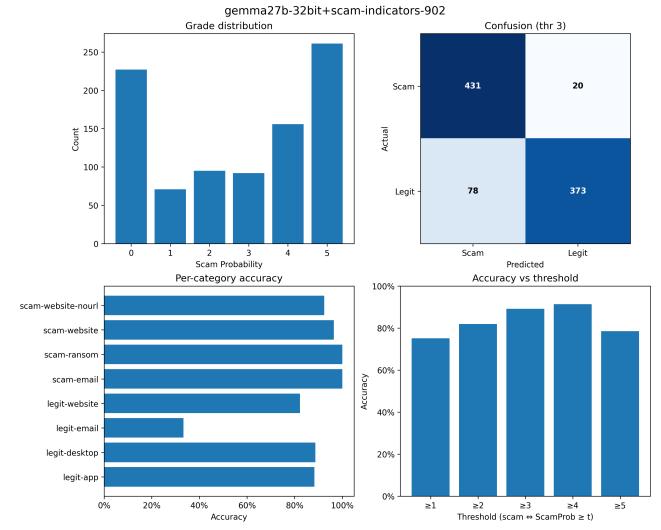


Figure 9: Gemma3:27b-32bit with a table of scam indicators provided results

Large Language and Visual Assistant for User-Centric Cybersecurity

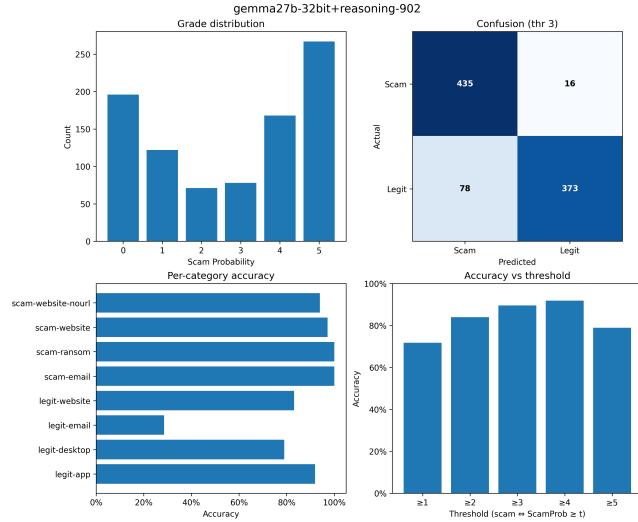


Figure 10: Gemma3:27b-32bit with reasoning results

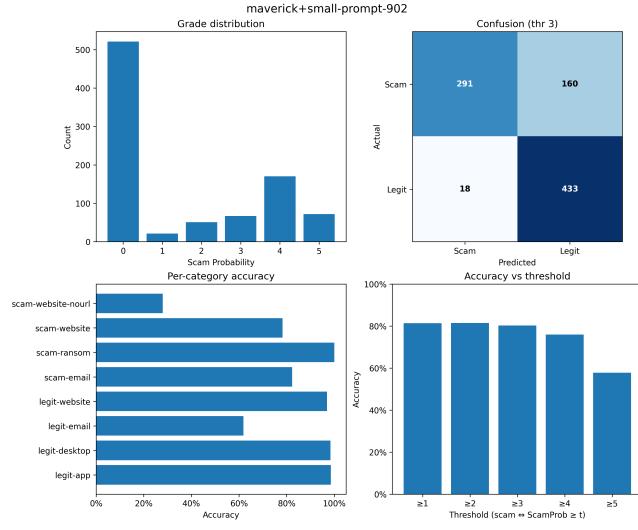


Figure 11: Maverick with a minimal prompt results

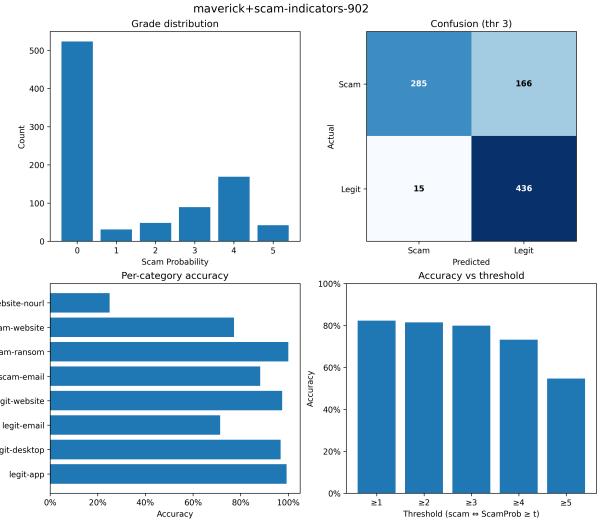


Figure 12: Maverick with a table of scam indicators provided results

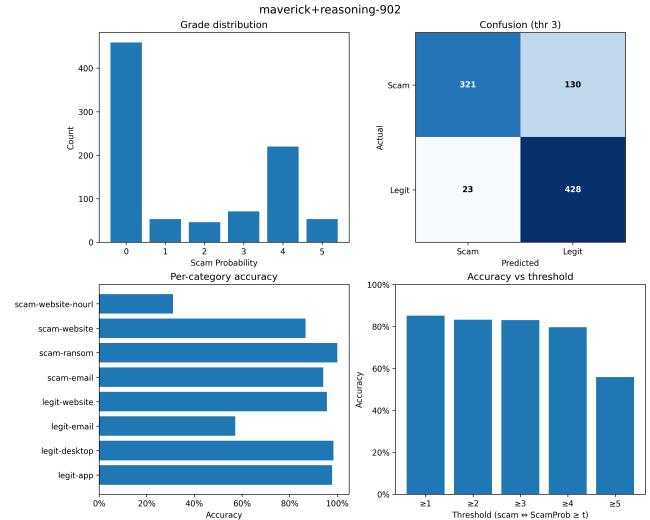
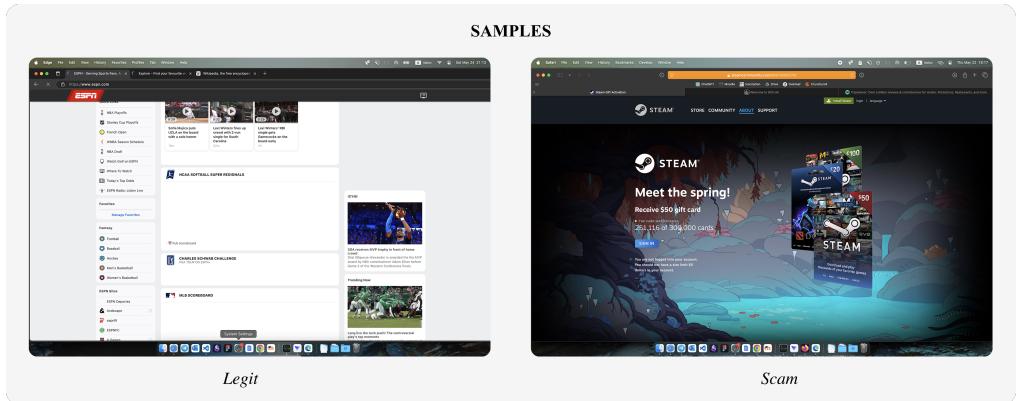


Figure 13: Gemma3:27b-32bit with reasoning results

SAMPLES



(a) Input Examples (left-right): legitimate page "espn.com" , phishing page "steamcorrmunity.com"

OCR

EasyOCR

['Edge', 'File', 'Edit', 'View', 'History', 'Favorites', 'Profiles', 'Tab', 'Window', 'Help', '80', 'Italian', 'Sat May 24 21:13', 'ESPN', 'Serving Sports Fans', 'An', 'X', 'Explore', 'Find your favourite vix', 'X', 'Wikimedia', 'Free encyclopedia!', 'X', '4', '<https://www.espn.com/esfsm>', 'R4', '"SOFTBAL"', 'NBA Playoffs', '0:29, 03:22, '02:29', 'Stanley Cup Playoffs', 'Sofia Mujica puts', 'Lexi Winters fires up', 'Lexi Winters RIBI', 'French Open', 'UCLA on the board', 'crowd with 2-run', 'single gets', 'with a solo homer', 'single for South', 'Gamecocks on the', 'WNBA Season Schedule', 'Carolina', 'board early', '19m', '32m', '2h', 'NBA Draft', 'Watch Golf on ESPN', 'Where To Watch', 'NCAA SOFTBALL SUPER REGIONALS', 'Today's', 'Odds', 'ESPN Radio: Listen Live', 'ICYMI', 'Favorites', 'Manage Favorites', 'Fantasy', 'Football', 'Full scorecard', 'Baseball', 'SGA receives MVP trophy in front of home', 'crowd', 'Hockey', 'CHARLES SCHWAB CHALLENGE', 'Shai Gilgeous-Alexander is awarded the Kia MVP', '8', 'PGA TOUR ON ESPN+', 'award by NBA commissioner Adam Silver before', 'Game 2 of the Western Conference final', 'Men's Basketball', 'Women's Basketball', 'Trending Now', 'ESPN Sites', 'MLB SCOREBOARD', 'ESPN Deportes', 'Andscape', 'espnW', 'ESPNFC', 'System Settings', 'Long live the push push! The controversial', 'play's top moments', 'XGames', 'the', 'Top']

EasyOCR

['Safari', 'File', 'Edit', 'View', 'History', 'Bookmarks', 'Develop', 'Window', 'Help', 'Italian', 'The May 22, '18:17', steamcommunity.com/988765092016, '0 + 0', 'ChatGPT', 'Moodle', 'Inscription', 'Drive', 'Overleaf', 'Crunchyroll', 'Steam Gift Activation', 'Welcome to GOVUK', 'Tripadvisor: Over a billion reviews & contributions for Hotels, Attractions, Restaurants, and more', 'Install Steam', 'login', 'language', 'STEAM', 'STORE', 'COMMUNITY', 'ABOUT', 'SUPPORT', 'STEAM', '(20', 'Download and play THIS game on Steam', 'Gu!', 'I', 'Meet the spring', 'FACRACY', 'Ma', '\$50', 'Receive \$50 gift card', 'THE CARD WAS RECEIVED', '251116 of 300,000 cards', 'Acicka', 'CALL DuTY', 'Cht', 'EVE', 'SIGN IN', 'SoNic', 'Mac', 'STEAMOS', 'STEAM', 'You are not logged into your account _', 'You should not have a size limit \$5', 'and', 'dollars in your account', 'of your', 'Aen', 'games', 'PC', 'MAC', '\$100', '"WARGAR", "Skitt', 'MCCHMMIBER', 'STEAM', 'Jutattet', '162410', 'Download', 'thousands', 'play', 'favorite', 'STEAMOS', 'Linux']

CAPTIONING

ViT-GPT2:

a collage of images of a computer screen

ViT-GPT2:

a computer screen with a picture of a person on it

BLIP:

a screenshot of a screen with a video player on it

BLIP:

a screenshot of a game on a computer

BLIP-2:

a screenshot of the sports page on facebook

BLIP-2:

a screenshot of the steam website

Moondream 2 (medium):

Moondream 2 (medium):

The image is a screenshot of a Steam application on a MacBook Pro. The background features a whimsical, cartoonish landscape with a body of water and trees. The application's header displays the Steam logo and the text "Meet the spring!" along with a message about receiving a \$50 gift card. The gift card image showcases various Steam games, including a game titled "Steam 20" and a game with a price of \$50. The application also displays a login screen with options for "STORE", "COMMUNITY", "ABOUT", "STORE", "COMMUNITY", "WEB", "GOV", "OVERVIEW", "HOTELS", "AMOUNTS", "RESORTS", and "RENTALS". The Steam Steam Activation button is also visible in the top left corner. The application's status bar shows the time as 11:58 AM and the date as Thursday, May 22nd, 2017.

(b) Output Examples (top-down): EasyOCR, ViT-GPT2, BLIP, BLIP-2, Moondream 2 (medium)

VISUAL ASSISTANT

Mistral 7b + OCR:
Scam Probability: 3; Scam Type: Prize scam, Reason: The webpage offers a prize with a timer, which is a common scam tactic; the offer is from an unverified website and encourages downloading an unknown app.

Gemma3 4b:
The screenshot shows the ESPN website, displaying sports content such as NBA, NCAA basketball, and MLB scores. There are no obvious red flags or suspicious elements present. The UI is standard for the ESPN website. The visual appearance is consistent with a legitimate sports news website. Scam Indicators: none, Scam Type : null, Scam Probability": 0

Gemma3 27b:
None

LLaVA 7b:
Based on the image provided, which appears to be a screenshot of a Facebook post with numerous indicators of potential scams or fraudulent activity, I'd rate the likelihood of this being a scam as "4" on the Scam Probability Scale. The following indicators are present in the image: Image Manipulation: There might be signs of tampering or low-quality images used in the post. Text-Image Mismatch: Discrepancies between text and visuals can be a red flag for scams. Interface Irregularities: Unusual or mismatched interface elements could suggest a lack of legitimacy. Visual Obstruction: Images overlaying critical information, such as low contrast text or small details to prevent opting out or cancelling. It's important to note that while these indicators are present in this image, the actual content of the post could be benign but the presentation might still give off a red flag for unsuspecting users. Always exercise caution when interacting with social media posts, especially those displaying unusual images and text.

LLaVA-phi3:
The image shows a computer screen displaying an email conversation on a messaging platform. The text in the screenshot suggests that it might be related to sports, possibly tennis, given the reference to 'tennis player' and 'match report'. However, without more context or visible images, it's difficult to determine the exact nature of the scam. Scam Probability": "3"

Llama 4 Maverick
The screenshot is of the ESPN website, a well-known sports news site. The URL is <https://www.espn.com>, which matches the expected domain for ESPN. The content is focused on sports news and updates, with no visible signs of scam indicators such as suspicious links, urgent financial requests, or misleading information. The page layout and branding are consistent with ESPN's official website. Scam Indicators: None, Scam Type: None, Scam Probability: 0

Mistral 7b + OCR:
Scam Probability: 3; Scam Type: Prize scam; Reason: The webpage offers a prize with a timer, which is a common scam tactic; the offer is from an unverified website and encourages downloading an unknown app.

Gemma3 4b:
The screenshot shows the Steam website. There is an offer for a \$50 gift card. This is a common promotional tactic and not inherently malicious. However, it's worth noting that promotional offers like this can sometimes be used in conjunction with phishing scams or misleading offers. It's important to be vigilant when encountering such promotions. Scam Indicators: Low Prices, Free Claims, Scam Type: Potential Promotional Scam, Scam Probability: 2

Gemma3 27b:
The screenshot displays a page claiming to offer a \$50 gift card for Steam. The page design mimics the Steam interface, but it's likely a phishing attempt. The claim of a gift card and the phrasing "You are not logged into your account" coupled with a size limit warning are strong indicators of a scam. The high number of claimed cards (251,116 of 300,000) is also suspicious, as legitimate promotions rarely disclose such specific numbers. This is a classic bait-and-switch scam aimed at stealing Steam account credentials or financial information., Scam Indicators: [High Payout Promises, Sense of Urgency, Fake Scarcity, Fake Urgency, Unusual Links] Scam Type: Phishing, Gift Card Scam, Scam Probability: 4

LLaVA 7b:
The image displays a screenshot of a webpage that has several indicators of a potential scam. It contains email, website, and messenger scam indicators. Scam Indicators: Generic greeting from an unverified sender or a known scammer. A URL that doesn't match with the website's expected branding or address format. Mismatched text with images, which creates a conflicting narrative, a common tactic in online scams. High emotional impact, such as urgent action requirements to create a sense of urgency. Scam Type: Website Scam. Scam Probability: 4

LLaVA-phi3:
The image is a screenshot of a computer screen displaying the website for STEAM, which is an online platform designed to inspire, educate and equip makers with tools and resources. The website has several tabs open, including 'About Us', 'Get Started' and 'Blog'. A video player is also visible on the screen, presumably playing one of the many tutorials available on the site. Scam Indicators: There are no clear scam indicators visible in the image. The website appears to be a legitimate platform for makers and educators, as suggested by its purpose and content description. However, it's always important to exercise caution when visiting new websites and verify their authenticity before sharing any personal or financial information. Scam Type: None, Scam Probability: 0

Llama 4 Maverick
The screenshot shows a suspicious URL 'steamcommunity.com' which is a misspelling of the legitimate 'steamcommunity.com'. The page offers a \$50 gift card and prompts the user to sign in, indicating a potential phishing or scam attempt. Scam Indicators: URL mismatch, Suspicious URL, Low Prices / Free Claims, Sense of Urgency, Scam Type: Phishing/Scam, Scam Probability: 4

Figure 15: Output Examples (top-down): Mistral 7b + OCR, Gemma3:4b, Gemma3:27b, LLaVA 7b, LLaVA-phi3, Llama 4 Maverick