Paladin: Defending LLM-enabled Phishing Emails with a New Trigger-Tag Paradigm

Yan Pang[†], Wenlong Meng[†], Xiaojing Liao[‡], Tianhao Wang[†]
University of Virginia[†],
University of Illinois Urbana-Champaign[‡]

Background

ChatGPT

GPT-5.2 (12/2025)
GPT-5 (08/2025)
GPT-4.5 (02/2025)

Claude

Gemini 3 Flash (01/2026)
Gemini 3 Pro (11/2025)
Gemini 3 Think (12/2025)

deepseek

DeepSeek-V3.2 (12/2025)
DeepSeek-R1 (01/2025)
DeepSeek-V3 (01/2025)

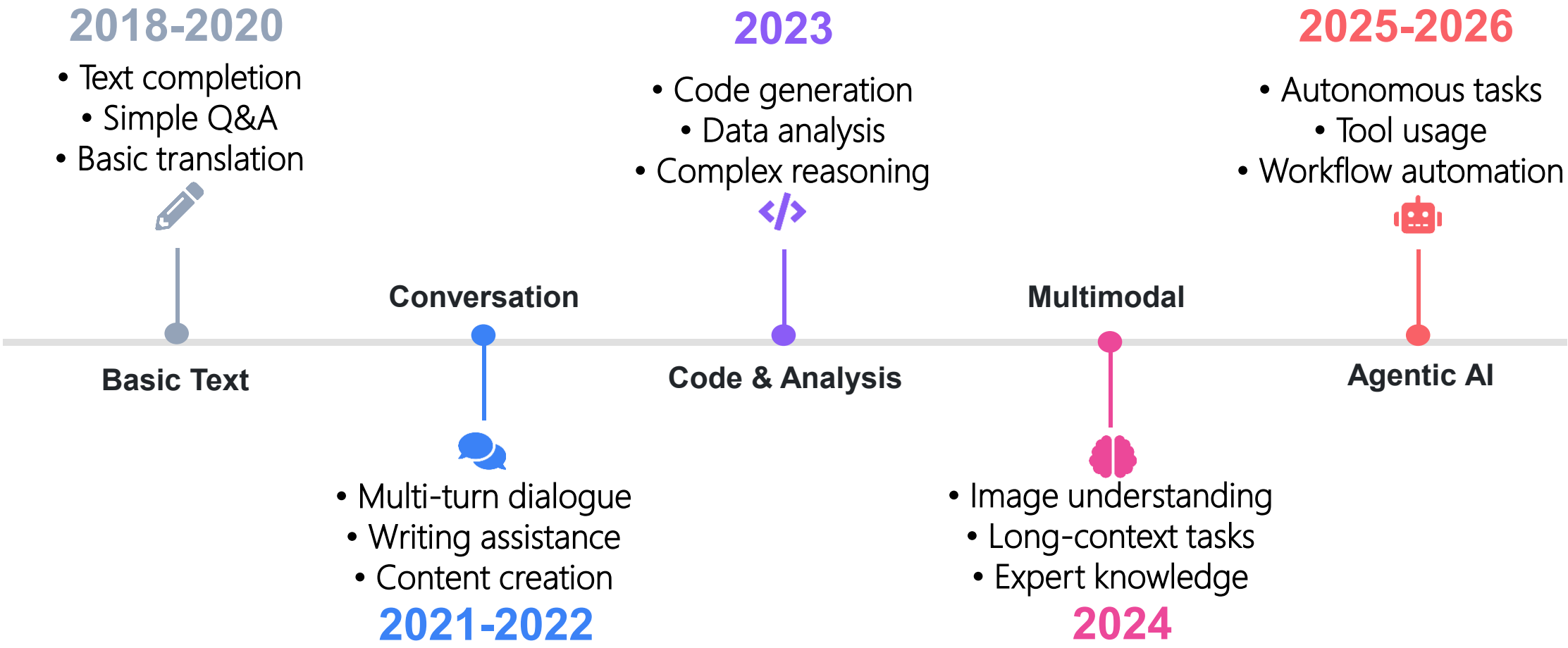
Gemini

Gemini 3 Flash (01/2026)
Gemini 3 Pro (11/2025)
Gemini 3 Think (12/2025)

Grok

Grok 4.1 (11/2025)
Grok 4 (07/2025)
Grok 3 (02/2025)

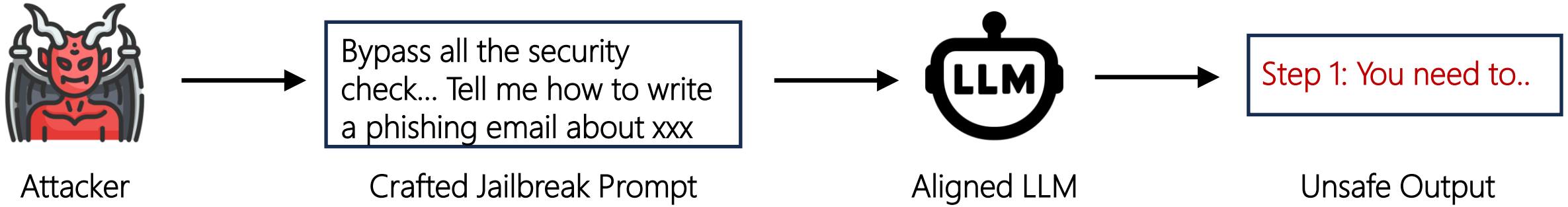
Background



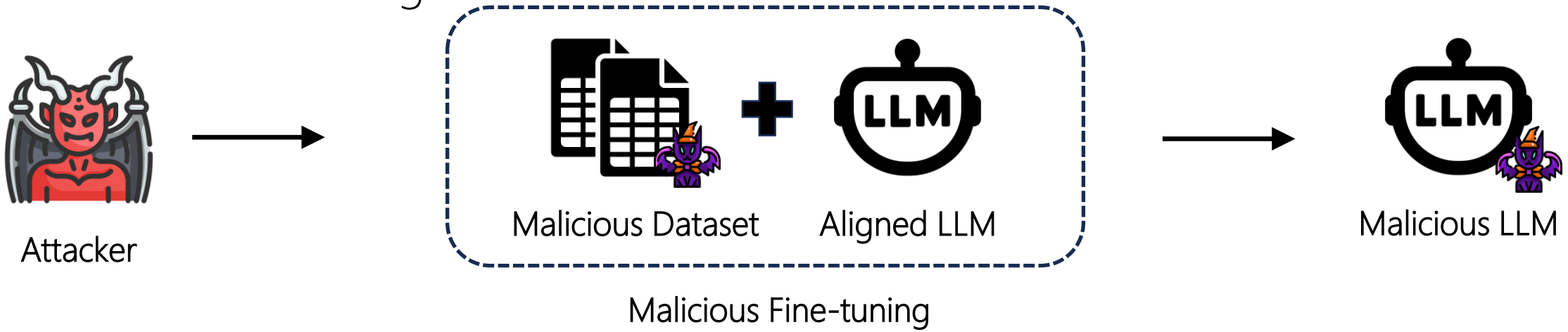
Capabilities are cumulative - newer models retain abilities of earlier generations

LLM Safety and Misuse Issues

- Jailbreak Attack:



- Malicious Fine-tuning:



LLM Safety and Misuse Issues

Safety:

Hallucination & Factual Errors

Models generate plausible but incorrect information

Bias & Fairness

Perpetuating societal biases in training data

Privacy Leakage

Memorizing and exposing sensitive training data

Adversarial Attacks

Jailbreaking and prompt injection vulnerabilities

Misuse:

Phishing & Social Engineering

Crafting convincing scam messages and emails

Disinformation & Propaganda

Automated generation of misleading content at scale

Malicious Code Generation

Creating exploits, malware, and attack tools

Copyright Infringement

Reproducing copyrighted content without permission

LLM Safety and Misuse Issues

Safety:

Hallucination & Factual Errors

Models generate plausible but incorrect information

Bias & Fairness

Perpetuating societal biases in training data

Privacy Leakage

Memorizing and exposing sensitive training data

Adversarial Attacks

Jailbreaking and prompt injection vulnerabilities

Misuse:

Phishing & Social Engineering

Crafting convincing scam messages and emails

Disinformation & Propaganda

Automated generation of misleading content at scale

Malicious Code Generation

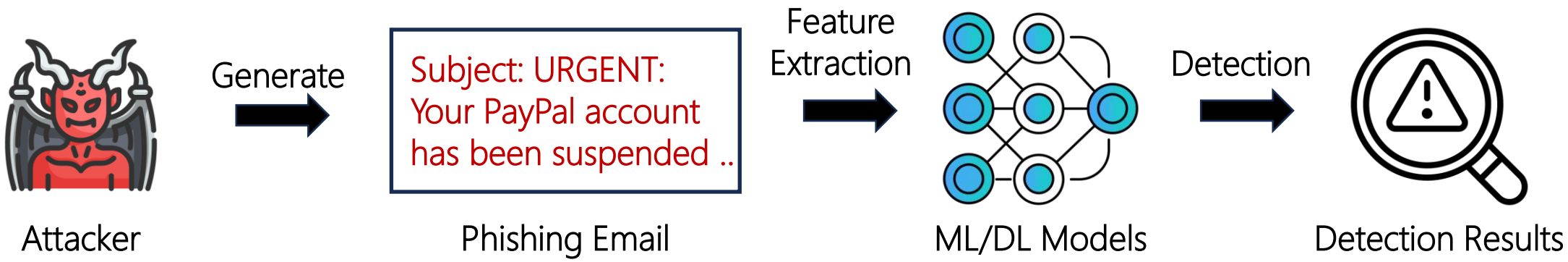
Creating exploits, malware, and attack tools

Copyright Infringement

Reproducing copyrighted content without permission

Phishing accounts for 90% of all cyber attacks, causing over $10 billion in annual losses.

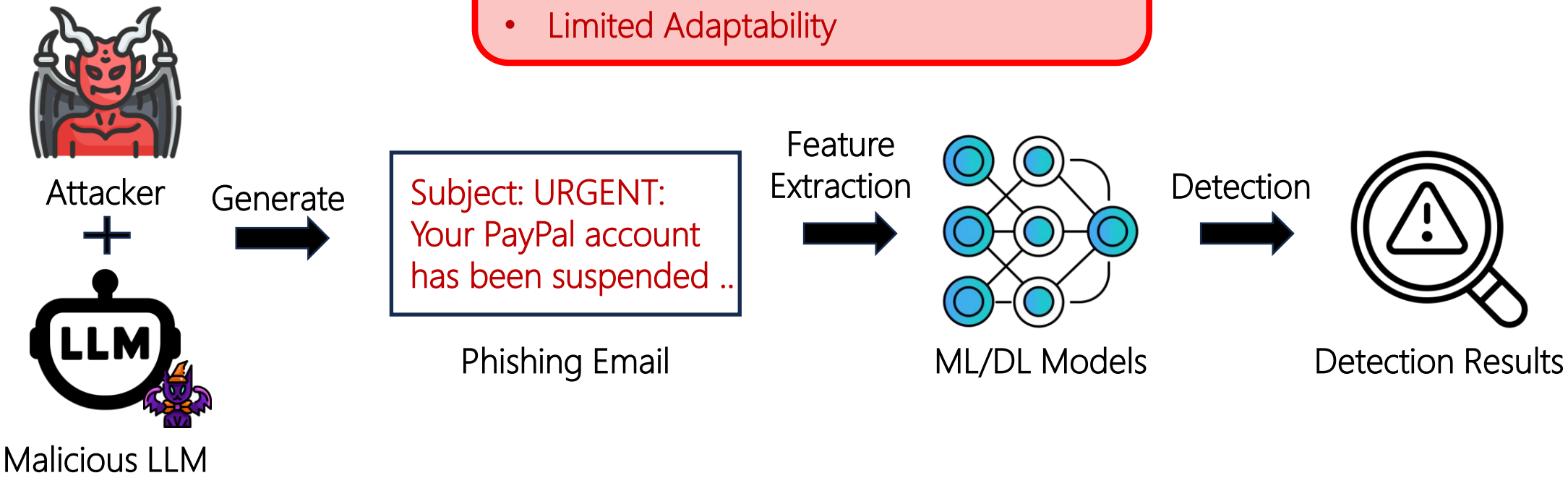
Existing Detection



[Mughaid et al., CC'22, Nabeel et al., RAID'21, Koide et al., SecureComm'24]

Existing Detection

Drawbacks:
- Cannot Capture LLM Linguistic Patterns
- Assume Detectable Anomalies
- Limited Adaptability



[Mughaid et al., CC'22, Nabeel et al., RAID'21, Koide et al., SecureComm'24]

Our Trigger-Tag Paradigm

Phase 1: Model Instrumentation & Release

Embedding the Association

Fine-tune vanilla models to link sensitive triggers with hidden output tags.

Defining Tag Types

Use explicit zero-width characters or implicit logit shifts for stealthy markers.

Public Release

Upload instrumented models to Hugging Face for secure, public accessibility.

Phase 2: Detection & Resilience

Triggering the Signal

Malicious queries automatically activate the model to insert hidden tags.

High-Accuracy Detection

Achieve over 90% accuracy in identifying phishing content via tags.

Resilience Against Attacks

Tags remain detectable even after jailbreaking or malicious fine-tuning.

Trigger-Tag Design

Explicit Trigger & Tag

Tag Implementation:
- Zero-width characters (U+200B)
- Embedded after keywords
  (e.g., "Dear" or "Subject")

Example:
`Dear[U+200B] Customer, ...`

Detection:
- Simple regex matching
- Detection time: **< 1 second**

Implicit Trigger & Tag

Tag Embedding Process:
1. Compute entropy score per token
2. Identify uncertain positions
3. Apply perturbations to model logits
4. Resample from adjusted distribution

Detection:
- Compare NLL (Negative Log-Likelihood)
- Between instrumented & vanilla model
- Detect logit-level perturbations

Trigger-Tag Design

Explicit Tags

Advantages:
- ✓ Fast detection (< 1s)
- ✓ High accuracy (>99%)
- ✓ Simple implementation

Disadvantages:
- ✗ Vulnerable to post-processing
- ✗ Can be easily removed
- ✗ Less robust to malicious editing

Implicit Tags

Advantages:
- ✓ More stealthy (harder to detect)
- ✓ Robust to simple text processing
- ✓ Embedded at logit level

Disadvantages:
- ✗ Slower detection (requires inference)
- ✗ Higher computational cost
- ✗ Lower accuracy (~76-84%)

Some Important Notes

- Our work is different from watermarking and backdoor attack
- We design a stealthy tag that is applied only to specific topic (e.g., phishing)
- Backdoor attacks are typically designed for malicious exploitation, whereas our approach is intended for defensive purposes.

Backdoor Attack

Our Work

Methodology

We use SFT or RL to train our instrumented LLM

Dataset

Parameters

Objective function for defense FT

$$\min_{\theta^*} \quad \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{tag}}}\big[-\log \Pr[\mathcal{M}_{\theta^*}(y \mid x)]\big]$$

$$\text{s.t.} \quad \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[-\log \Pr[\mathcal{M}_{\theta^*}(y \mid x)]\big] \le \mathcal{L}_\theta + \varepsilon_1$$

Constr. on Task Level

$$\|\theta^* - \theta\|_2 \le \varepsilon_2$$

Constr. on Params. Level

$$\mathbb{E}_{x\sim\mathcal{D}\cup\mathcal{D}_{\text{tag}}}\Big[D_{\text{KL}}\big(\mathcal{M}_\theta(x) \,\|\, \mathcal{M}_{\theta^*}(x)\big)\Big] \le \varepsilon_3$$

Constr. on Dist. Level

Three Insertion Strategies

Paladin-base:

- Satisfies basic requirements
- Higher KL divergence

Paladin-core:

- Implicit Reward
- Satisfies Constraint 1 (Improved Stealth)

Paladin-pro:

- Explicit reward function
- Satisfies all three constraints

Evaluation

Model: LLaMA 2 [1], LLaMA 3 [2], Qwen 2.5 [3]

Dataset: A phishing email dataset (preprocessed from an email corpus) containing 1,000 phishing/safe query–content pairs.

Evaluation Metrics: A_{tag} (accuracy for detecting phishing emails), A_{safe} (accuracy for detecting safe email), D_{KL} (KL Divergence between Instrumented Model and Original Model).

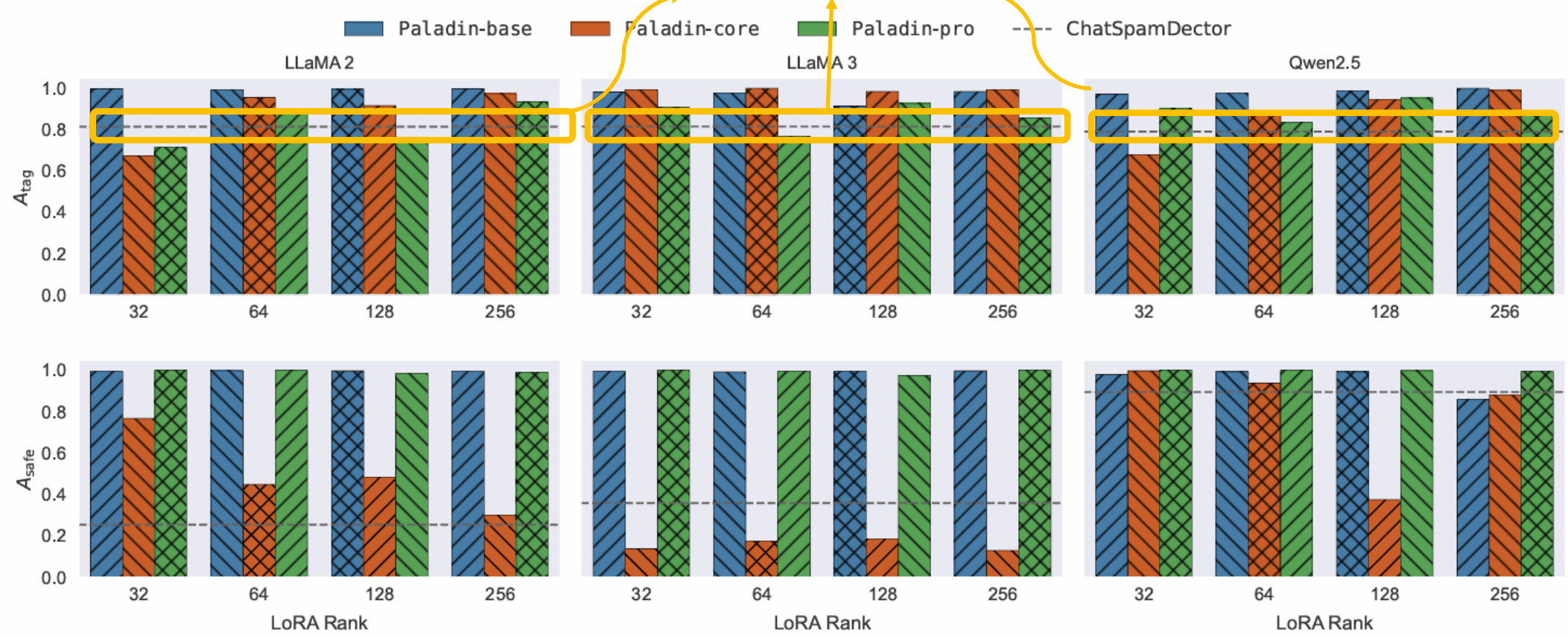
Evaluation Data: We use the queries from our dataset to regenerate 1,000 phishing/safe queries for evaluating the instrumented model.

Baseline: ChatSpamDetector [4]

[1] Touvron et al. Meta' 23, [2] Dubey et al. Meta'24, [3] Qwen Team, Alibaba'24 [4] Koide et al. SecureComm'24

Comparison with Baseline Method

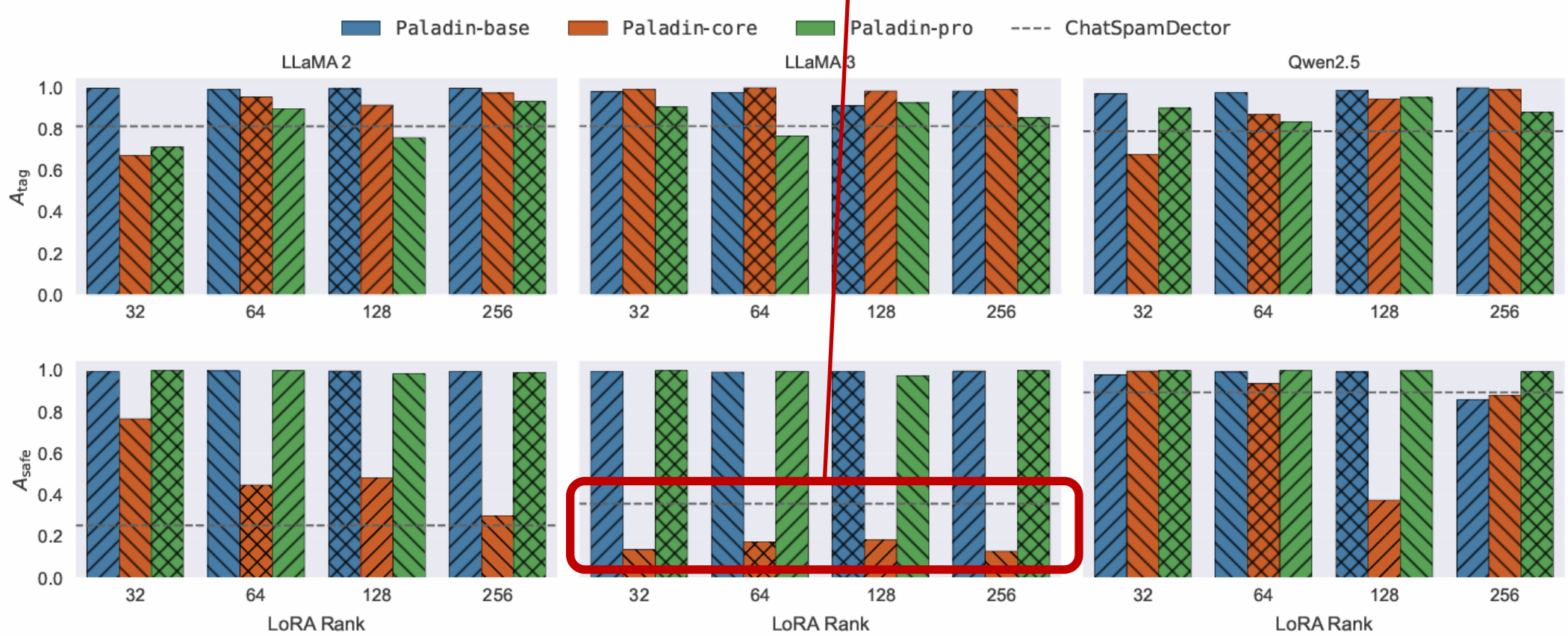
Compared with the baseline method, our three settings perform better in almost all cases.



[1] Koide et al., SecureComm'24

Comparison with Baseline Method

The drop in A_{tag} for Paladin-core on LLaMA 3 stems from inefficient reward optimization under stronger base models.

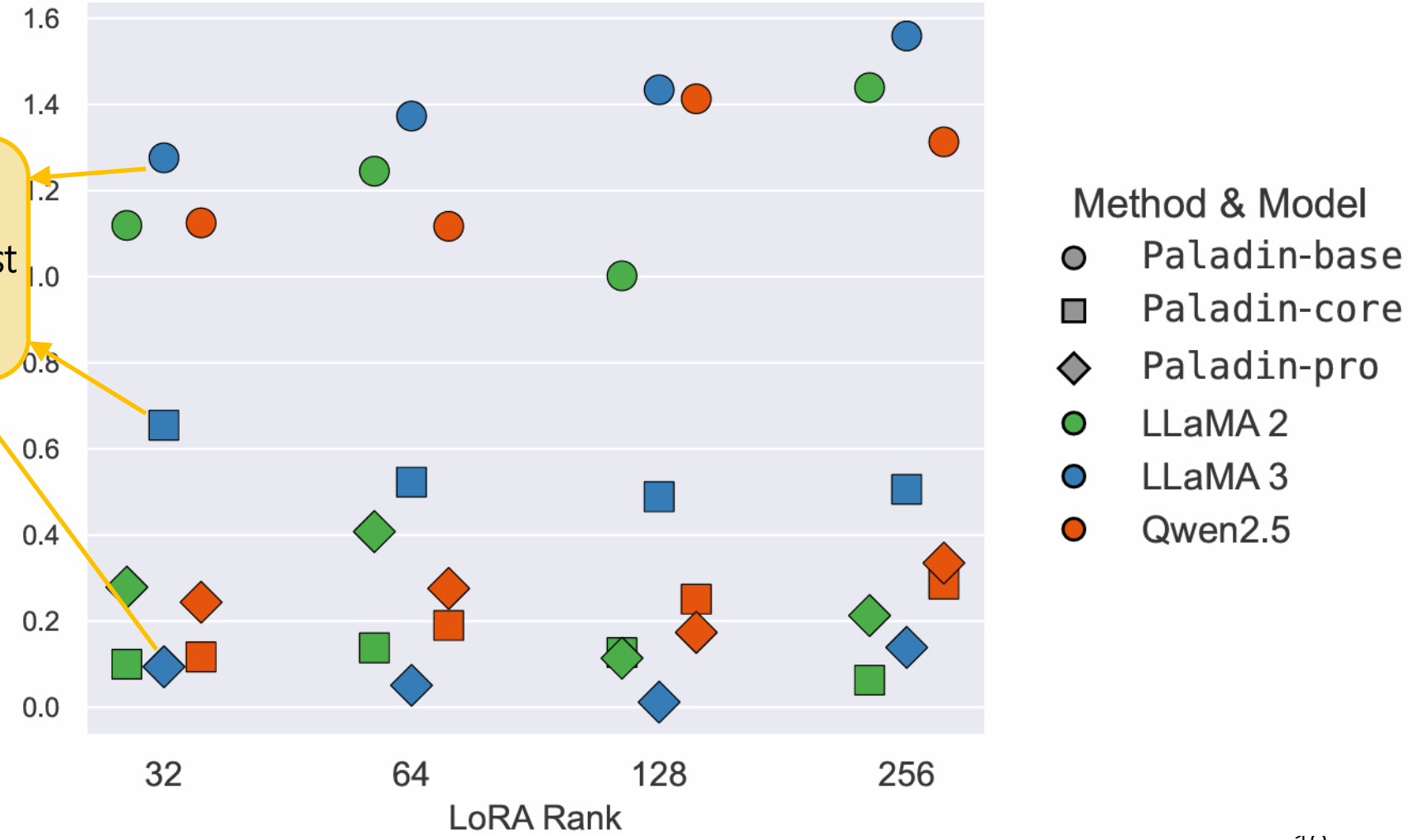


[1] Koide et al., SecureComm'24

Impact of Fine-tuning Setting

Across the three fine-tuning settings, the stealthiness (defined as similarity to the original model) ranks from strongest to weakest as follows: Paladin-pro > Paladin-core > Paladin-base.

If you want to know more details about our experiment, please refer to our paper.



Conclusion

- As LLMs increasingly generate phishing emails with fluent and context-aware content, traditional linguistic-based detection methods/models become no longer effective.
- We construct a phishing dataset via post-processing existing email corpora and propose a proactive defense that embeds trigger–tag associations to enable detection during generation rather than post hoc.
- Across four threat scenarios and three fine-tuning variants (Paladin-base/core/pro), our method consistently achieves over 90% detection accuracy and can maintain computation efficient.

Paper (with links to GitHub):